Hazara, Murtaza; Kyrki, Ville

Model selection for incremental learning of generalizable movement primitives

Published in:
Proceedings of the 2017 18th International Conference on Advanced Robotics, ICAR 2017

DOI:
10.1109/ICAR.2017.8023633

Published: 30/08/2017

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Hazara, M., & Kyrki, V. (2017). Model selection for incremental learning of generalizable movement primitives. In *Proceedings of the 2017 18th International Conference on Advanced Robotics, ICAR 2017* (pp. 359-366). Article 8023633 IEEE. https://doi.org/10.1109/ICAR.2017.8023633

Model Selection for Incremental Learning of Generalizable Movement Primitives

Murtaza Hazara and Ville Kyrki

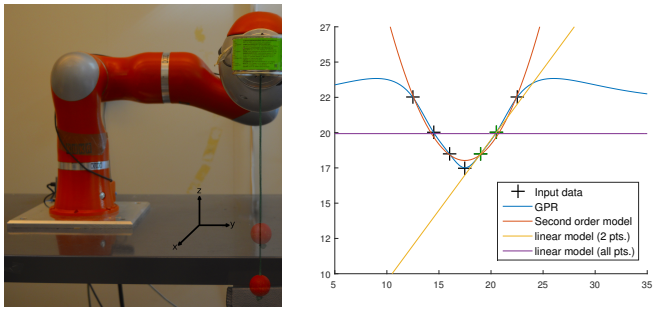
***Abstract*— Although motor primitives (MPs) have been studied extensively, much less attention has been devoted to studying their generalization to new situations. To cope with varying conditions, a MP's policy encoding must support generalization over task parameters to avoid learning separate primitives for each condition. Local and linear parameterized models have been proposed to interpolate over task parameters to provide limited generalization.**

In this paper, we present a global parametric motion primitive which allows generalization beyond local or linear models. Primitives are modelled using a linear basis function model with global non-linear basis functions. Using the global parametric model, we developed an online incremental learning framework for constructing a database of MPs from a single human demonstration. Above all, we propose a model selection method that can choose an optimal model complexity even with few training samples, which makes it suitable for online incremental learning. Experiments with a ball-in-a-cup task with varying string lengths demonstrate that the global parametric approach can successfully extract underlying regularities in a database of MPs leading to enhanced generalization capability of the parametric MPs and increased speed (convergence rate) of learning. Furthermore, it significantly excels over locally weighted regression both in terms of inter- and extrapolation.

I. INTRODUCTION

Learning a skill in a perturbed environment often requires practising it under various conditions. For example, to learn to score in basketball, an individual needs to practice throwing from different locations. Subsequently, generalizing to a new situation (e.g. location) becomes easier as the individual learns incrementally the underlying regularities of the task. Incremental learning has been studied [1] in the context of iterative learning control (ILC) where a desired trajectory is adapted to a known reference trajectory in an online incrementally manner. However, in this paper, we propose incremental learning in the context of reinforcement learning (RL) where such a reference trajectory is unknown. In fact, finding the reference trajectory for the new perturbed environment is our objective.

In this paper, we propose a new global parametric learning from demonstration (LfD) approach for generalizing an imitated task to new unseen situations. We selected the ball-in-a-cup task to assess how effective our method is in generalizing from initial demonstration with certain string length to changed lengths. We demonstrate that our approach is capable of interpolation (new string length in the range of training set) and extrapolation (string length outside the range of training set). The kinematics of the initial demonstration are encoded using a Dynamic Movement Primitive (DMP). Afterwards, the shape parameters of the DMP are optimized using PoWER [2] for a new task parameter (string length) and added to a database (DB) of MPs. The global model is updated in an online fashion and its complexity is controlled as new MPs are added to the database. The global model can provide RL with a good initial policy boosting up the learning process, and in return RL provides the global model with more training data leading to an enhanced prediction of policies for new unseen task parameters.



(a) (b)

Fig. 1: (a) Ball-in-a-cup game with two different string lengths. (b) Models fit to 7 points of a non-linear function. Only the second order model captures the global pattern, while the GPR model tends to the mean when extrapolating, and the linear model is either (when trained with all samples) leaning toward the mean, or finds only the local pattern when fit to only two points marked by the green color.

The main contribution of this paper is providing an incremental learning framework in the context of RL for constructing a DB of MPs. The underlying regularities of this DB are extracted online using a novel global parametric model generalizing task parameters to policy parameters. Above all, we propose a novel penalized log-likelihood based model selection for controlling the complexity of the global model as new MPs are added to the DB. We also show that this model selection works even with few training samples indicating its superiority for online incremental learning over traditional model selection methods such as Akaike information criterion (AIC), Bayesian information

M. Hazara and V. Kyrki are with School of Electrical Engineering, Aalto University, Finland murtaza.hazara@aalto.fi, ville.kyrki@aalto.fi

This work was supported by Academy of Finland, decisions 264239 and 268580.

criterion (BIC), and leave-one-out cross validation. Our experiments demonstrate also that our global parametric approach generalizes significantly better than LWR in terms of both inter- and extrapolation.

II. Related Work

To adapt LfD models to new environments, the model parameters need to be adjusted according to parameters characterizing the new environment or task. Existing generalizable LfD models can be categorized as (i) generalization by design where the parameters are explicit in the model structure such as the goal of a DMP; (ii) generalization based on interpolation which uses a weighted combination of training models; and (iii) generalization by global linear models where the model parameters depend linearly on the environment/task parameters.

Kober et al. [3] utilized a cost regularized kernel regression (based on Gaussian process regression) for learning the mapping of new situations to meta-parameters including the initial position, goal, amplitude and the duration of the MPs; they model the automatic adjustment of the meta-parameters as a RL problem. The approach allows then adjusting these designed aspects of motion based on the task but does not enable modifying other characteristics of the trajectory (policy), making the approach suitable for learning tasks where the DMPs are adapted spatially and temporally without changing the overall shape of the motion, but unsuitable for tasks where the dynamics during motion are changed such as in this paper.

Researchers have recently shown interest also in generalizing DMP shape parameters to new situations [4], [5], [6], [7], [8], [9]. The approaches are primarily based on local regression methods. For example, Da Silva et al. [4] extract lower dimensional manifolds (latent space) from learned policies using ISOMAP algorithm; they achieve a generalizable policy by mapping the manifolds (representing task parameters) to DMPs shape parameters. Support Vector Machines with local Gaussian kernels are used for learning the mapping. Similarly, Forte et al. [5] utilize Gaussian process regression (GPR) to learn the mapping of task parameters to DMP shape parameters. Ude et al. [6] and Nemec et al. [8] utilize Locally Weighted Regression (LWR) and kernels with positive weights for learning the mapping. Stulp et al. [7] learn the original shape parameters and generalize it with one single regression using Gaussian kernels. Mülling et al. [9] propose a linear mixture of MPs for generalization and refine the generalized behavior using RL.

Although the local regression approaches might interpolate within the range of demonstrations, their extrapolation capability is not guaranteed, which is mainly because of the kernels learning the local structure; thus they typically tend towards the mean of training data when extrapolating. This problem is illustrated using a simple example in Fig. 1b which shows local (GPR) and global (linear, second order) models fit to a set of points. When a line is fit to two points close to each other, it learns the local pattern allowing interpolation and extrapolation in a small neighborhood. If the line fit is made using all points, the fit tends to become poor everywhere. A local regression such as GPR performs well in interpolation, but not in extrapolation. When a well fitting higher order global model can be found, it typically outperforms the others in extrapolation.

Carrera et al. [10] developed a parametric MPs model based on a mixture of several DMPs. First, they record multiple demonstrations. A DMP model is fit to each demonstration, and a parametric value is assigned to it representing the task environment in which the demonstration was recorded. Then, they calculate the influence of each model using a distance function between the model parameters and the parametric value describing the current environment perturbation. Using the influence of each model as its mixing coefficient, the mixture of models is computed at the acceleration level. Since it is a linear combination with positive coefficients, the mixture model is not expected to be capable of extrapolation. Calinon et al. [11] proposed an MPs model based on a Gaussian mixture model and generalize it to new situations using expectation maximization. Although their model is capable of linear extrapolation, it is only applicable when the task parameters can be represented in the form of coordinate systems. All things considered, few researchers [12] have considered the extrapolation capability in generalizing task parameters to model parameters, which is the main focus of this paper.

III. Method

In this section, we review dynamic movement primitives (DMPs). After that, we clarify our global parametric dynamic movement primitives (GPDMPs) method which incorporates both linear and non-linear parametric models. Besides that, we review traditional model selection approaches and then explain our penalized log-likelihood based model selection method.

A. Dynamic Movement Primitives

DMPs encode a policy for a one-dimensional system using two differential equations. The first differential equation

$$\dot{z} = -\tau \alpha_z z \tag{1}$$

formulates a canonical system where z denotes the phase of a movement; $\tau = \frac{1}{T}$ represents the time constant where T is the duration of a demonstrated motion, and α_z is a constant controlling the speed of the canonical system. This first order system resembles an adjustable clock driving the transform system

$$\frac{1}{\tau}\ddot{x} = \alpha_x(\beta_x(g-x) - \dot{x}) + f(z;\mathbf{w}) \tag{2}$$

consisting of a simple linear dynamical system acting like a spring damper perturbed by a non-linear component (forcing function) $f(z;\mathbf{w})$. x denotes the state of the system, and g represents the goal. The linear system is critically

damped by setting the gains as $\alpha_x = \frac{1}{4}\beta_x$. The forcing function

$$f(z;\mathbf{w}) = \mathbf{w}^T\mathbf{g} \tag{3}$$

controls the trajectory of the system using a time-parameterized kernel vector $\mathbf{g}$ and a modifiable policy parameter vector (shape parameters) $\mathbf{w}$. Each element of the kernel vector

$$[\mathbf{g}]_n = \frac{\psi^n(z)z}{\sum_{i=1}^{N}\psi^i(z)}(g - x_0) \tag{4}$$

is determined by a normalized basis function $\psi^n(z)$ multiplied by the phase variable z and the scaling factor $(g - x_0)$ allowing for the spatial scaling of the resulting trajectory. Normally, a radial basis function (RBF) kernel

$$\psi^n(z) = \exp(-h_n(z - c_n)^2) \tag{5}$$

is selected as the basis function. The centres of kernels (c_n) are usually equispaced in time spanning the whole demonstrated trajectory. It is also a common practice to choose the same temporal width $(h_n = \frac{2}{3}|c_n - c_{n-1}|)$ for all kernels. Furthermore, the contribution of the non-linear component (3) decays exponentially by including the phase variable z in the kernels. Hence, the transform system (2) converges to the goal g.

The shape parameter vector $\mathbf{w}$ can be learned using weighted linear regression [13]; firstly, the nominal forcing function f^{ref} is retrieved by integrating the transform system (2) with respect a human demonstration x^{demo}; next, the shape parameter for every kernel is estimated using

$$[\mathbf{w}]_n = (\mathbf{Z}^T\Psi\mathbf{Z})^{-1}\mathbf{Z}^T\Psi\mathbf{f}^{ref} \tag{6}$$

where $[\mathbf{f}^{ref}]_t = f_t^{ref}$, $[\mathbf{Z}]_t = z_t$, and $\Psi = \text{diag}(\psi_1^n, ..., \psi_t^n, ..., \psi_T^n)$.

B. Global Parametric Dynamic Movement Primitives

Using DMPs, a task can be imitated from a human demonstration; however, the reproduced task cannot be adapted to different environment conditions. To overcome this limitation, we have integrated a parametric model to DMPs capturing the variability of a task from multiple demonstrations. We transform the basic forcing function (3) into a parametric forcing function

$$f(z, \mathbf{l}; \mathbf{w}) = \mathbf{w}(\mathbf{l})^T\mathbf{g} \tag{7}$$

where the kernel weight vector $\mathbf{w}$ is parametrized using a parameter vector $\mathbf{l}$ of measurable environment factors.

We model the dependency of the weights with respect to parameters as a linear combination of J basis vectors $\mathbf{v}_i$ with coefficients depending on parameters in a non-linear fashion,

$$\mathbf{w}(\mathbf{l}) = \sum_{i=0}^{J}\phi_i(\mathbf{l})\mathbf{v}_i = \mathbf{V}^T\phi(\mathbf{l}) \tag{8}$$

where $\mathbf{V}$ s a $J \times N$ matrix of parameters with N referring to the number of kernels $\mathbf{g}$. $\phi(\mathbf{l})$ is a J dimensional column vector with elements $\phi_j(\mathbf{l})$. The non-linear basis $\phi_j(\mathbf{l})$ for a polynomial model in one parameter is $\phi_j(l) = l^j$.

The formulation captures linear models such as [12] as a special case. Considering a single parameter l for presentational simplicity, the linear model can be written

$$\mathbf{w}(l) = l\mathbf{v}_1 + \mathbf{v}_0. \tag{9}$$

For a chosen non-linear basis (known functions ϕ_i), the basis vectors can be calculated by minimizing the difference between modelled and initial non-parametric DMP shape parameters,

$$\underset{\mathbf{V}}{\text{argmin}}\sum_{k=1}^{K}\|\mathbf{w}(\mathbf{l}_k) - \mathbf{w}_k\|_2 \tag{10}$$

where $\mathbf{w}_k$ denotes the initial weight vector of a non-parametric DMP optimized for parameter values $\mathbf{l}_k$. The initial weights can be merely imitated from a human demonstration using (6) or improved using a policy search method [14]. In either case, reproducing an imitated task using $\mathbf{w}_k$ should lead to a successful performance in an environment parametrized by $\mathbf{l}_k$.

In order to solve (10), one needs to construct the design matrix

$$\Phi = \begin{bmatrix} \phi_1(\mathbf{l}_1) & \phi_2(\mathbf{l}_1) & \dots & \phi_J(\mathbf{l}_1) \\ \phi_1(\mathbf{l}_2) & \phi_2(\mathbf{l}_2) & \dots & \phi_J(\mathbf{l}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{l}_K) & \phi_2(\mathbf{l}_K) & \dots & \phi_J(\mathbf{l}_K) \end{bmatrix} \tag{11}$$

where K denotes the number of initial DMPs weight vectors which must be at least equal to or greater than the order of the model J to avoid unconstrained optimization problems. Furthermore, the rows of the target matrix

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_K^T \end{bmatrix} \tag{12}$$

represent an initial DMPs weight vector. We can minimize (10) with respect to the matrix of basis vectors $\mathbf{V}$, giving

$$\hat{\mathbf{V}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{W}. \tag{13}$$

C. Model selection

The order of complexity for a parametric regression model needs to be selected for overcoming the so-called over-fitting problem. The model complexity can be determined for every single policy parameter (e.g. weight of every single DMP kernel). However, in such a univariate model, the correlation among policy parameters are ignored, thus resulting in a very non-smooth trajectories for new task parameters. Hence, we consider a single model complexity for the global model.

The model choice for a particular application is a compromise between complexity of the modelled attractor landscape and overfitting due to insufficient data. The best generalization can be achieved by choosing an optimal order of complexity for the model, which is addressed in a model selection method such as cross validation, AIC, or BIC.

Leave-one-out cross validation is a traditional model selection technique which separates training data into a training set and a validation set for assessing the generalization performance. On the other hand model selection methods based on information criteria such as AIC and BIC exploit the whole training set for both training the model and choosing its complexity. Both of these methods are based on penalized log-likelihood, where the penalty term constrain the model complexity. AIC selects the model which maximizes

$$A_I = \log p(\mathbf{W}|\hat{\Theta}) - J \tag{14}$$

where $p(\mathbf{W}|\hat{\Theta})$ denotes the likelihood of training data; however, AIC favours higher order of complexity resulting in overfitting.

BIC put a stronger penalty on the number of free parameters (in this case J), and thereby controlling the complexity of the model. The optimal model order can be selected by minimizing the BIC-cost

$$B = -2\log p(\mathbf{W}|\hat{\Theta}) + J\log K. \tag{15}$$

We assume a linear regression model with additive white Gaussian noise

$$\mathbf{W} = \Phi\mathbf{V} + \mathscr{E} \tag{16}$$

where $\mathscr{E}$ is the error matrix

$$\mathscr{E} = \begin{bmatrix} \epsilon_1^T \\ \vdots \\ \epsilon_K^T \end{bmatrix} \tag{17}$$

with each row

$$\epsilon_i = \mathbf{w}_i - \hat{\mathbf{V}}^T\phi(\mathbf{l}_i) \tag{18}$$

representing the difference between the ith training sample $\mathbf{w}_i$ and its prediction $\hat{\mathbf{V}}^T\phi(\mathbf{l}_i)$. Hence, the likelihood of data is

$$\begin{aligned} \log p(\mathbf{W}|\hat{\Theta}) &= \log \prod_{i=1}^{K} \mathscr{N}(\mathbf{w}_i|\mathbf{l}_i, \hat{\mathbf{V}}, \hat{\Sigma}) \\ &= -\frac{KN}{2} log(2\pi) - \frac{K}{2} log(\det(\hat{\Sigma})) - \\ &\quad \frac{1}{2} tr\{(\mathbf{W} - \Phi\hat{\mathbf{V}})^T(\mathbf{W} - \Phi\hat{\mathbf{V}})\hat{\Sigma}^{-1}\} \end{aligned} \tag{19}$$

where N is the number of DMPs kernels (size of $\mathbf{g}$ in 4); $\hat{\mathbf{V}}$ denotes the maximum likelihood estimate (MLE) of parameter matrix (using (13)); and, $\hat{\Sigma}$ represent MLE estimate of the covariance

$$\hat{\Sigma} = \frac{1}{K}(\mathbf{W} - \Phi\hat{\mathbf{V}})^T(\mathbf{W} - \Phi\hat{\mathbf{V}}) \tag{20}$$

of DMPs weight vector $\mathbf{w}$. After eliminating the constant term ($-\frac{KN}{2} log(2\pi)$) in (19), one can rewrite (15) into

$$B = K\log(\det(\hat{\Sigma})) + tr((\mathbf{W} - \Phi\hat{\mathbf{V}})^T(\mathbf{W} - \Phi\hat{\mathbf{V}})\hat{\Sigma}^{-1}) + J\log K. \tag{21}$$

The determinant of the MLE estimate of the covariance, $\hat{\Sigma}$ will be zero with few training samples causing the BIC cost to go toward negative infinity. Hence, we have modified the traditional definition of the BIC cost

$$B_M = -2\log p(\mathscr{E}|\hat{\Theta}) + J\log K \tag{22}$$

by considering the distribution of the noise which can be written into

$$\begin{aligned} \log p(\mathscr{E}|\hat{\Theta}) &= \log \prod_{i=1}^{K} \mathscr{N}(\epsilon_i|0,\Sigma) = -\frac{KN}{2}\log(2\pi) - \frac{K}{2}\log(\det(\Sigma)) \\ &\quad - \frac{1}{2}\sum_{i=1}^{K}(\epsilon_i^T\Sigma\epsilon_i) \\ &= -\frac{KN}{2}\log(2\pi) - \frac{K}{2}\log(\det(\Sigma)) \\ &\quad - \frac{1}{2}\sum_{i=1}^{K}(\mathbf{w}_i - \hat{\mathbf{V}}^T\phi(\mathbf{l}_i))^T\Sigma(\mathbf{w}_i - \hat{\mathbf{V}}^T\phi(\mathbf{l}_i)) \\ &= -\frac{KN}{2}\log(2\pi) - \frac{K}{2}\log(\det(\Sigma)) \\ &\quad - \frac{1}{2} tr((\mathbf{W} - \Phi\hat{\mathbf{V}})^T(\mathbf{W} - \Phi\hat{\mathbf{V}})\Sigma^{-1}) \end{aligned} \tag{23}$$

due to the the i.i.d assumption on the white additive noise ($\epsilon_i \sim \mathscr{N}(0,\Sigma)$), where Σ represents a constant covariance matrix which needs to be determined prior to the model selection process. In our experiments, we have selected an scaled identity matrix $s\mathbf{I}$ as the constant covariance matrix where s denotes the scale. The scale can be determined with respect to the magnitude of the error (difference between $\mathbf{w}_k$ and $\hat{\mathbf{V}}^T\phi(\mathbf{l}_k)$). A simple way to estimate the scale is to look at the largest eigenvalue of the MLE estimate of the covariance matrix (20) with linear fitting. After eliminating constant terms ($-\frac{KN}{2}\log(2\pi)$ and $\frac{K}{2}\log(\det(\Sigma))$) in (23), one can rewrite the modified BIC cost (22) into

$$B_M = tr((\mathbf{W} - \Phi\hat{\mathbf{V}})^T(\mathbf{W} - \Phi\hat{\mathbf{V}})\Sigma^{-1}) + J\log K. \tag{24}$$

The model which minimizes B_M will be selected. The first term in B_M favours higher order model while the second term discourages a very high order; hence, it guarantees best prediction while avoiding over-fitting. Since the determinant of the MLE covariance does not appear in our modified BIC B_M (24), it is more suitable than the traditional BIC when the training samples are scarce.

D. Reinforcement Learning

Executing a DMP with imitated shape parameters might not lead to a successful reproduction of a task. One way to refine the shape parameters is to learn them through trial-and-error using policy search reinforcement learning (RL). Next, we briefly review the state-of-the-art policy search method PoWER [2] which was used in this work to optimize individual primitives.

PoWER (see Algorithm 1) updates the DMP shape parameters $\boldsymbol{\theta} \equiv \mathbf{w}$ iteratively. In each iteration, (several) stochastic roll-out(s) of the task is performed, each of which is achieved by adding random (Gaussian) noise to the DMPs shape parameters. Each noisy vector is weighted by the returned accumulated reward. Hence, the higher

the returned reward, the more the noisy vector contribute to the updated policy parameters. This exploration process continues until the algorithm converges to the optimal policy.

Algorithm 1 PoWER [2].

Input: The initial policy parameters $\boldsymbol{\theta}$, the exploration variance $\boldsymbol{\Sigma}$

1: **repeat**

2: *Sample*: Perform rollout(s) using $\mathbf{a} = (\theta + \epsilon_{\mathbf{t}})^{\mathbf{T}} \phi(\mathbf{s}, \mathbf{t})$ with $\epsilon_{\mathbf{t}}^{\mathbf{T}} \phi(\mathbf{s}, \mathbf{t}) \sim \mathcal{N}(\mathbf{0}, \phi(\mathbf{s}, \mathbf{t})^{\mathbf{T}} \hat{\Sigma} \phi(\mathbf{s}, \mathbf{t}))$ as stochastic policy and collect all $(\mathbf{t}, \mathbf{s}_{\mathbf{t}}^{\mathbf{h}}, \mathbf{a}_{\mathbf{t}}^{\mathbf{h}}, \mathbf{s}_{\mathbf{t+1}}^{\mathbf{h}}, \epsilon_{\mathbf{t}}^{\mathbf{h}}, \mathbf{r}_{\mathbf{t+1}}^{\mathbf{h}})$ for $\mathbf{t} = \{\mathbf{1}, \mathbf{2}, \ldots, \mathbf{T+1}\}$.

3: *Estimate*: Use unbiased estimate

$$\hat{Q}^{\pi}(s, a, t) = \sum_{\widetilde{t}=t}^{T} r(s_{\widetilde{t}}, a_{\widetilde{t}}, s_{\widetilde{t}+1}, \widetilde{t}).$$

4: *Reweight*: Compute importance weights and reweigh rollouts, discard low-importance rollouts.

5: *Update* policy using

$$\theta_{k+1} = \theta_k + \frac{\langle \sum_{t=1}^{T} \epsilon_t Q^{\pi}(\mathbf{s}, \mathbf{a}, t) \rangle_{w(\tau)}}{\langle \sum_{t=1}^{T} Q^{\pi}(\mathbf{s}, \mathbf{a}, t) \rangle_{w(\tau)}}$$

6: **until** convergance $\theta_{k+1} \approx \theta_k$

The structure of the noise is a key element influencing the convergence speed of a policy search method but the choice is a trade-off. In the case of DMP shape parameters and uncorrelated noise, high noise variance causes large accelerations of the system, causing a safety hazard and possibly surpassing the physical capabilities of the robot. In contrast, low noise variance makes the learning process slow.

To address this trade-off, we propose to use correlated noise instead of the earlier works employing uncorrelated noise. Since the elements of a DMP parameter vector correspond to temporally ordered perturbing forces, we want to control their temporal statistics. To achieve this, an intuitive structure for the covariance matrix $\Sigma = \mathbf{R}^{-1}$ can be used where the quadratic control cost matrix

$$\mathbf{R} = \sum_{j=1}^{F} h_j \mathbf{A}_j^T \mathbf{A}_j \tag{25}$$

is a weighted combination of quadratic costs related to finite difference matrices $\mathbf{A}_1 \cdots \mathbf{A}_F$. h_j denotes the weight of the j-th finite difference matrix, where j is the order of differentiation. This structure allows us then to control statistics of any order. In experiments, we consider variation only in acceleration (second order). Thus, $h_2 = 1$ and all other weights $h_j = 0, j \neq 2$, and the second order finite difference matrix $\mathbf{A}_2$ can be written

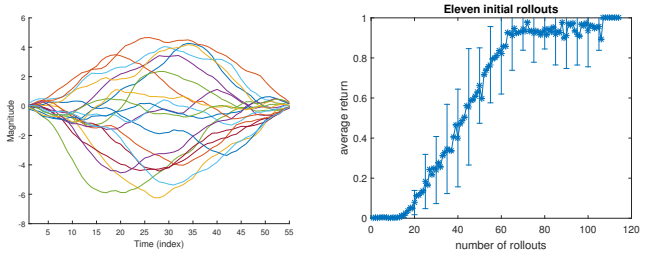
$$\mathbf{A}_2 = \begin{bmatrix} 1 & 0 & 0 & & 0 & 0 & 0 \\ -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -2 & 1 & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & 1 & -2 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & -2 \\ 0 & 0 & 0 & & 0 & 0 & 1 \end{bmatrix}. \tag{26}$$

With this covariance matrix, the noise signal is smooth (see Fig. 2a) due to limited acceleration and it has small magnitude in the beginning and at the end of the trajectory. Hence, safe exploration is provided. It is worth mentioning that a similar covariance matrix has been applied in [15] and [16] for direct trajectory encoding.

In order to control the magnitude of noise, we used a further modified covariance matrix $\Sigma = \gamma \beta \mathbf{R}^{-1}$ where γ is a constant controlling the initial magnitude and convergence factor

$$\beta = \frac{1}{\sum_{i=1}^{I} r_i^2}, \tag{27}$$

reduces the magnitude of noise as the policy search algorithm is converging to the optimal policy.



(a) (b)

Fig. 2: (a) Noises ϵ sampled from a zero mean multivariate Gaussian distribution $\mathcal{N}(0, \Sigma)$ with $\gamma = 1$ and $\beta = 0.01$. (b) Mean and variance of returns over 12 trials where 11 initial roll-outs were used before re-weighting DMP weights.

IV. Experimental Evaluation

We studied experimentally the generalization performance of the proposed model using a Ball-in-a-Cup task taught to KUKA LBR 4+ initially using kinesthetic teaching. In this section, we explain the incremental learning scenario and compare it with LWR and non-incremental learning in terms of convergence speed and the extrapolation capability of the model.

A. Ball-in-a-Cup Task

The Ball-in-a-Cup game consists of a cup, a string, and a ball; the ball is attached to the cup by the string (see Fig. 1a). The objective of the game is to get the ball in the cup by moving the cup in a suitable fashion. In practice, the cup needs to be moved back and forth at first; then, a movement is induced on the cup, thus pulling the ball up and catching it with the cup.

We chose the Ball-in-a-Cup game because variation in the environment can be generated simply by changing the string length. The string length is observable and easy to evaluate, thus providing a suitable parameter representing the environment variation. Nevertheless, changing the length requires a complex change in the motion to succeed in the game. Hence, the generalization capability of a parametric LfD model can be easily assessed using this game.

The state of the robot is defined in a seven dimensional space $\mathbf{X} = \{x, y, z, q_x, q_y, q_z, q_w\}$, where $\mathbf{X}_p = \{x, y, z\}$ represents the position of the robot end effector (cup), and $\mathbf{X}_q = \{q_x, q_y, q_z, q_w\}$ formulates its orientation using a quaternion. The ball-in-a-cup is essentially a two-dimensional game and thus only motion along two axes, y and z was used. In the demonstration phase, the robot was set compliant along y and z, while it was set stiff rotationally and along x, which were considered as constant states. The plane spanning y and z was orthogonal with respect to the table upon which the robot was mounted (see Fig. 1a).

The trajectories along y and z were encoded using separate DMPs with same number of parameters. We found experimentally that 55 kernels (shape parameters) were required so that the reproduced movement was able to put the ball above the rim of the cup in the execution phase. In total, 110 shape parameters were then learned using weighted linear regression (6). However, using these initial shape parameters, the reproduced movement did not put the ball back into the cup. Hence, the shape parameters were optimized in a trial-and-error fashion using RL as described in Sec. III-D.

Reward function is the most fundamental ingredient of RL. We formulated the reward function similar to [2] as

$$r(t) = \begin{cases} e^{-\alpha d^2}, & \text{if } t = t_c, \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

where t_c denotes the time instant when the ball crosses the rim of the cup with a downward motion; d represents the horizontal distance between the rim of the cup and the centre of the ball; and α is a scaling parameter set to 100 in our experiments. The closer the ball is to the rim of the cup, the higher the reward will be. As the shape parameters are fine-tuned in a trial-and-error approach, the ball would get closer to the cup. Furthermore, the reward is zero if the ball does not reach above the rim of the cup.

B. Incremental Learning

Learning without a prior is a very time-consuming process. The speed of RL will improve significantly when the learning process is started from a good initial policy. In fact, it is customary to initiate the policy search process with a policy that imitates a human demonstration. However, the optimized policy is not guaranteed to work successfully in a new environment characterized by a different task parameter. In this case, a policy must be optimized for the new task parameter. Starting the optimization process from an imitated policy is still time-consuming. The speed of learning process can increase with a more accurate initial policy such as the policy optimized for the closest task parameter. In this way, a database of MPs can be created.

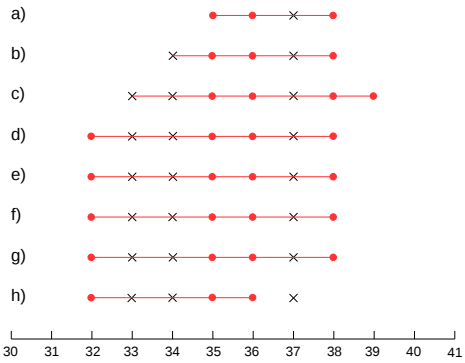
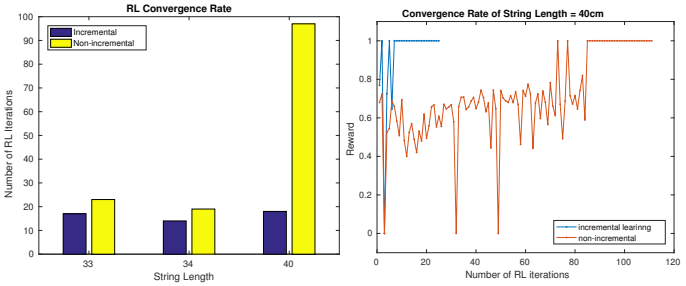


Fig. 3: Validity ranges (red lines) of models learned from database of different sizes. (a) zero order (model selection) global model trained only on one MPs. (b) zero order model trained on 2 MPs. (c) zero-order (model selection) model trained on 3 MPs. (d),(e),(f), and (g) represent first, second, third, and fourth order models trained on the same DB of MPs as in (c). (h) Locally weighted regression.

The underlying regularities in this database can be extracted in an online manner for enhancing the extrapolation capabilities and boosting up the speed of learning process. In fact, we are constructing a global model in an online fashion and control its complexity as new MPs are added to the database. The global model can provide the policy search process with an initial policy which is more accurate than the policy optimized for the closest task parameter. Utilizing only the policy optimized for the closest task parameter is equivalent of using a DB of only one MP and a global model of order zero.

We studied first the generalization performance of global models with different complexities and compared them with LWR. We started the incremental learning process in a Ball-in-a-cup game with a string length of 37 cm. The model selection indicated a zero-order global model. With this model, the game could be re-enacted successfully for string length of 35 to 38 cm. Next, we used the model for providing an initial policy for RL and optimized it for string length of 34 cm. This newly optimized MP was added to the database, and subsequently the global model and its complexity were updated. The model selection again indicated zeroth order for the complexity of the global model. This model works successfully for string length of 34 to 38 cm. An initial policy for string length of 33 cm was estimated using the model learned from the current database, optimized using PoWER, and then added to the database. With this database of three MPs, the model selection still indicated a zero order model. These results are depicted in Fig. 3 where X indicates training samples in the database, and red line the validity region of the model



(a) (b)

Fig. 4: (a) RL convergence rate of incremental vs. non-incremental learning. The number of iterations in (a) include the 11 initial roll-outs. (b) The convergence rate of incremental learning vs. non-incremental for string length of 40cm. The rewards of 11 intial roll-outs are not shown in (b).

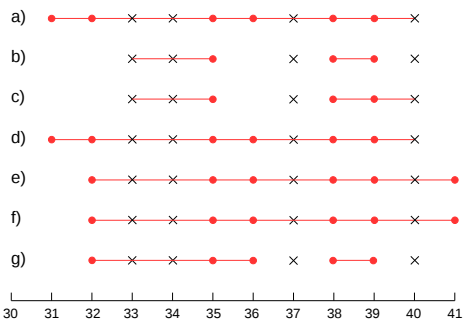


Fig. 5: Validity ranges (red lines) of models learned from database of incrementally learned MPs. (a) model selection indicates 2nd order for Y and zero order for Z. (b), (c), (d), (e), and (f) represent zero, first, second, third, and fourth order models trained on the same DB of MPs as in (a). (g) Locally weighted regression.

where 10 consecutive roll-outs of the same policy were successful. Fitted to the DB of same three MPs, a higher order model such as linear (Fig. 3.d), second (Fig. 3.e), third (Fig. 3.f), and fourth order(Fig. 3.g) could not improve the extrapolation capability. This indicates that the proposed model selection is able to identify the required complexity. Moreover, LWR (see 3.h) could extrapolate to string length of 32 cm because there are more training samples nearby, but it lost both inter- and extrapolation capability for string lengths of 37–38 cm. This indicates superior generalization capabilities of global models in this task.

C. Incremental vs Non-incremental learning

We next studied the effect of incremental learning on the convergence rate of RL for optimizing the policy parameters for a new task parameter. As a starting point, the model trained on MPs for string lengths of 33, 34, and 37 could interpolate among the training samples and extrapolate to length 39 cm. When the MP estimated by this model was used as a starting point of RL for string length of 40 cm, 18 roll-outs (including 11 initial roll-outs) of RL were needed. On the other hand, when only MP of length 37 was used as the initial policy, the policy search took 97 roll-outs (including 11 initial roll-outs) to optimize the MP (compare the red vs. blue in Fig 4.b). Studying other string lengths, incremental learning consistently led to faster convergence when extrapolating (compare the red vs. blue in Fig 4.a), demonstrating that incremental learning speeds up the learning process by providing a more accurate starting point.

D. Model Selection

We finally studied if the proposed model selection criterion can choose an optimal complexity for both incremental and non-incremental learning. The MP optimized for string length 40 cm was added to the database. The global model was constructed for different orders of complexity. The model selection criterion indicated 2 (parabolic) as the order for Y direction and zero for the Z direction. This model (Fig. 5.a) could inter- and extrapolate between 31 and 40 cm. Constant (Fig. 5.b) and linear (Fig. 5.c) models were not sufficient as they could not even interpolate in the whole range. The interpolation and extrapolation range of a 2nd order model (Fig. 5.d) was the same as the model selected by the proposed criterion (Fig. 5.a) but the model selection gives a simpler (zero order) model for Z direction. Furthermore, higher order models (see 5.e and 5.f) improved neither interpolation nor extrapolation capability. Besides, a higher order model can overfit to the training samples leading to very limited extrapolation. For example, a 4th order model fit to the database (see Fig. 5.f) led to bad trajectory (see Fig. 7) for string length of 45. Therefore, it was not safe to exploit a very high order model for extrapolating to a distant task parameter. A similar problem was encountered with LWR, which yielded in a too high acceleration for both Y and Z in the beginning of the trajectory (marked with black star in Fig. 7). Hence, model selection is a key process for constructing an online incremental DB of MPs using parametric models. All in all, the proposed model selection method resulted in the simplest yet most effective and safest model, excelling over LWR (Fig. 5.g).

To study the effectiveness of model selection in non-incremental learning, we constructed models of varying complexity from a DB of MPs learned non-incrementally. The validity ranges of the models are shown in Fig. 6. Linear (Fig. 6.b) and second-order (Fig. 6.c) models had the largest validity region. The model selection criterion chose the linear model, which was the simplest with the best extrapolation capability.

The extrapolation capability dwindled when non-incremental DB was used (compare Fig. 6.b and Fig. 5.a). Moreover, higher order models (6.d and e) lost their interpolation capability since each MP in the non-incremental database was optimized by starting from the policy optimized for the closest task parameter, thus ignoring the underlying regularities in distant task parameter space.

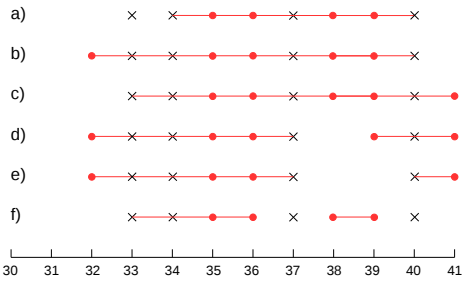
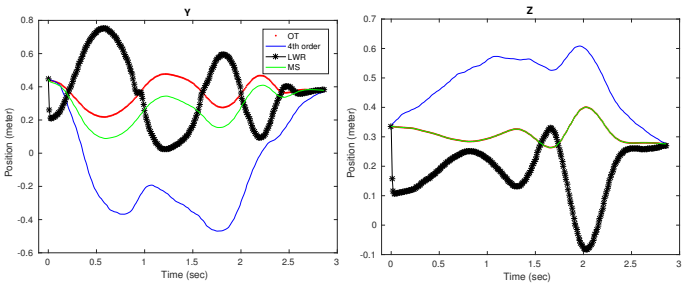


Fig. 6: Validity ranges (red lines) of models learned from databases of non-incrementally learned MPs. (a), (b), (c), (d), (e) represent zero, first, second, third, and fourth order models trained on the same DB of MPs as in (a). Model selection chose (b) as the best model. (f) Locally weighted regression.



(a) (b)

Fig. 7: Extrapolated trajectories for string length of 45cm.(a) trajectories in Y direction. (b) trajectories in Z. Optimal trajectory for string length of 40cm (OT), 4th order global model (4th order), global model with complexity determined by model selection (MS), and LWR trajectories (LWR) are demonstrated. In (b), model selection and optimal trajectory overlaps because model selection indicates zero as the best complexity for Z direction.

Similarly, LWR on the non-incremental DB (Fig. 6.f) performed worse than our approach (Fig. 6.b). These results indicate that model selection and incremental learning are key ingredients for learning generalizable motion primitives.

V. Conclusion

In this paper, we proposed a global model for parametric MPs. The model maps a task parameter to policy parameters, allowing for generalizing a policy to new situations and incremental construction of a database of primitives. The complexity of the model is updated online as a new training example is added to the database. Only a single human demonstration is then needed for constructing the database. Our experiments demonstrated that the global model can provide RL with a more accurate initial policy resulting in a faster convergence; in return, RL provides the global model with additional training examples leading to a better predictive accuracy. Comparing incremental versus non-incremental learning, we showed that incremental learning performs better than non-incremental approach both in terms of generalization and the speed of learning.

All things considered, the global model is simple and can be scaled to accommodate for non-linearities in the task space. Furthermore, we proposed a novel penalized log-likelihood based model selection method which is integral for constructing online incremental DB of MPs using a parametric approach. Experiments showed that the model selection approach lead to a global model which is simple; overcomes over-fitting; and, performs better than locally weighted regression both in terms of inter- and extrapolation. It also works even with few training samples, which indicates its suitability for online incremental learning.

References

[1] A. Gams, M. Denisa, and A. Ude, "Learning of parametric coupling terms for robot-environment interaction," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pp. 304–309, IEEE, 2015.

[2] J. Kober and J. R. Peters, "Policy search for motor primitives in robotics," in *Advances in neural information processing systems*, pp. 849–856, 2009.

[3] J. Kober, E. Oztop, and J. Peters, "Reinforcement learning to adjust robot movements to new situations," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, p. 2650, 2011.

[4] B. Da Silva, G. Konidaris, and A. Barto, "Learning parameterized skills," *arXiv preprint arXiv:1206.6398*, 2012.

[5] D. Forte, A. Gams, J. Morimoto, and A. Ude, "On-line motion synthesis and adaptation using a trajectory database," *Robotics and Autonomous Systems*, vol. 60, no. 10, pp. 1327–1339, 2012.

[6] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 800–815, 2010.

[7] F. Stulp, G. Raiola, A. Hoarau, S. Ivaldi, and O. Sigaud, "Learning compact parameterized skills with a single regression," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 417–422, IEEE, 2013.

[8] B. Nemec, R. Vuga, and A. Ude, "Efficient sensorimotor learning from multiple demonstrations," *Advanced Robotics*, vol. 27, no. 13, pp. 1023–1031, 2013.

[9] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

[10] A. Carrera, N. Palomeras, N. Hurtós, P. Kormushev, and M. Carreras, "Learning multiple strategies to perform a valve turning with underwater currents using an i-auv," in *OCEANS 2015-Genova*, pp. 1–8, IEEE, 2015.

[11] S. Calinon, T. Alizadeh, and D. G. Caldwell, "On improving the extrapolation capability of task-parameterized movement models," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 610–616, IEEE, 2013.

[12] T. Matsubara, S.-H. Hyon, and J. Morimoto, "Learning parametric dynamic movement primitives from multiple demonstrations," *Neural Networks*, vol. 24, no. 5, pp. 493–500, 2011.

[13] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2, pp. 1398–1403, IEEE, 2002.

[14] J. R. Peters, *Machine learning of motor skills for robotics*. PhD thesis, University of Southern California, 2007.

[15] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning force control policies for compliant manipulation," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 4639–4644, IEEE, 2011.

[16] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 4569–4574, IEEE, 2011.