2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Online Quantum Mixture Regression for Trajectory Learning by Demonstration

Dimitrios Korkinof and Yiannis Demiris

Abstract— **In this work, we present the online Quantum Mixture Model (oQMM), which combines the merits of quantum mechanics and stochastic optimization. More specifically it allows for quantum effects on the mixture states, which in turn become a superposition of conventional mixture states. We propose an efficient stochastic online learning algorithm based on the online Expectation Maximization (EM), as well as a generation and decay scheme for model components. Our method is suitable for complex robotic applications, where data is abundant or where we wish to iteratively refine our model and conduct predictions during the course of learning. With a synthetic example, we show that the algorithm can achieve higher numerical stability. We also empirically demonstrate the efficacy of our method in well-known regression benchmark datasets. Under a trajectory Learning by Demonstration setting we employ a multi-shot learning application in joint angle space, where we observe higher quality of learning and reproduction. We compare against popular and well-established methods, widely adopted across the robotics community.**

I. Introduction

Robot learning by demonstration (LbD) has been a field of vibrant research for several years now. It originally attracted a great deal of attention as a highly promising means of teaching robots new skills [1], [2], [3]. Tasks were previously manually programmed and predefined, an endeavor which can be rather tedious and time-consuming. Task demonstration methods include guiding, tele-operating, vision [4], motion capturing and kinesthetics [5] (manually moving the robot joints in place).

The objective of LbD entails severe difficulties, for that reason a variety of methods from different fields need to be employed, namely methods stemming from machine learning, computer vision, human-robot interaction. There are two popular approaches to the problem in question [6]: trajectory level and symbolic level task encoding. The former being a lower and the latter a higher level approach.

Statistical Machine learning has been a popular approach in robotics, with valuable contribution to LbD as well [7], [8], [9]. This can be mainly attributed to the inherent ability of statistical algorithms to not only train from the data, but also generalize learned tasks. Additionally, such methods perform prediction by means of a full predictive distribution, rather than point estimates, thus enabling us to also assess the uncertainty of prediction.

Two well-established approaches towards trajectory level LbD are the Gaussian mixture regression (GMR) [10], [11] and the locally weighted projection regression (LWPR) [12]. Regarding the comparative merits of both methods, it has been shown that GMR performs better for low dimensional demonstrations [13], while LWPR should be preferred for inputs of high dimensionallity, which lie in lower dimensional manifolds, and/or inputs that may contain irrelevant dimensions.

Several researchers have previously drawn inspiration from the influential ideas of quantum physics and probability, with a variety of advances in machine learning stemming from this source of inspiration. A notable approach is the quantum mixture model, first introduced in [14]. The model was later extended for quantum mixture regression (QMR) in [15] and has proven very effective in a variety of learning by demonstration applications, yielding higher performances with a small increase in computational cost.

Online and big-data solutions are also increasingly popular in the past few years, with a rekindled interest in stochastic optimization [16], [17]. Online versions of complex algorithms however still remain a formidable challenge. In this regard, we believe that the field of robotics could highly benefit from recent advances in stochastic optimization and that online methods could provide the momentum needed towards more effective real-life robotic applications. This belief is reinforced by an increasing amount of work towards formulating online algorithms for robotic applications, such as learning robot dynamics [18] and kinematics [19], [20].

Motivated by the aforementioned points, we present a novel online training algorithm for the quantum mixture model, which we shall dub the oQMM. The proposed approach builds upon recent advances in several fields, such as machine learning, quantum statistics and stochastic optimization, to yield a powerful framework for incremental learning and prediction from multiple demonstrations. We also present an effective component production and pruning scheme to facilitate learning in cases when the data depart form the i.i.d.[1] assumption.

We demonstrate the efficacy of our algorithm in a synthetic example, a series of benchmark datasets and a learning by demonstration task.

II. Theoretical Background

A. Quantum Statistics

The generalization of conventional probability theory has given rise to a whole new field of mathematics with particular applicability in physics, namely the field of quantum

The authors are with the Department of Electrical and Electronic Engineering, Imperial College London. Exhibition Road, South Kensington Campus, SW7 2BT, London, UK. E-mail: {d.korkinof10, y.demiris}@imperial.ac.uk

[1] iid: independent and identically distributed.

978-1-4673-6357-0/13/$31.00 ©2013 IEEE

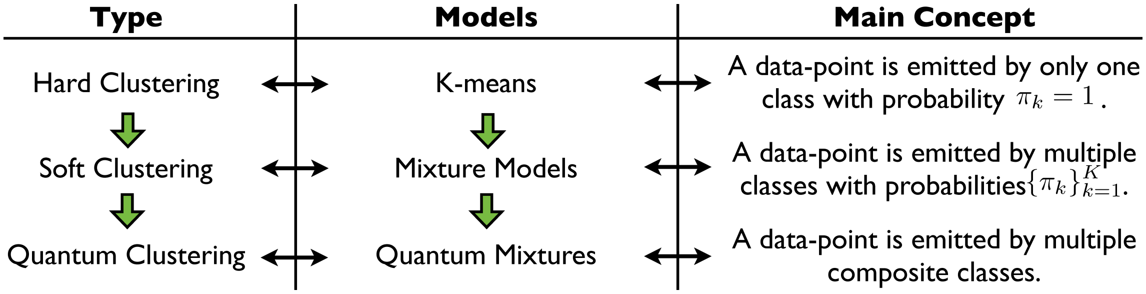

Type
Models
Main Concept
Hard Clustering
K-means
A data-point is emitted by only one class with probability $\pi_k = 1$.
Soft Clustering
Mixture Models
A data-point is emitted by multiple classes with probabilities $\{\pi_k\}_{k=1}^{K}$.
Quantum Clustering
Quantum Mixtures
A data-point is emitted by multiple composite classes.

Fig. 1: Model evolution in time (vertical arrows) and the main underlying concepts associated with each one.

statistics. According to the notion of quantum probability, a classical probability density can be generalized by a density matrix, let us denote it as Ψ, with the following properties:

- $\boldsymbol{x}^T \Psi \boldsymbol{x} \geq 0, \ \forall \boldsymbol{x}$: Positive semi-definite.
- $\Psi = \Psi^{\dagger}$: Hermitian (or self-adjoint)[2].
- $tr\left\{\Psi\right\} = 1$: Normalized.

For instance, a conventional probability density of an event u having K distinct outcomes with probability $p(u=k) = \pi_k$ can be described by the following diagonal probability matrix.

$$\Psi = diag\left([\pi_1, ..., \pi_k]\right) = \sum_{k=1}^{K} \pi_k \boldsymbol{e}_k \boldsymbol{e}_k^T \tag{1}$$

with $\{\boldsymbol{e}_k\}_{k=1}^{K}$ being a set of basis vectors of pure states, such as:

$$[\boldsymbol{e}_k]_i = \begin{cases} 1, & i = k \\ 0, & i \neq k \end{cases}$$

and $[\boldsymbol{e}_k]_i$ is the i^{th}element of vector $\boldsymbol{e}_k$.

In quantum statistics we are able to extend the probability matrix Ψ so as to allow the manifestation of non-diagonal elements. This, in turn gives rise to composite states, formed as a superposition of the system's pure states:

$$\Psi = \sum_{k=1}^{K} \pi_k \boldsymbol{u}_k \boldsymbol{u}_k^T$$

where a basis vector $\boldsymbol{u}_k = \left[\begin{array}{ccc} \frac{\sqrt{2}}{3} & \frac{2}{3} & \frac{\sqrt{3}}{3} \end{array}\right]^T$, would mean that this state corresponds to a mixture of the three system pure states with probabilities $\frac{2}{9}$, $\frac{4}{9}$ and $\frac{3}{9}$ respectively.

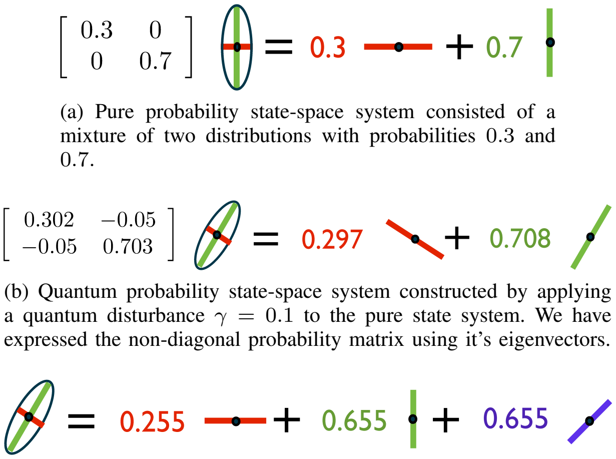
1) Numerical Example: Let us suppose a conventional mixture of two distributions with probability vector:

$$\left[\begin{array}{cc} \pi_1 & \pi_2 \end{array}\right]^T = \left[\begin{array}{cc} 0.3 & 0.7 \end{array}\right]^T$$

Under a quantum statistical perspective the system's probability matrix is the following:

$$\left[\begin{array}{cc} \pi_1 & 0 \\ 0 & \pi_2 \end{array}\right] = \pi_1 \boldsymbol{e}_1 \boldsymbol{e}_1^T + \pi_2 \boldsymbol{e}_2 \boldsymbol{e}_2^T$$

where $\boldsymbol{e}_1 = \left[\begin{array}{cc} 1 & 0 \end{array}\right]^T$ and $\boldsymbol{e}_2 = \left[\begin{array}{cc} 0 & 1 \end{array}\right]^T$.


$\left[\begin{array}{cc} 0.3 & 0 \\ 0 & 0.7 \end{array}\right]$
= 0.3
+ 0.7
$\left[\begin{array}{cc} 0.302 & -0.05 \\ -0.05 & 0.703 \end{array}\right]$
= 0.297
+ 0.708
= 0.255
+ 0.655
+ 0.655

(a) Pure probability state-space system consisted of a mixture of two distributions with probabilities 0.3 and 0.7.

(b) Quantum probability state-space system constructed by applying a quantum disturbance $\gamma = 0.1$ to the pure state system. We have expressed the non-diagonal probability matrix using it's eigenvectors.

(c) Alternative view of the same non-diagonal probability matrix, in this case not by means of it's eigenvectors. It should be noted that the number of alternative representation is unbounded.

Fig. 2: Numerical illustrative example of the relation between quantum and conventional statistics.

Introducing an off-diagonal quantum disturbance $\gamma = 0.1$ to the pure $\log$ probability matrix we observe the following effect:

$$A = -\left[\begin{array}{cc} 1.204 & 0.1 \\ 0.1 & 0.357 \end{array}\right] \Rightarrow e^A = \left[\begin{array}{cc} 0.302 & -0.05 \\ -0.05 & -0.703 \end{array}\right]$$

It should be noted that adding non-diagonal elements in that manner always results in positive semi-definite probability matrices, due to the fact that A is symmetric.

The latter quantum state system can be analyzed with respect to the probability matrix eigenvectors (fig. 2b) as follows:

$$\begin{aligned} e^A &= 0.297 \left[\begin{array}{c} -0.993 \\ -0.116 \end{array}\right] \left[\begin{array}{c} -0.993 \\ -0.116 \end{array}\right]^T \\ &+ 0.708 \left[\begin{array}{c} 0.115 \\ -0.993 \end{array}\right] \left[\begin{array}{c} 0.115 \\ -0.993 \end{array}\right]^T \end{aligned}$$

The resulting system is consisted of two quantum classes, with probabilities 0.297 and 0.708. Each quantum class is composite and formed by linear superposition of the system's pure classes with probability vectors $\left[\begin{array}{cc} 0.987 & 0.013 \end{array}\right]^T$ and

[2] With † we denote the conjugate transpose of a matrix.

$\begin{bmatrix} 0.013 & 0.987 \end{bmatrix}^T$. Of course, the probability matrix can be also expressed in any other alternative way (fig. 2c).

Concluding the quantum statistical extension of the conventional mixture model can also be viewed as an elegant mixture of mixtures. Succeeding the concepts of hard clustering (k-means algorithm) and the soft clustering (mixture model), the quantum extension can be viewed as the next evolutionary step (fig. 1).

B. Quantum Mixture Regression

The concept of quantum Gaussian mixture models, first presented in [14], involves the introduction of quantum states, which are a superposition of pure states. More specifically, let us consider the following matrices, corresponding to the conventional mixture model:

$$\begin{aligned} \boldsymbol{F} &= -diag\left([\ln \pi_1, ..., \ln \pi_K]\right) && (2) \\ \boldsymbol{G}(\boldsymbol{y}_n) &= diag\left([\ln f_1(\boldsymbol{y}_n), ..., \ln f_K(\boldsymbol{y}_n)]\right) && (3) \end{aligned}$$

where $\{\pi_k\}_{k=1}^{K}$ is a normalized set of probabilities, $\boldsymbol{Y} = \{\boldsymbol{y}_n\}_{n=1}^{N}$, $\boldsymbol{y}_n \in \mathbb{R}^D$ is a set of multidimensional observations and $\{f_k(\cdot)\}_{k=1}^{K}$ is some set of appropriate density functions.

Then the log-likelihood of the model is given as follows:

$$\boldsymbol{H}(\boldsymbol{y}_n) = \boldsymbol{F} - \boldsymbol{G}(\boldsymbol{y}_n)$$

Introducing quantum effects to the density matrix $\boldsymbol{F}$ by means of equal non-diagonal elements γ, yields the following generalization towards a quantum mixture model:

$$\boldsymbol{F} = -\begin{bmatrix} \ln \pi_1 & \cdots & \gamma \\ \vdots & \ddots & \vdots \\ \gamma & \cdots & \ln \pi_K \end{bmatrix} \tag{4}$$

$$\begin{aligned} \boldsymbol{H}(\boldsymbol{y}_n) &= -\sum_{k=1}^{K}\sum_{k'=1}^{K} B_{kk'}^{(n)} \boldsymbol{X}_{kk'} = \\ &- \begin{bmatrix} \ln(\pi_1 f_1(\boldsymbol{y}_n)) & \cdots & \gamma \\ \vdots & \ddots & \vdots \\ \gamma & \cdots & \ln(\pi_K f_K(\boldsymbol{y}_n)) \end{bmatrix} \end{aligned} \tag{5}$$

$$B_{kk'}^{(n)} = \begin{cases} \ln\left(\pi_k f_k(\boldsymbol{y}_n)\right) & , k = k' \\ \gamma & , otherwise \end{cases} \tag{6}$$

$$(\boldsymbol{X}_{kk'})_{ij} = \begin{cases} 1 & , i = j \\ 0 & , otherwise \end{cases} \tag{7}$$

The likelihood of the data under the aforementioned model is given by:

$$\mathcal{L}(\boldsymbol{Y}) = p\left(\boldsymbol{Y} \middle| \{\pi_k, \Theta_k\}_{k=1}^{K}\right) \backsimeq \prod_{n=1}^{N} \frac{tr\left\{e^{-\boldsymbol{H}(\boldsymbol{y}_n)}\right\}}{tr\left\{e^{-\boldsymbol{F}}\right\}} \tag{8}$$

Extrema conditions are obtained by direct log-likelihood optimization as follows:

$$\pi_k \backsimeq \frac{1}{N}\sum_{n=1}^{N} \phi_{nk}, \quad \boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \phi_{nk} \boldsymbol{y}_n}{\sum_{n=1}^{N} \phi_{nk}} \tag{9}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \phi_{nk} \left(\boldsymbol{y}_n - \boldsymbol{\mu}_k\right)\left(\boldsymbol{y}_n - \boldsymbol{\mu}_k\right)^T}{\sum_{n=1}^{N} \phi_{nk}} \tag{10}$$

$$\phi_{nk} = \frac{\partial \ln tr\left\{e^{-\boldsymbol{H}(\boldsymbol{y}_n)}\right\}}{\partial B_{kk}} = \frac{tr\left\{\boldsymbol{X}_{kk} e^{-\boldsymbol{H}(\boldsymbol{y}_n)}\right\}}{tr\left\{e^{-\boldsymbol{H}(\boldsymbol{y}_n)}\right\}} \tag{11}$$

The above derivative follows directly due to linear response theory.

The quantum extension of the mixture model previously described, was recently tailored to a regression setting in [15], giving rise to quantum mixture regression (QMR). Where we were able to show it performs remarkably well at trajectory LbD, yielding state-of-the-art results under a variety of settings, namely one- and multi-shot LbD.

Predictions are made possible regarding the feature vector as consisted of a set of predictor and response variables: $\boldsymbol{y}_n = [\boldsymbol{y}_n^p, \boldsymbol{y}_n^r]$, with $\boldsymbol{y}_n^p \in \mathbb{R}^{d_p}$ and $\boldsymbol{y}_n^r \in \mathbb{R}^{d_r}$. With this regard and due to the properties of Gaussian distributions [21], the predictive posterior can be formulated as follows:

$$p\left(\boldsymbol{y}_n^r | \boldsymbol{y}_n^p; \{\pi_k, \Theta_k\}_{k=1}^{K}\right) = \mathcal{N}(\boldsymbol{y}_n^r | \widetilde{\boldsymbol{\mu}}, \widetilde{\boldsymbol{\Sigma}}) \tag{12}$$

$$\widetilde{\boldsymbol{\mu}} = \textstyle\sum_{k=1}^{K} \tau_k(\boldsymbol{y}_n^p)\left[\boldsymbol{\mu}_k^r + \boldsymbol{\Sigma}_k^{rp}\left(\boldsymbol{\Sigma}_k^p\right)^{-1}\left(\boldsymbol{y}_n^p - \boldsymbol{\mu}_k^p\right)\right] \tag{13}$$

$$\widetilde{\boldsymbol{\Sigma}} = \textstyle\sum_{k=1}^{K} \tau_k^2(\boldsymbol{y}_n^p)\left[\boldsymbol{\Sigma}_k^r - \boldsymbol{\Sigma}_k^{rp}\left(\boldsymbol{\Sigma}_k^p\right)^{-1}\boldsymbol{\Sigma}_k^{pr}\right] \tag{14}$$

where for the quantum mixture regression it holds that [15]:

$$\tau_k(\boldsymbol{y}_n^p) = \frac{tr\left\{\boldsymbol{X}_{kk} e^{-\boldsymbol{H}(\boldsymbol{y}_n^p)}\right\}}{tr\left\{e^{-\boldsymbol{H}(\boldsymbol{y}_n^p)}\right\}} \tag{15}$$

$$\begin{aligned} &H(\boldsymbol{y}_n^p) = \\ &- \begin{bmatrix} \ln \pi_1 p(\boldsymbol{y}_n^p | \boldsymbol{\mu}_1^p, \boldsymbol{\Sigma}_1^p) & \cdots & \gamma \\ \vdots & \ddots & \vdots \\ \gamma & \cdots & \ln \pi_M p(\boldsymbol{y}_n^p | \boldsymbol{\mu}_M^p, \boldsymbol{\Sigma}_M^p) \end{bmatrix} \end{aligned}$$

For robotic applications of conventional mixture regression and Dirichlet process mixture regression we refer to [8], [22], [11], where trajectory positions serve as predictor and velocities as response variables $\boldsymbol{y}_t = [\boldsymbol{x}_t, \boldsymbol{v}_t]$.

III. The Online Quantum Mixture Model

We envision an online training algorithm with the following key traits: The capacity to processes data on-the-fly, no need to store processed data-points and no need to iterate through the dataset.

To achieve this purpose we shall orient ourselves towards stochastic gradient ascent methods, known to posses the desirable traits mentioned. More specifically, averaged and second-order stochastic gradient algorithms have been shown to be asymptotically efficient even after a single pass through the training set [17]. A popular choice is normalizing the noisy gradient with the Fisher Information Matrix (FIM) [23], a quantity also known as the natural gradient, which captures the Riemannian structure of the parameter space [24].

A. Proposed Approach

A shortcoming, however, of stochastic gradient ascent is that the updates may not meet all constrains. The main problem is focused on the covariance matrix, whose positive semi-definiteness is not guaranteed at each time-step, a fact that impairs our ability to perform on-line predictions.

For that reason we shall employ a similar stochastic update algorithm, based on the EM algorithm, which guarantees that all constraints imposed are in fact met at each time-step. The online EM was first introduced in [25]. It was subsequently generalized in [26], where the authors present a proof convergence and a proposition of asymptotic equivalency to natural gradient ascent. Moreover, the algorithm performs simple and efficient updates, making good use of the sufficient statistics of the exponential family, so as to avoid redundant calculations. The online EM is summarized by the following set of equations:

$$
\begin{aligned}
M-step: &\quad \begin{cases} s_t^{(i,1)} &= \tau_{it} = p(i|\boldsymbol{y}_t;\Theta^{(t)}) \\ \boldsymbol{s}_t^{(i,2)} &= \tau_{it}\boldsymbol{y}_t \\ \boldsymbol{s}_t^{(i,2)} &= \tau_{it}\boldsymbol{y}_t\boldsymbol{y}_t^T \end{cases} \quad ,\forall i \\
Updates: &\quad \begin{cases} S_t^{(i,p)} = (1-\eta_t)S_{t-1}^{(i,p)} + \eta_t s_t^{(i,p)} \end{cases} \quad ,\forall i,p \\
E-step: &\quad \begin{cases} \pi_i = S_t^{(i,1)}, \quad \boldsymbol{\mu}_i = \frac{S_t^{(i,2)}}{S_t^{(i,1)}} \\ \boldsymbol{\Sigma}_i = \frac{S_t^{(i,3)} - {S_t^{(i,1)}}^{-1} S_t^{(i,2)} {S_t^{(i,2)}}^T}{S_t^{(i,1)}} \end{cases}
\end{aligned}
$$

where as $p(i|\boldsymbol{y}_t,\Theta^{(t)})$ we denote the responsibility of cluster i for data-point $\boldsymbol{y}_t$.

The algorithm is guaranteed to asymptotically converge, provided that the stochastic weights η_t satisfy the conditions [17]: $\sum_t^{+\infty}\eta_t = +\infty$ and $\sum_t^{+\infty}\eta_t^2 < +\infty$.

In the case of the quantum mixture, the point-wise responsibilities τ_{it} may be approximately updated using eq. 15. We have observed that this assumption works very well in practice. During our empirical evaluation, we were able to established that the proposed algorithm converges at least as fast and achieves state-of-the-art performance.

Concluding, the algorithm requires an inhibition phase, an equivalent to bootstrapping or warm-up. A period during which we update the global statistics, but suppress the M-step until sufficient information has been accumulated. The stability of each subsequent update can also be facilitated by processing the data in mini-batches, which accounts for a notable speed-up and is also reported to lead to performance gains [27].

During our empirical evaluation we have also compared against the more robust Student-t mixture model, for which we have adapted the online-EM algorithm (oSMM) accordingly. The formulation is presented in eq. 16 for reasons of completeness and since it does not, to the best of our knowledge, appear elsewhere in the bibliography.

$$
\begin{aligned}
M-step: &\quad \begin{cases} s_t^{(i,1)} &= \tau_{it} = \frac{\pi_i t(\boldsymbol{y}_t|\boldsymbol{\Theta}_i)}{\sum_{j=1}^K \pi_j t(\boldsymbol{y}_t|\boldsymbol{\Theta}_j)} \\ u_{ti} &= \frac{\nu_i + D}{\nu_i + \delta(\boldsymbol{y}_t;\boldsymbol{\mu}_i,\Sigma_i)} \\ s_t^{(i,2)} &= \tau_{ti}u_{ti} \\ \boldsymbol{s}_t^{(i,3)} &= \tau_{ti}u_{ti}\boldsymbol{y}_t \\ s_t^{(i,4)} &= \tau_{ti}\ln u_{ti} \\ \boldsymbol{s}_t^{(i,5)} &= \tau_{ti}u_{ti}\boldsymbol{y}_t\boldsymbol{y}_t^T \end{cases} \\
Updates: &\quad \begin{cases} S_t^{(i,p)} = (1-\eta_t)S_{t-1}^{(i,p)} + \eta_t s_t^{(i,p)} \end{cases} \\
E-step: &\quad \begin{cases} \pi_i = S_t^{(i,1)}, \quad \boldsymbol{\mu}_i = \frac{S_t^{(i,3)}}{S_t^{(i,2)}} \\ \boldsymbol{\Sigma}_i = \frac{S_t^{(i,5)} - {S_t^{(i,2)}}^{-1} S_t^{(i,3)} {S_t^{(i,3)}}^T}{S_t^{(i,1)}} \\ \nu_i : \ \ln\left(\frac{\nu_i}{2}\right) - \psi\left(\frac{\nu_i}{2}\right) + \psi\left(\frac{\nu_i^{n-1}+D}{2}\right) \\ -\ln\left(\frac{\nu_i^{n-1}+D}{2}\right) + \frac{S_t^{(i,4)} - S_t^{(i,2)}}{S_t^{(i,1)}} + 1 = 0 \end{cases}
\end{aligned} \tag{16}
$$

where

$$
t(\boldsymbol{y}_t|\boldsymbol{\Theta}_i) = \frac{\Gamma\left(\frac{\nu_i+D}{2}\right)|\Sigma_i|^{-\frac{1}{2}}}{(\pi\nu_i)^{\frac{D}{2}}\Gamma\left(\frac{\nu_i}{2}\right)\left[1+\nu_i^{-1}\delta(\boldsymbol{y}_t;\boldsymbol{\mu}_i,\Sigma_i)\right]^{\frac{\nu_i+D}{2}}}
$$

and ν_i are the degrees of freedom, D the dimensionality of the observations and $\delta(\cdot)$ the Mahalanobis distance.

B. Unit Manipulation (cm)

Manipulating the number of components is important in online algorithms, especially so when we depart from the assumption that the data are presented independently and identically distributed (i.i.d.) to the algorithm. An effective mechanism of component birth and decay can not only account for a substantial speed-up, by pruning unneeded components, but also for a performance boost, by adding components in areas misrepresented by the model.

Following the analysis in [25], we propose the following mechanism.

1) Component Pruning and Reset: In case the mass of a component diminishes, we need to act to remove it or reset it.

Deleting components is straightforward, however we note that it is necessary to renormalize global statistics $S^{(i,1)}$.

Resetting a component involves resetting it's mixing coefficient and covariance matrix. The mixing coefficient is set to $1/K$, where K is the number of components, and the covariance is set broad enough. We have randomly initialized the covariances to diagonal matrices drawn from a uniform distribution $\mathcal{U}[0, 0.1]$, however this is expected to differ according to the data variance.

In this case, we also perform an inverse M-step to update global statistics as follows:

TABLE I: Main hyper-parameters used in all experiments.

	M	N_m	α	γ	N_i
Synthetic	10	1	0.6	0.09	100
S.-A. [32]	30	1	0.8	0.09	50
pumadyn [33]	30	1	0.8	0.09	50
L8s (t, x)	30	30	0.8	0.09	170
L8s $(t, x) + cm$	-	30	0.85	0.09	30
L8s $(\dot{x}, x)$	30	20	0.8	0.09	170

$$\begin{array}{c} inverse \\ M-step \end{array} : \begin{cases} S_t^{(i,1)} & = \pi_i^{(new)}, \forall i \\ S_t^{(k,2)} & = S_t^{(k,1)} \boldsymbol{\mu}_k \\ S_t^{(k,3)} & = S_t^{(k,1)} \boldsymbol{\Sigma}_i + \\ & + {S_t^{(k,1)}}^{-1} S_t^{(k,2)} {S_t^{(k,2)}}^T \end{cases}$$

2) Component Production: If one or more data-points are not represented with sufficiently large likelihood by any of the existing components, we create a new one centered at the mean of the misrepresented points and with a sufficiently large default covariance.

To be more specific, a data-point is considered as misrepresented by the current mixture model, when its likelihood under all current components is below a certain threshold. Similar to the previous case, the threshold has to be chosen wisely and depends on the model. We have used a likelihood threshold of 0.001 for Gaussian densities and 1 for Student-t densities.

The initial mixing coefficient is set to $1/K$.

We should note that the stochastic weights η_t are run separately for each component so as to diminish as a component accumulates evidence.

IV. Empirical Evaluation

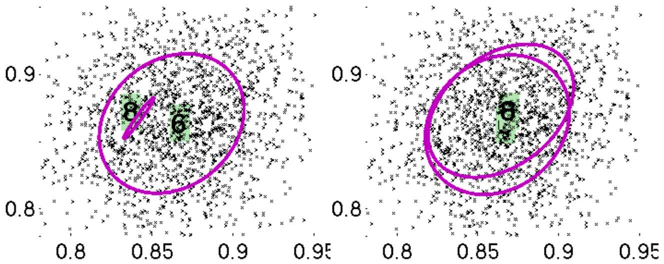
A. Implementation and Setting

Apart from the quantum mixture, we have also implemented online versions of conventional mixture models for Gaussian (oGMM) and Student-t (oSMM) densities. For the locally weighted projection regression [12], we have used the authors' LWPR library [28], [29]. For the online Gaussian Process (oGP), we utilize a matlab version of the OTL library that appears in [30], [31]. Testing was conducted in Matlab 2012b on an Ubuntu Linux PC, i7 3.4GHz, 16GB RAM.

Parameters for the LWPR and oGP were fixed to the recommended defaults. For the oGP, we have chosen a moderate number of 50 basis vectors for all experiments. All other methods, are run with common parametrization and initialization for impartial comparisons. The main hyper-parameters for each experiment are presented in Table I, where M is the number of components, N_m is the mini-batch size, α is the parameter of the stochastic coefficients $\eta_t = n^{-\alpha}$, γ is the quantum parameter and N_i is the length of the inhibition phase.

B. Synthetic Experiment

During our experimental evaluation and besides the performance gains achieved by the oQMM, we have also observed consistently higher numerical stability. This could be attributed to the more balanced component weighting induced by the quantum effects, as illustrated in the following synthetic example.



(a) Online GMM. (b) Online QMM.

Fig. 3: Component #8 has a low rank covariance matrix leading to numerical instabilities in higher dimensional datasets.

For the purpose of the experiment we have randomly generated 20 datasets of 5000, 2D data-points, sampled from an equally weighted mixture of 3 Gaussians with randomly chosen covariances and deterministically chosen means sufficiently far apart. We have repeated training over 10 independent runs for each dataset to accumulate statistics and reduce variance attributed to random initializations. The model was purposely over-specified with the number of components set to $K = 10$.

Under this setting, we observe that the oGMM tends to be more prone to producing unstable, spurious clusters with a typical case shown in fig. 3. Examining the components with regard to the largest eigenvalue of their covariance, we found that the oGMM produced 445 components with maximum eigenvalue below 10^{-4}, while the oQMM produced only 2. This effect becomes more intense in higher dimensional datasets and can lead the oGMM to numerical instability, whereas the oQMM remains less affected.

C. Benchmark Data

1) Noisy Function Approximation: The first benchmark is a non-linear function approximation problem under the presence of noise, proposed in [32] and also adopted in [25],[12].

$$y = \max\left\{e^{-10x_1^2}, e^{-50x_2^2}, e^{-5(x_1^2+x_2^2)}\right\} + \mathcal{N}(0, 0.01) \quad (17)$$

Points drawn from the noisy function, the form of the noisy surface and the original surface can be seen in the first 3 subplots of fig. 4 respectively.

We have randomly generated 20 training datasets using eq. 17, by drawing 5000 points $\boldsymbol{x}$ from a uniform distribution $\mathcal{U}[-1, 1]$. In order to account for different random initializations, we execute each of the oGMM, oSMM and oQMM 10 times for each dataset with common parametrization and initialization. The LWPR and the oGP are executed once for every dataset, as they exhibit an almost deterministic behavior.

TABLE II: Test-set mean square error.

$^{Method}/_{Dataset}$	Schaal-Atkeson [32]	`pumadyn` [33]			
		`8fm`	`8fh`	`8nm`	`8nh`
LWPR	0.00544 ± 0.0007	$\mathbf{0.0136}$	0.0887	0.0971	0.1280
oGP	0.04070 ± 0.0079	0.0234	0.1377	0.0565	0.1432
oGMM	0.00455 ± 0.0011	0.0444 ± 0.0128	0.0942 ± 0.006	0.0819 ± 0.029	0.1145 ± 0.020
oSMM	0.00520 ± 0.0009	0.0383 ± 0.0117	0.0970 ± 0.007	0.0683 ± 0.014	0.1068 ± 0.011
oQMM	$\mathbf{0.00402 \pm 0.0012}$	0.0216 ± 0.0027	$\mathbf{0.0886 \pm 0.003}$	$\mathbf{0.0512 \pm 0.010}$	$\mathbf{0.0934 \pm 0.005}$

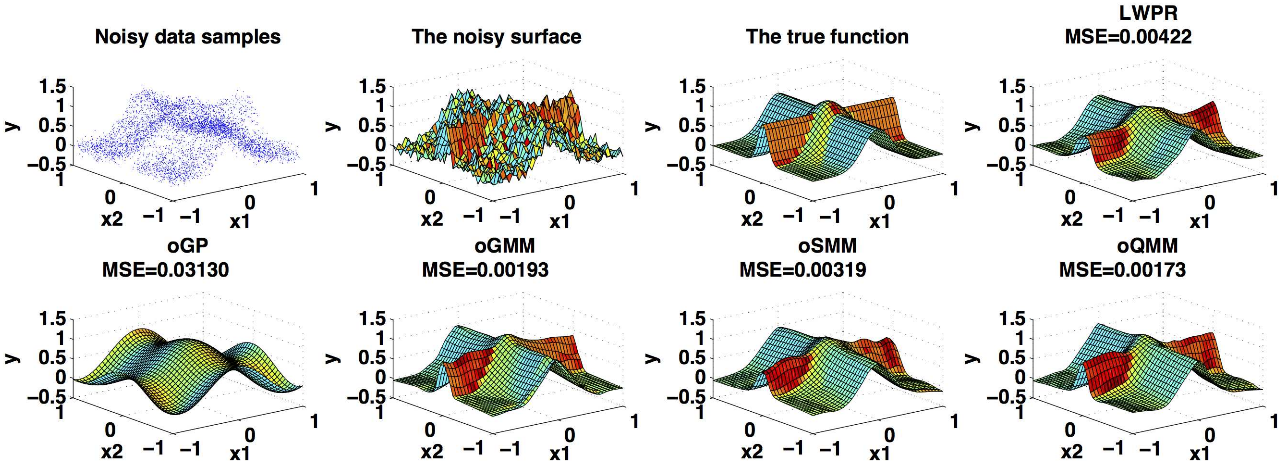


Fig. 4: Best surfaces for all evaluated methods.

The accumulated statistics are presented in Table II. By applying the student-t test we have established that the differences of all presented results are statistically significant at the confidence level of 5% or less. The best predicted surface for each evaluated method is presented in fig. 4.

The quantum mixture achieves the best results in this experiment, around 13% better than the conventional mixture, 29% better than the Student-t mixture and over 35% better than LWPR. The online Gaussian process performs poorly in this experiment.

2) Puma Robot Dynamics: For our second experiment we consider 4, 9-dimensional datasets from the well-known Puma robot dynamics benchmark (`pumadyn`) [33]. The task is to learn the simulated forward dynamics of a Puma 560 robot arm.

Specifically, the 8 first dimensions are used as inputs, consisted of joint angular positions and velocities for 3 links and torque values for two joints. The last dimension is the target variable and represents the angular acceleration of the third link. The name of each dataset starts with an integer indicating dimensionality, followed by two letters denoting the non-linearity and noise levels respectively (with `f` standing for "fairly linear", `n` "non-linear", `m` "medium noise" and `h` "high noise").

Out of 8192 available data-points, we use the first 7192 for training and the rest for testing. We have executed 50 repetitions of the oGMM, oSMM and oQMM to accumulate statistics and account for random initialization. The LWPR and oGP were executed once for each dataset, as their performance is almost deterministic.

Our results can be seen in Table II. We can observe that LWPR proves especially effective in fairly linear datasets, regardless of their noise levels, while the oGP is effective in cases of moderate noise, regardless of the level of non-linearity. On the contrary, the oQMM seems to perform well in both aforementioned cases, exhibiting invariant performance regardless the non-linearity or noise level, thus constituting a more generally applicable method. It should be noted that at the least challenging dataset `8fm`, LWPR performs better than the oQMM. However, in all remaining datasets the oQMM achieves equivalent or superior results to all rival methods.

TABLE III: Test-set and trajectory reconstruction mean square errors. (10^{-3})

	(t, x)	$(t, x) + cm$	$(x, \dot{x})$
LWPR	4.221		3.558
oGP	5.947		4.774
oGMM	2.902 ± 0.6	2.517 ± 0.6	2.850 ± 1.7
oSMM	2.986 ± 0.7	4.240 ± 0.1	2.214 ± 0.7
oQMM	$\mathbf{2.688 \pm 0.4}$	$\mathbf{2.296 \pm 0.4}$	$\mathbf{2.173 \pm 0.8}$

D. Case Study: Multi-Shot Trajectory Learning by Demonstration

Our case study is a multi-shot LbD task, namely drawing *lazy figure 8s*, as shown in fig. 5. The task might appear trivial, however in the high dimensional joint space it entails severe challenges for learning algorithms and for that reason is regarded a classical benchmark [12].

In our experiments, we make use of the NAO robotic

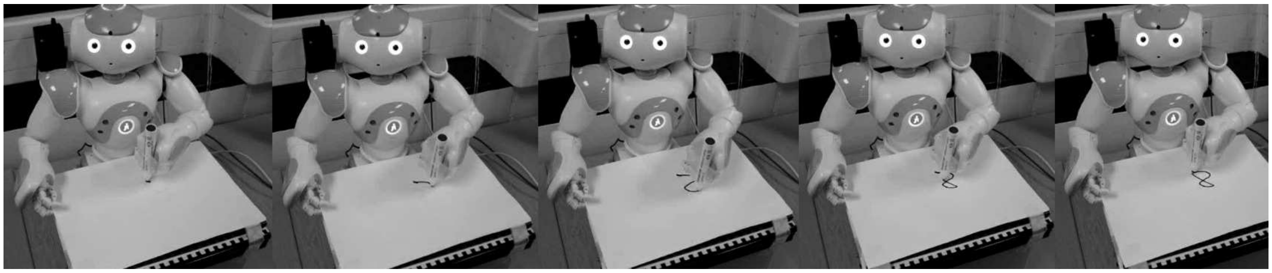
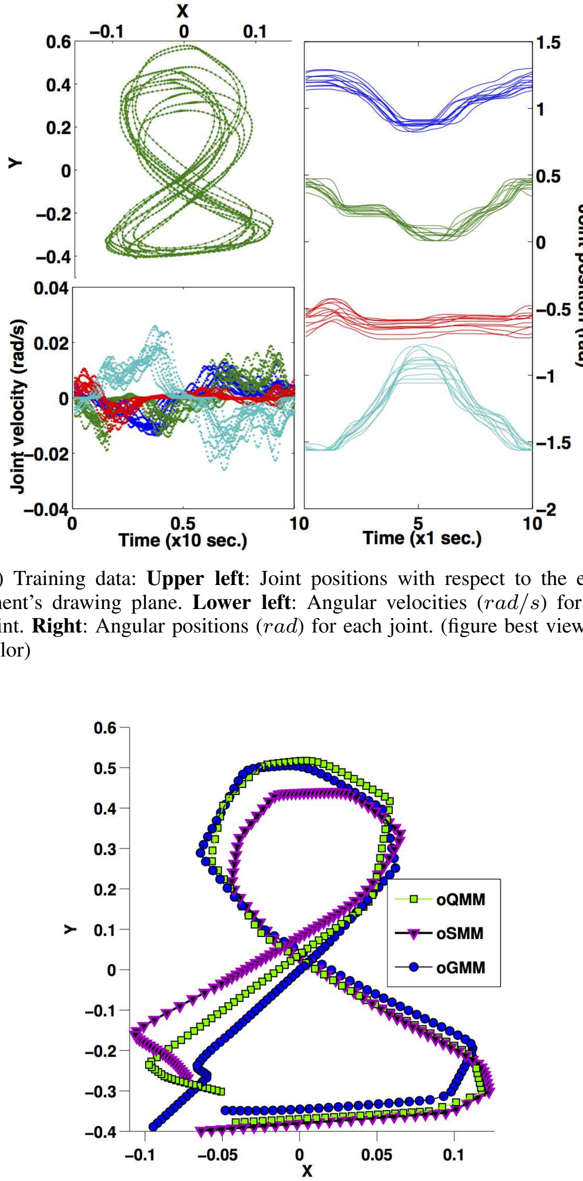


Fig. 5: Experimental setup of NAO robot drawing *lazy figure 8s*.



(a) Training data: **Upper left**: Joint positions with respect to the experiment's drawing plane. **Lower left**: Angular velocities (rad/s) for each joint. **Right**: Angular positions (rad) for each joint. (figure best viewed in color)

(b) Predicted trajectories obtained by all evaluated methods. X-Y is the drawing plane of our experiment. (figure best viewed in color)

Fig. 6: Multi-shot LbD case-study.

platform (academic edition); a humanoid robot with 27 degrees of freedom (DoF), a subset of which is employed in this case. We have obtained 12 distinct demonstrations of *lazy figure 8s* presented to the NAO robot by means of kinesthetics. Extra care has to be devoted so as each demonstration to be performed in consistent speed or alternatively the trajectories could be subjected to online time warping. Each demonstration is consisted of 170, 5-dimensional data-points: 4 joint angle positions $\boldsymbol{x}$ and the time component t. We have used the first 11 demonstrations for training and the last for testing. As can be seen in fig. 6a, the dataset is severely ridden with noise. Furthermore, each demonstrations has different points of origin and relatively few data-points. All those characteristics combined constitute a formidable challenge for any algorithm.

We consider two different learning scenarios. According to the first, the predictor variable is the time component of the demonstrated task t and joint angle positions $\boldsymbol{x}$ serve as response variables. This setting is frequently employed with Gaussian processes, where time is considered the free variable of the experiment. The second scenario is a more challenging one and consists of predictions regarding the next step velocities $\dot{\boldsymbol{x}}$, given previous step joint angle positions $\boldsymbol{x}$. This is in fact more challenging as the velocities are considerably noisier (fig. 6a) and task reproduction requires predictions of higher precision. In this case the error metric we employ is the trajectory reconstruction error, calculated from the predicted velocities.

The results, as shown in Table III, reveal that for the first scenario (t, x), the oQMM performs better than both oSMM and oGMM from around $8\% - 11\%$ and much better than LWPR and oGP. Regarding the component manipulation (cm) scheme, we have found that it performs reasonably well, achieving around 17% higher accuracy with lower computational costs.

In the case of the second scenario $(x, \dot{x})$, the best method is still the oQMM, with the oSMM also performing at the same level. In fig. 6b we can see an example of the best fit of reconstructed trajectories for the oGMM, oSMM and oQMM. We can see that the shape of the trajectory yielded by the oQMM is considerably better. Although generally achieving low reconstruction errors, the component manipulation scheme was not consistent enough in this scenario.

E. Note on complexity

The fastest method is the online Gaussian process, followed by the conventional mixture model. The oQMM is slightly more computationally intensive than the oGMM. Finally, the LWPR and especially the oSMM are the most computationally intensive, with the latter posing a severe computational burden.

V. Conclusion

In this work, we have presented the online quantum mixture model, a powerful framework for robot learning by demonstration. Our approach is based on quantum mixture regression and recent advances in stochastic optimization. We also provide a component manipulation scheme, which can result in higher performance and lower computational costs.

Our method is especially suited for large, complex datasets. It exhibits higher numerical stability and is generally applicable regardless the noise or the non-linearity level of the data. We have also shown that it performs very well in a demanding multi-shot learning by demonstration application, where it enjoys higher accuracy of prediction and trajectory reconstruction.

The implementation is available at the author's website http://www.korkinof.com and the Personal Robotics website http://www3.imperial.ac.uk/personalrobotics.

Acknowledgements

The authors would like to thank the members of the Personal Robotics Lab at Imperial College London for their support, and particularly Harold Soh for his help on online methods and the online GP.

Dimitrios Korkinof is supported by the State Scholarship Foundation of Greece through the EU NSRF 2007-2013. This work was also partially funded by the EU FP7 ALIZ-E project (248116).

References

[1] Y. Demiris and B. Khadhouri, "Hierarchical attentive multiple models for execution and recognition of actions," *Robotics and autonomous systems*, vol. 54, no. 5, pp. 361–369, 2006.

[2] J. Demiris and G. Hayes, "Imitation as a dual-route process featuring predictive and learning components: A biologically plausible computational model," *Imitation in animals and artifacts*, 2002.

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

[4] Y. Wu and Y. Demiris, "Towards one shot learning by imitation for humanoid robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2889–2894, IEEE, 2010.

[5] P. Kormushev, D. N. Nenchev, S. Calinon, and D. G. Caldwell, "Upper-body kinesthetic teaching of a free-standing humanoid robot," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2011.

[6] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, (Secaucus, NJ, USA), pp. 1371–1394, 2008.

[7] S. Calinon, A. Pistillo, and D. G. Caldwell, "Encoding the time and space constraints of a task in explicit-duration hidden Markov model," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2011.

[8] S. Calinon and A. Billard, "Recognition and reproduction of gestures using a probabilistic framework combining PCA, ICA, and HMM," in *Proc. Int. Conf. on Machine Learning (ICML)*, pp. 105–112, 2005.

[9] A. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 370–284, 2006.

[10] S. Calinon and A. Billard, "Statistical learning by imitation of competing constraints in joint space and task space," *Adv. Robot.*, vol. 23, no. 15, pp. 2059–2076, 2009.

[11] S. P. Chatzis, D. Korkinof, and Y. Demiris, "A nonparametric bayesian approach toward robot learning by demonstration," *Robotics and Autonomous Systems*, vol. 60, no. 6, pp. 789 – 802, 2012.

[12] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 2, pp. 2602–2634, 2005.

[13] T. Cederborg, M. Li, A. Baranes, and P. Y. Oudeyer, "Incremental local on-line gaussian mixture regression for imitation learning of multiple tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, October 2010.

[14] K. Tanaka and K. Tsuda, "A quantum-statistical-mechanical extension of Gaussian mixture model," in *Proc. Int. Workshop on Statistical-Mechanical Informatics*, vol. 95, 2007.

[15] S. Chatzis, D. Korkinof, and Y. Demiris, "A quantum-statistical approach toward robot learning by demonstration," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1371–1381, 2012.

[16] L. Bottou, *Online learning and stochastic approximations.* Cambridge University Press, Cambridge, UK, 1998.

[17] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Proc. of the Int. Conf. on Computational Statistics (COMPSTAT)*, 2010.

[18] J. de la Cruz, W. Owen, and D. Kulic, "Online learning of inverse dynamics via gaussian process regression," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2012.

[19] B. Damas and J. Santos-Victor, "An online algorithm for simultaneously learning forward and inverse kinematics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2012.

[20] A. Droniou, S. Ivaldi, V. Padois, and O. Sigaud, "Autonomous online learning of velocity kinematics on the icub: A comparative study," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2012.

[21] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[22] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression," *IEEE Robotics and Automation Magazine*, vol. 17, no. 2, pp. 44–54, 2010.

[23] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics, 2008.

[24] S. Amari, "Natural gradient works efficiently in learning.," *Neural Computation*, vol. 10, pp. 251–276, 1998.

[25] M. Sato and S. Ishii, "On-line EM algorithm for the normalized gaussian network.," *Neural computation*, vol. 12, pp. 407–32, Feb. 2000.

[26] O. Cappé and E. Moulines, "On-line expectation-maximization algorithm for latent data models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 71, no. 3, pp. 593–613, 2009.

[27] P. Liang and D. Klein, "Online EM for unsupervised models," in *Proc. of the Annual Conf. of the N. American Association for Computational Linguistics (NAACL)*, 2009.

[28] S. Klanke, S. Vijayakumar, and S. Schaal, "A library for locally weighted projection regression," *Journal of Machine Learning Research (JMLR)*, vol. 9, pp. 623–626, 2008.

[29] S. Vijayakumar and S. Klanke, *A library for Locally Weighted Projection Regression - Supplementary Documentation*.

[30] H. Soh, Y. Su, and Y. Demiris, "Online spatio-temporal gaussian process experts with application to tactile classification," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4489–4496, 2012.

[31] H. Soh and Y. Demiris, "Iterative temporal learning and prediction with the sparse online echo state gaussian process," in *Proc. of the Int. Joint Conf. on Neural Networks (IJCNN)*, pp. 1–8, 2012.

[32] S. Schaal and C. Atkeson, "Constructive incremental learning from only local information," *Neural Computat.*, vol. 10, no. 8, pp. 2047–2084, 1998.

[33] P. I. Corke, "A robotics toolbox for matlab," *IEEE Robotics & Automation Magazine*, vol. 3, no. 1, pp. 24–32, 1996.