In Proceedings of International Conference on Artificial Neural Networks (ICANN 2006), Athens, Greece, 2006.

Learning Multiple Models of Non-Linear Dynamics for Control under Varying Contexts

Georgios Petkos, Marc Toussaint, and Sethu Vijayakumar

Institute of Perception, Action and Behaviour, School of Informatics
University of Edinburgh EH9 3JZ

Abstract. For stationary systems, efficient techniques for adaptive motor control exist which learn the system's inverse dynamics online and use this single model for control. However, in realistic domains the system dynamics often change depending on an external unobserved context, for instance the work load of the system or contact conditions with other objects. A solution to context-dependent control is to learn multiple inverse models for different contexts and to infer the current context by analyzing the experienced dynamics. Previous multiple model approaches have only been tested on linear systems. This paper presents an efficient multiple model approach for non-linear dynamics, which can bootstrap context separation from context-unlabeled data and realizes simultaneous online context estimation, control, and training of multiple inverse models. The approach formulates a consistent probabilistic model used to infer the unobserved context and uses Locally Weighted Projection Regression as an efficient online regressor which provides local confidence bounds estimates used for inference.

1 Introduction

Learning dynamics for control is essential in situations where analytical derivation of the plant dynamics is not feasible. This can be either due to the complexity of the system or due to lack of or inaccurate knowledge of the physical properties of the system being controlled. Adaptive control is an established research area that has offered a multitude of methods that can be used in such cases. However, the dynamics of the environment that the system has to interact with or even of the system itself are often changing in a rapid or discontinuous fashion. For example, a robot arm may be required to manipulate objects of different weights – an instantiation of control under multiple contexts. In these cases, classic adaptive control methods are inadequate since they result in large errors and instability during the period of adaptation. Furthermore, if the dynamics change back and forth, readapting everytime is a suboptimal and inefficient strategy.

Humans do not have difficulty controlling their limbs under different contexts. It has been suggested that they achieve this by using not just one model that is constantly adapted to new environments, but a set of models, each of which is appropriate for a different environment [1]. The key issue that needs to be resolved for this multiple model paradigm is that at any time the current context needs to be determined; this will be referred to as the context estimation problem and is central to this work. Context estimates are needed both during training and control, i.e., for deciding which model should be used for control and which model should be trained with the data experienced. Biological systems (e.g. humans) estimate contexts using a variety of sensory information like

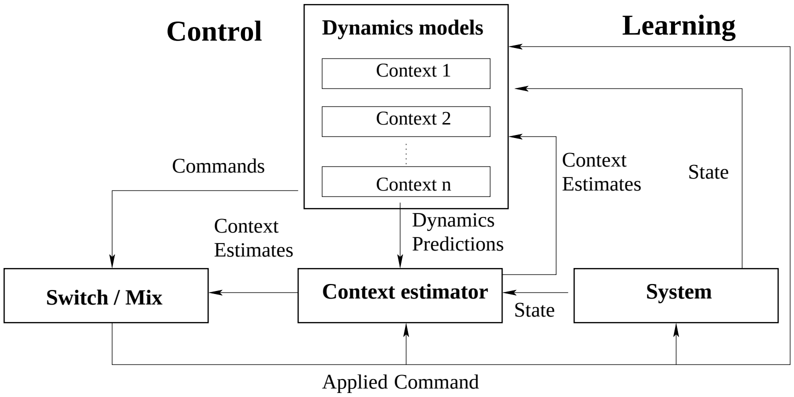

Fig. 1. Typical setup of a multiple model paradigm for control

vision or tactile input. In artificial systems though, the available sensory information may be much poorer and the context has to be estimated from the experienced dynamics only. Our approach will formulate a proper probabilistic model that represents the context as a latent switching variable. This model allows us to estimate the context online based only on the learned inverse models using a Markovian filtering. Further, an Expectation-Maximization procedure is used to bootstrap the distinction of contexts from context-unlabeled data.

There are some existing paradigms that implement the multiple model approach: Multiple Model Switching and Tuning (MMST) [4, 5], Multiple Paired Forward and Inverse Models (MPFIM) [3] and Modular Selection and Identification for Control (MOSAIC) [2]. Fig. 1 shows the typical setup of a multiple model paradigm, where a set of different context dynamics models is maintained. In most existing approaches, the dynamics models for each context is a pair consisting of a forward and an inverse model. *Context estimation* is performed by comparing the observed dynamics of the system with the dynamics predicted by each context's forward model. For control purposes, one can either switch between commands predicted by the most likely context or mix them. Similarly, context estimates can be used for 'hard' or 'soft' assignment of data for training the most likely contexts. The most general of the mentioned paradigms is MOSAIC, which is an extension of MPFIM. MOSAIC uses mixing instead of switching, with the hope that more contexts can be handled with a smaller number of models. This seems plausible in the case of linear dynamics and indeed MOSAIC has been realized only for linear systems. Real robotic systems are highly non-linear, requiring the ability to learn online and adjust model complexity in a data-driven manner. Existing multiple model approaches are, therefore, not scalable. In this paper we present a non-linear multiple model approach to control based on an efficient non-linear online learning algorithm (LWPR) that addresses these requirements. To the best of our knowledge, this is the first multiple model study that manages to learn non-linear dynamics under multiple contexts with online separation of contextual data.

2 Adaptive non-linear control with LWPR

Let us first consider the single context scenario of learning the dynamics of a system (e.g., a robot) and using them for control. At time step t, let Θ_t be the state of the system (which include the position and velocity components) and τ_t the control signal. A deterministic forward model f describes the discrete-time system dynamics as

$$\Theta_{t+1} = f(\Theta_t, \tau_t) \ . \tag{1}$$

Learning a forward model f of the dynamics is useful for predicting the behavior of the system. However, for control purposes, an inverse model is needed. The inverse model g maps from transitions between states to the control signal that is needed to achieve this transition:

$$\tau_t = g(\Theta_t, \Theta_{t+1}) \ . \tag{2}$$

A probabilistic graphical model representation of the forward and inverse model is shown in Fig. 2(a) and Fig. 2(c), respectively.

Idealistically, an accurate inverse model can be used to exactly follow a sequence of transitions that form a desired trajectory of the system. However, given only an approximate inverse model, the error in following the trajectory may accumulate and become unacceptably large. A standard approach for control with an approximate inverse model is to combine it with a conventional linear feedback controller that counteracts the deviation from the desired trajectory. Given a desired trajectory $\Theta^*_{1:T}$ and the true state Θ_t, the composite control command at time t is

$$\tau_t = g(\Theta^*_t, \Theta^*_{t+1}) + A\ (\Theta^*_t - \Theta_t) \ , \tag{3}$$

where A is a gain matrix. We will use this composite control with gains based on the Proportional Derivative (PD) control law. One effect of the composite control approach is that the more accurate the inverse model g, the smaller are the errors and the error-correcting PD control signals. Thus, the total amount of feedback control is a measure of the accuracy of the inverse ’predictive’ model.

To learn the inverse dynamics we need a *non-linear, online* regression technique which also provides error bounds that we may use for context separation. We use the Locally Weighted Projection Regression (LWPR) [6] – an algorithm which is extremely robust and efficient for incremental learning of non-linear models in high dimensions. A LWPR model consists of a set of local linear models that come paired with a kernel that defines the locality of the model. For a given input x, the kernel of the k-th local model determines a weighting $w_k(x)$ while the local linear model predicts an output $\psi_k(x)$. The combined prediction of LWPR is

$$\phi(x) = \frac{1}{W} \sum_k w_k(x)\ \psi_k(x) \ , \quad W = \sum_k w_k(x) \ . \tag{4}$$

Each locality kernel $w_k(x)$ has a parametric Gaussian form and the distance metric is adapted during learning in a data driven manner. The local models are trained using an online variant of Partial Least Squares using the collected sufficient statistics. LWPR is incremental and non-parametric in the sense that new local models are added when training proceeds and new areas of the input domain are explored.

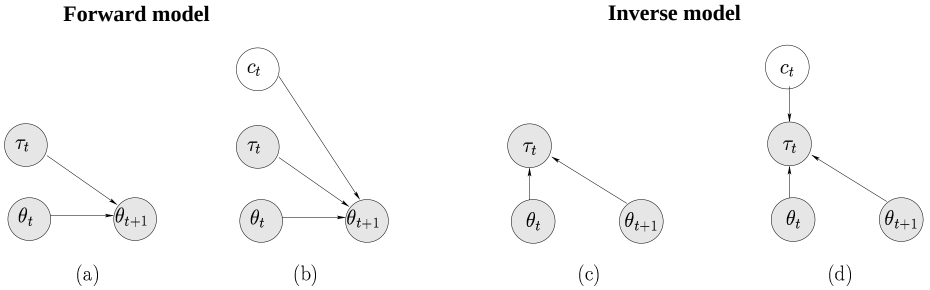

Fig. 2. Graphical model representation of the: (a) Forward model (c) Inverse model and (b,d) their respective context augmented models

The role of LWPR in the probabilistic inverse model of Fig. 2 can be summarized in the equation:

$$P(\tau \mid \Theta_{t+1}, \Theta_t) = \mathcal{N}(\phi(\Theta_{t+1}, \Theta_t),\ \sigma(\Theta_{t+1}, \Theta_t)), \tag{5}$$

whose $\phi(\Theta_{t+1}, \Theta_t)$ is a learned LWPR regression mapping desired state transitions to torques. Here, we have two options for choosing the variance: (1) we can assume a fixed noise level independent of the context and the input; (2) we can use the confidence bounds provided by each LWPR model which also depends on the current input (Θ_{t+1}, Θ_t). We will test both cases in our experiments. Please see [6] for more details on LWPR and the input dependent variance estimate.

3 Learning Multiple Models for multiple contexts

In the multiple context scenario, we assume that instead of having a single forward and inverse dynamics (Fig. 2(a,c)), the dynamics depend on an unobserved random variable c_t, the context. Fig. 2(b,d) illustrates this situation as augmented graphical models for the forward and inverse models. We assume a discrete context variable and maintain separate LWPR models to represent the inverse dynamics for each context. Thus, Eq.5 becomes:

$$P(\tau \mid \Theta_{t+1}, \Theta_t, c_t{=}i) = \mathcal{N}(\phi_i(\Theta_{t+1}, \Theta_t),\ \sigma_i(\Theta_{t+1}, \Theta_t))\ . \tag{6}$$

The problem we face in the context of adaptive online control is twofold: (1) Given a batch of yet unlabeled data and a set of yet untrained inverse models, we have to bootstrap the specialization of inverse models to different parts of the data while at the same time associating different data points to different contexts– we call this problem *data separation*; (2) Given a set of already trained inverse models and previous observations, we have to estimate the current context in order to choose the right inverse model in calculating the control signal– we call this problem *context estimation*. These problems are very closely related. We first address the simpler context estimation problem before discussing data separation.

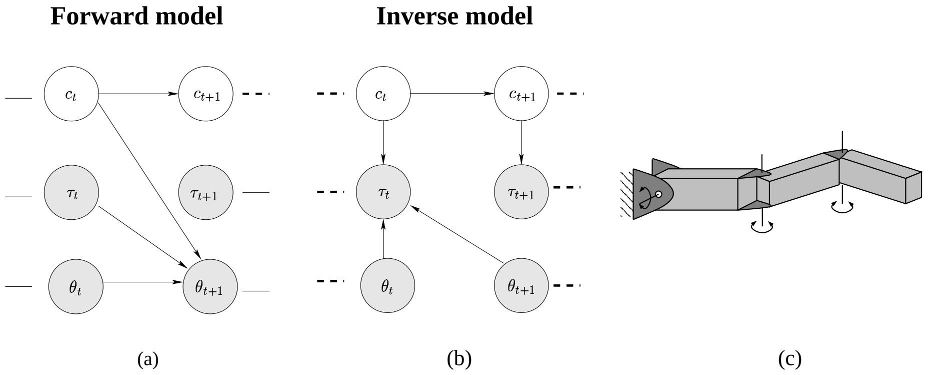

Fig. 3. Multiple model with temporal contextual dependencies using: (a) forward model or (b) inverse model for context estimation. (c) Schematic of the simulated 3-link robot arm

3.1 Context Estimation

In general, context estimation with a given set of models is performed comparing the predictions of each model with the observed dynamics. Usually this is done by comparing a set of trained forward models with the observed dynamics. However, the predictions of inverse models can equally be compared with the observed dynamics and thus, there is no need to learn additional forward models. Our viewpoint is that at each time step t we "observe" a state transition and an applied torque signal summarized in the triplet $(\Theta_t, \Theta_{t+1}, \tau_t)$, i.e., we have access to the true applied control command (which was generated via composite control) as part of the observation. To estimate the latent context variable c_t (without yet exploiting the temporal dependency) we can compute $P(c_t \,|\, \Theta_t, \Theta_{t+1}, \tau_t)$, i.e., the probability of being in a context given the observed transition between two consecutive states and the command that resulted in this transition. Using Bayes rule, we get

$$P(c_t\!=\!i \,|\, \Theta_t, \Theta_{t+1}, \tau_t) = P(\tau_t \,|\, c_t\!=\!i, \Theta_t, \Theta_{t+1}) \, \frac{P(c_t\!=\!i)}{P(\tau_t \,|\, \Theta_t, \Theta_{t+1})} \,. \tag{7}$$

Here, we used $P(c_t\!=\!i \,|\, \Theta_t, \Theta_{t+1}) = P(c_t\!=\!i)$, which is the context prior. Assuming a uniform prior, the RHS quotient is a normalization factor independent of the context i. Hence, the responsibility $P(c_t = i \,|\, \Theta_t, \Theta_{t+1}, \tau_t)$ is proportional to the i-th model likelihood (eq.6).

It is straight-forward to extend this to take a Markovian dependency between contexts into account: intuitively, we would expect that in most practical cases, the context would stay the same most of the time and switch only occasionally. For instance, in our current experiments we apply control signals at 100Hz and we expect that the frequency of context switches will be much lower. Thus, including the temporal dependency between contexts $P(c_{t+1} \,|\, c_t)$, the graphical models in Fig. 2(b,d) can be reformulated as the Dynamic Bayesian Networks shown in Fig. 3(a,b). Application of standard HMM techniques is straightforward by using eq.7 as the observation likelihood in the HMM, given the hidden state $c_t\!=\!i$.

A low transition probability penalizes too frequent transitions and using smoothing or Viterbi alignment produces more stable context estimates. In the experiments, we will assume a fixed transition matrix $P(c_t = j \,|\, c_t = i)$ with high value .999 for $i = j$ and .001 otherwise and use the HMM model only for filtering or smoothing, depending on whether we investigate an online or batch estimation scenario, respectively.

3.2 Data separation

In existing multiple model approaches, separation of data for learning happens online. The predictions of the models are compared with the observed behaviour of the system to give context estimates and train the models online. However, to get these context estimates we need a mechanism for getting relatively accurate (initial) models to bootstrap the context estimation procedure. Most of the existing multiple model paradigms do not give a satisfying answer to this issue. MMST assumes that relatively good models are available from the beginning, whereas MPFIM does not address this issue at all.

The problem of bootstrapping the context separation from context-unlabeled data is very similar to clustering problems using mixture of Gaussians. In fact, the context variable can be interpreted as a latent mixture indicator and each inverse model contributes a mixture component to give rise to the mixture model of the form $P(\tau_t \,|\, \Theta_t, \Theta_{t+1}) = \sum_i P(\tau_t \,|\, \Theta_t, \Theta_{t+1}, c_t = i)\; P(c_t = i)$. Clustering with mixtures of Gaussians is usually trained using Expectation-Maximization (EM), where initially the data are labeled with random responsibilities (are assigned randomly to the different mixture components). Then every mixture component is trained on its assigned (weighted) data (M-step) and afterwards the responsibilities for each data point is recomputed by setting them proportional to the likelihoods for each mixture component (E-step). Iterating this procedure, each mixture component will specialize on different parts of the data and the responsibilities encode the learned cluster assignments.

We will apply a common variant of the EM-algorithm where responsibilities are computed greedily, i.e., where the data is hard assigned to the mixture component with maximal likelihood instead of weighted continuously with the component's likelihood in the M-step. In our case, the likelihood of a data triplet $(\Theta_t, \Theta_{t+1}, \tau_t)$ under the ith inverse model is $P(\tau_t \,|\, \Theta_t, \Theta_{t+1}, c_t = i)$, which is a Gaussian with either fixed variance or the variance given by LWPR's confidence bounds. This approach is similar to MOSAIC's approach to data separation except that it is based on the inverse models, accounts for the possibility of non-linear models, and allows us to use the correct confidence bounds predicted by LWPR.

4 Experiments

The methods proposed earlier were tested on a simulated[1] 3 joint arm, with 3 degrees of freedom (see Fig.3(c)). The first joint allows up and down movements and the next two allow left and right movements. The target trajectories for the arm were a superposition of different phase-shifted sinusoidal trajectories for each joint:

$$\theta_i^* = a_i \cos(\alpha_i \, \frac{2\pi}{T} \, t) + b_i \cos(\beta_i \, \frac{2\pi}{T} \, t) \;, \tag{8}$$

[1] Robot arm simulation modeled in dynamical physics engine ODE/OpenGL

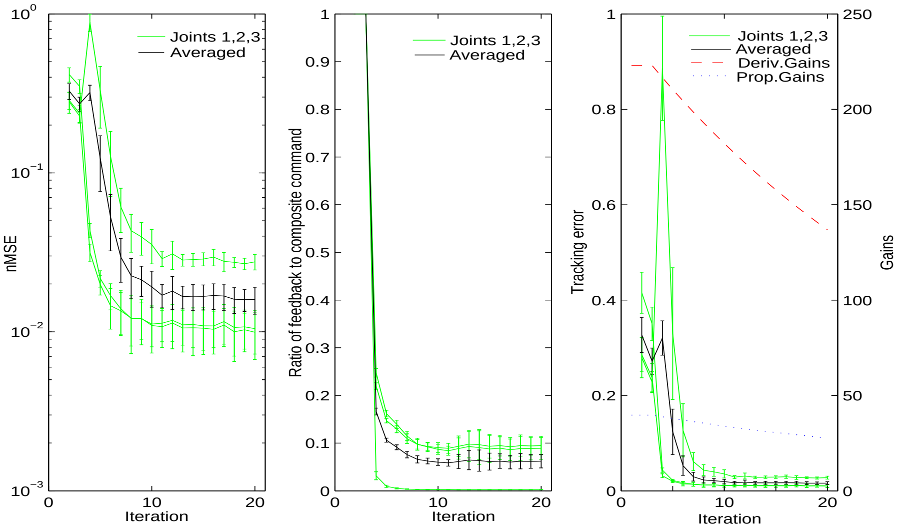

Fig. 4. Control performance of online trained LWPR on a single context over the training cycles. Left: normalized MSE on the test data. Middle: contribution of the error-correcting feedback PD control. Right: tracking error under decreasing PD gains.

where $T = 4000$ is the total length of the target trajectory, $a_i, b_i \in [-1, 1]$ are different amplitudes and $\alpha_i, \beta_i \in \{1, .., 15\}$ parameterize different frequencies. Different contexts are simulated by changing the weight of the third body of the arm. This is equivalent to varying work loads held by the arm.

4.1 Learning single context dynamics and using them for control

We will first demonstrate that LWPR can learn an accurate inverse model of the arm dynamics online and use it for control. Training was repeated independently for six different contexts. Twenty iterations of the trajectory were executed. In the first 3 iterations, a pure PD controller is used, whereas, after that a composite controller with the model being learnt is used. Every second sample of the dynamics experienced is used for training the inverse model online and every other sample is kept to test the accuracy of the inverse model. Fig. 4 (left) shows how the normalised mean square error (nMSE) on the test data drops as training proceeds through the 20 iterations, indeed, converging to very low nMSE for all joints.

The accuracy of the inverse model learned can also be judged based on the contribution of the feedback command to the total composite command. The smaller the contribution of the feedback command, the more accurate the inverse model learnt is. The average contribution of the feedback command through the 20 iterations can be seen in Fig. 4(middle). Already from the fourth iteration, when we switch from PD control to composite control, the contribution is quite low and drops further – in accordance to the behaviour of the nMSE. In Fig. 4(right), we can also see how the tracking error decreases as the model becomes more accurate while, at the same time, we decrease the gains of the feedback controller.

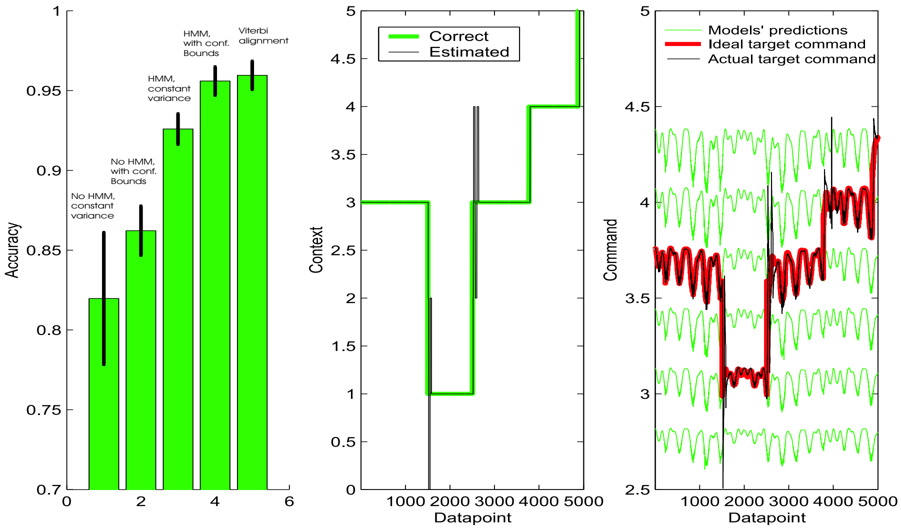

Fig. 5. Online context estimation and control in the case of six different contexts. Left: Context estimation accuracy using different estimation methods. Middle: example of random context switches and its estimate using HMM filtering over time. Right: The inverse model predictions of the six contexts along with the ideal and actual generated target command

4.2 Experiments with context estimation

The context estimation methods described in section 3.1 were used for online estimation and control with the six contexts learnt. Random switches between the six contexts were performed in the simulation, where at every time step we switch to a random context with probability .001 and stay in the current context otherwise. The context estimates were used online for selecting the model that will provide the feed-forward commands.

We have two classes of experiments, one is where we are not using HMM filtering of the contextual variable and the other is where we use it. Also, we have two choices for the variance of the observation model, one is where we use a constant (found empirically) and the other is where we use the more principled confidence bounds provided by LWPR. The simulation was run for 10 iterations. The percentage of accurate online context estimates for the four cases along with offline Viterbi alignment can be seen in the Fig. 5(left).

Fig. 5(middle) gives an example of how the best context estimation method that we have, the HMM filtering using LWPR's confidence bounds, performs when used for online context estimation and control. Sometimes the context estimation lags behind a few time steps when there are context switches, which is a natural effect of online filtering (as opposed to retrospect smoothing).

The performance of online context estimation and control is close to the control performance we achieved for the single context displayed in Fig. 4. Using the HMM filtering based on LWPR's confidence bounds, the average tracking error over the 10 cycles was 0.0019 and the ratio of feedback PD control was 0.074.

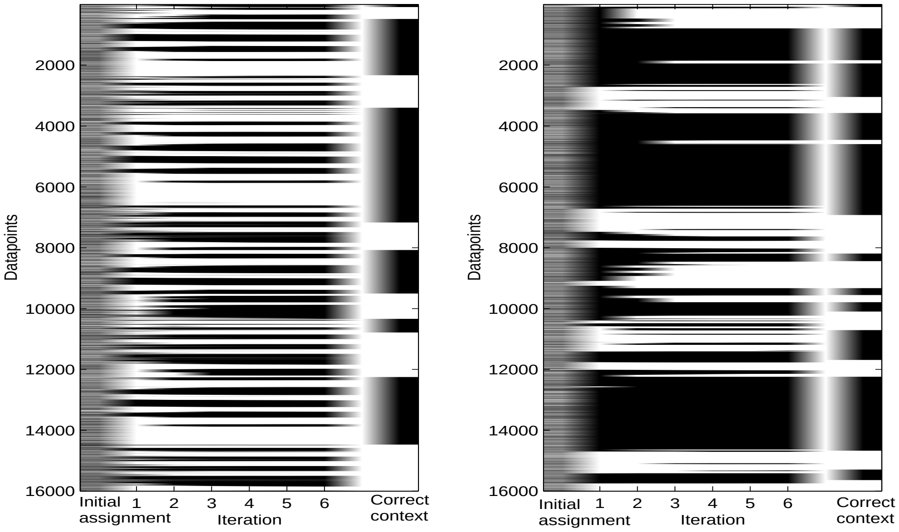

Fig. 6. The evolution of the data separation from unlabeled data over six iterations of the EM-procedure. Left: without exploiting temporal modelling of the context variable. Right: using a Viterbi alignment according to the temporal modelling. Both methods use LWPR's confidence bounds as local variance estimate. The first column displays the initial random assignment of datapoints to contexts. The last column displays the correct context for each datapoint.

4.3 Experiments with data separation

Finally, we investigate the bootstrapping of data separation from unlabeled data. Here, when generating the data, we switched between two different contexts (work loads) with probability .001 at each time step. We first collected a batch of context-unlabeled data from 4 cycles through the target trajectory where the arm was controlled by pure feedback PD control. The EM procedure for data separation (section 3.2) was tested on this data with and without temporal modelling (always using LWPR's confidence bounds as a basis). In the temporal case, Viterbi alignment was used to assign datapoints to contexts rather than filtered estimates. Fig. 6 compares the evolution of the data separation for the two methods over six EM-iterations. Using the temporal context performs much better, i.e., 84% of the datapoints were assigned to the correct context.

The bootstrapping of the context separation from unlabeled data gives rise to two separate inverse models for the two different contexts. To further improve these models, we then used them for online context estimation and control, just as investigated in the previous section, for another 12 cycles through the target trajectory. Simultaneously, the context estimates were used for selecting data for further training of the models. The accuracy of context estimation was 88% while the tracking error was 0.0051 and the ratio of feedback PD control was 0.23. The errors are slightly higher than in the case where models were trained using labeled data, but this is satisfying considering the fact that we started with unlabeled data.

5 Discussion

In this paper we presented an efficient multiple model paradigm for the general case of non-linear control. The approach is based on a probabilistic model of multiple-context dynamics, using LWPR as an efficient online regressor for each inverse model. We have demonstrated that it is possible to bootstrap multiple models of non-linear dynamics from context-unlabeled data and use them for simultaneous online context estimation, control, and training.

In comparison to previous multiple model approaches, most notably MOSAIC, our approach is the first to handle the case of non-linear dynamics. Further, we showed that it is unnecessary to maintain pairs of forward and inverse models. Context estimation can more efficiently be based solely on the learned inverse models for each context. We have seen that including a Markovian model of context switching greatly enhances the context estimation performance. If additional knowledge about the context is available, for instance, if it is related to sensory information, one can easily extend our framework by augmenting the likelihood term in the Markovian model.

An issue yet unaddressed by any existing method is that of determining the number of separate contexts based on data only, if it is not known a priori. As detailed in section 3.2, our formulation of data separation is very similar to that of clustering using mixture of Gaussians. Hence, existing techniques for determining the necessary number of clusters in mixtures of Gaussians literature can directly be exploited. More specifically, a common approach is to incrementally add new mixture components when the new data cannot, with sufficient likelihood, be explained with existing mixture components [7]. This can also be realized online, which will be the subject of future research to extend the presented approach.

References

1. Imamizu H. Osu R. Yoshioka T. Flanagan R., Nakano E. and Kawato M. Composition and decomposition of internal models in motor learning under altered kinematic and dynamic environments. *The Journal of Neuroscience*, 19, 1999.
2. Wolpert D. M. Haruno M. and Kawato M. Mosaic model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220, 2001.
3. Wolpert D. M. and Kawato M. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11:1317–1329, 1998.
4. K. S. Narendra and J. Balakrishnan. Adaptive control using multiple models. *IEEE Transactions in automatic control*, 42:171–187, 1997.
5. K. S. Narendra and C. Xiang. Adaptive control of discrete-time systems using multiple models. *IEEE Transactions in automatic control*, 45:1669–1686, 2000.
6. Aaron D'Souza Sethu Vijayakumar and Stefan Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17:2602–2634, 2005.
7. J. J. Verbeek, N. Vlassis, and B. J. A. Kröse. Efficient greedy learning of gaussian mixture models. *Neural Computation*, 15(2):469–485, 2003.