Policy Search with High-Dimensional Context Variables

Voot Tangkaratt
The University of Tokyo,
113-0033 Tokyo, Japan
voot@ms.k.u-tokyo.ac.jp

Herke van Hoof
McGill University,
3480 Rue University, Montreal, Canada

Simone Parisi
Technical University of Darmstadt,
64289 Darmstadt, Germany
simone@robot-learning.de

Gerhard Neumann
Technical University of Darmstadt,
64289 Darmstadt, Germany
geri@robot-learning.de

Jan Peters
MPI for Intelligent Systems,
72076 Tuebingen, Germany
Technical University of Darmstadt,
64289 Darmstadt, Germany
mail@jan-peters.net

Masashi Sugiyama
The University of Tokyo,
277-8561 Chiba, Japan
RIKEN AIP Center,
351-0198 Saitama, Japan
sugi@k.u-tokyo.ac.jp

Abstract

Direct contextual policy search methods learn to improve policy parameters and simultaneously generalize these parameters to different context or task variables. However, learning from high-dimensional context variables, such as camera images, is still a prominent problem in many real-world tasks. A naive application of unsupervised dimensionality reduction methods to the context variables, such as principal component analysis, is insufficient as task-relevant input may be ignored. In this paper, we propose a contextual policy search method in the model-based relative entropy stochastic search framework with integrated dimensionality reduction. We learn a model of the reward that is locally quadratic in both the policy parameters and the context variables. Furthermore, we perform supervised linear dimensionality reduction on the context variables by nuclear norm regularization. The experimental results show that the proposed method outperforms naive dimensionality reduction via principal component analysis and a state-of-the-art contextual policy search method.

Introduction

An autonomous agent often requires different policies for solving tasks with different contexts. For instance, in a ball hitting task the robot has to adapt his controller according to the ball position, i.e., the context. Direct policy search approaches (Baxter and Bartlett 2000; Rosenstein and Barto 2001; Deisenroth, Neumann, and Peters 2013) allow the agent to learn a separate policy for each context through trial and error. However, learning optimal policies for many large contexts, such as in the presence of continuous context variables, is impracticable. On the other hand, direct *contextual* policy search approaches (Kober, Oztop, and Peters 2011; Neumann 2011; da Silva, Konidaris, and Barto 2012) represent the contexts by real-valued vectors and are able to learn a context-dependent distribution over the policy parameters. Such a distribution can generalize across context values and therefore the agent is able to adapt to unseen contexts.

Yet, direct policy search methods (both contextual and plain) usually require a lot of evaluations of the objective and may converge prematurely. To alleviate these issues, Abdolmaleki et al. (2015) recently proposed a stochastic search framework called *model-based relative entropy stochastic search (MORE)*. In this framework, the new search distribution can be computed efficiently in a closed form using a learned model of the objective function. MORE outperformed state-of-the-art methods in stochastic optimization problems and single-context policy search problems, but its application to contextual policy search has not been explored yet. One of the contributions in this paper is a novel contextual policy search method in the MORE framework.

However, a naive extension of the original MORE would still suffer from high-dimensional contexts. Learning from high-dimensional variables, in fact, is still an important problem in statistics and machine learning (Bishop 2006). Nowadays, high-dimensional data (e.g., camera images) can often be obtained quite easily, but obtaining informative low-dimensional variables (e.g., objects positions) is non-trivial and requires prior knowledge and/or human guidance.

In this paper, we propose a method to handle high-dimensional context variables by learning a low-rank representation of the objective function. We show that learning a low-rank representation corresponds to performing linear dimensionality reduction on the context variables. Since optimization with a rank constraint is generally NP-hard, we minimize the *nuclear norm* (also called trace norm), which is a *convex* surrogate of the rank function (Recht, Fazel, and Parrilo 2010). This minimization allows us to learn a low-rank representation in a fully supervised manner by just solving a convex optimization problem. We evaluate the proposed method on a synthetic task with known ground truth and on robotic ball hitting tasks based on camera images. The evaluation shows that the proposed method with nuclear norm minimization outperforms the methods that naively perform principal component analysis to reduce the dimensionality of context variables.

Contextual Policy Search

In this section, we formulate the direct contextual policy search problem and briefly discuss existing methods.

Problem Formulation

The direct contextual policy search is formulated as follows. An agent observes the context variable $\boldsymbol{c} \in \mathbb{R}^{d_\mathbf{c}}$ and draws a parameter $\boldsymbol{\theta} \in \mathbb{R}^{d_\boldsymbol{\theta}}$ from a search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$. Subsequently, the agent executes a policy with the parameter $\boldsymbol{\theta}$ and observes a scalar reward computed by a reward function $R(\boldsymbol{\theta}, \boldsymbol{c})$. The goal is to find a search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$ maximizing the expected reward

$$\iint \mu(\boldsymbol{c}) p(\boldsymbol{\theta}|\boldsymbol{c}) R(\boldsymbol{\theta}, \boldsymbol{c}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c}, \tag{1}$$

where $\mu(\boldsymbol{c})$ denotes the context distribution. We assume that the reward function $R(\boldsymbol{\theta}, \boldsymbol{c})$ itself is unknown, but the agent can always access the reward value. We stress that context variables are fixed during task execution and they are drawn independently from $\mu(\boldsymbol{c})$. Thus, context variables are different from state variables in standard direct policy search.

Related Work

In the basic direct contextual policy search framework, the agent iteratively collects samples $\{(\boldsymbol{\theta}_n, \boldsymbol{c}_n, R(\boldsymbol{\theta}_n, \boldsymbol{c}_n))\}_{n=1}^N$ using a sampling distribution $q(\boldsymbol{\theta}|\boldsymbol{c})$. Subsequently, it computes a new search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$ such that the expected reward increases or is maximized. In literature, different approaches have been used to compute the new search distribution, e.g., evolutionary strategies (Hansen, Müller, and Koumoutsakos 2003), expectation-maximization algorithms (Kober, Oztop, and Peters 2011), or information theoretic approaches (Deisenroth, Neumann, and Peters 2013).

Most of the existing direct contextual policy search methods focus on tasks with low-dimensional context variables. To learn from high-dimensional context variables, usually the problem of learning a low-dimensional context representation is separated from the direct policy search by preprocessing the context space. However, unsupervised linear dimensionality reduction techniques are insufficient in problems where the latent representation contains distractor dimensions that do not influence the reward. A prominent example is principal component analysis (PCA) (Jolliffe 1986), that does not take the supervisory signal into account and therefore cannot discriminate between relevant and irrelevant latent dimensions. On the other hand, supervised linear dimensionality reduction techniques require a suitable response variable. However, manually defining such variables in nontrivial for many problems. Moreover, they often involve non-convex optimization and suffer from local optima (Fukumizu, Bach, and Jordan 2009; Suzuki and Sugiyama 2013).

In the last years, non-linear dimensionality reduction techniques based on deep learning have gained popularity (Bengio 2009). For instance, Watter et al. (2015) proposed a generative deep network to learn low-dimensional representations of images in order to capture information about the system transition dynamics and allow optimal control problems to be solved in low-dimensional spaces. More recently, Silver et al. (2016) successfully trained a machine to play a high-level game of *go* using a deep convolutional network. Although their work does not directly focus on dimensionality reduction, deep convolutional networks are known to be able to extract meaningful data representations. Thus, the effect of dimensionality reduction is achieved.

However, deep learning approaches generally require large datasets that are difficult to obtain in real-world scenarios (e.g., robotics). Furthermore, they involve solving non-convex optimization, which can suffer from local optima.

In this paper, we tackle the issues raised above. First, the proposed approach integrates supervised linear dimensionality reduction on the context variables by learning a *low-rank representation* for the reward model. Second, the problem is formalized as a *convex* optimization problem and is therefore guaranteed to converge to a global optimum.

Contextual MORE

The original MORE (Abdolmaleki et al. 2015) finds a search distribution (without context) maximizing the expected reward while upper-bounding the Kullback-Leibler (KL) divergence (Kullback and Leibler 1951) between successive search distributions and lower-bounding the entropy of the new search distribution. The KL and the entropy bounds control the exploration-exploitation trade-off. The key insight of MORE is to learn a reward model to efficiently compute a new search distribution in closed form. Below, we propose our method called *contextual model-based relative entropy stochastic search* (C-MORE), which is a direct contextual policy search method in the MORE framework.

Learning the Search Distribution

The goal of C-MORE is to find a search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$ that maximizes the expected reward while upper-bounding the expected KL divergence between $p(\boldsymbol{\theta}|\boldsymbol{c})$ and $q(\boldsymbol{\theta}|\boldsymbol{c})$, and lower-bounding the expected entropy of $p(\boldsymbol{\theta}|\boldsymbol{c})$. Formally,

$$\begin{aligned}
&\max_p \iint \mu(\boldsymbol{c}) p(\boldsymbol{\theta}|\boldsymbol{c}) R(\boldsymbol{\theta}, \boldsymbol{c}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c}, \\
&\text{s.t.} \iint \mu(\boldsymbol{c}) p(\boldsymbol{\theta}|\boldsymbol{c}) \log \frac{p(\boldsymbol{\theta}|\boldsymbol{c})}{q(\boldsymbol{\theta}|\boldsymbol{c})} \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c} \leq \epsilon, \\
&- \iint \mu(\boldsymbol{c}) p(\boldsymbol{\theta}|\boldsymbol{c}) \log p(\boldsymbol{\theta}|\boldsymbol{c}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c} \geq \beta, \\
&\iint \mu(\boldsymbol{c}) p(\boldsymbol{\theta}|\boldsymbol{c}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c} = 1,
\end{aligned}$$

where the KL upper-bound ϵ and the entropy lower-bound β are parameters specified by the user. The former is fixed for the whole learning process. The latter is adaptively changed according to the percentage of the relative difference between the sampling policy's expected entropy and the minimal entropy, as described by Abdolmaleki et al. (2015), i.e.,

$$\beta = \gamma(\mathbb{E}[H(q)] - H_0) + H_0,$$

where $\mathbb{E}[H(q)] = -\iint \mu(\boldsymbol{c}) q(\boldsymbol{\theta}|\boldsymbol{c}) \log q(\boldsymbol{\theta}|\boldsymbol{c}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\boldsymbol{c}$ is the sampling policy's expected entropy and H_0 is the minimal entropy. In the experiments, we set $\gamma = 0.99$ and $H_0 =$

Algorithm 1: C-MORE

Input: Parameters ϵ and β, initial distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$

for $k = 1, \ldots, K$ **do**
  for $n = 1, \ldots, N$ **do**
    Observe context $\boldsymbol{c}_n \sim \mu(\boldsymbol{c})$
    Draw parameter $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta}|\boldsymbol{c}_n)$
    Execute task with $\boldsymbol{\theta}_n$ and receive $R(\boldsymbol{\theta}_n, \boldsymbol{c}_n)$
  Learn the quadratic model $\widehat{R}(\boldsymbol{\theta}, \boldsymbol{c})$
  Solve $\operatorname{argmin}_{\eta>0,\omega>0} g(\eta, \omega)$ using Eq. (6)
  Set new search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$ using Eq. (7)

-150. The above optimization problem can be solved by the method of Lagrange multipliers[1]. The solution is given by

$$p(\boldsymbol{\theta}|\boldsymbol{c}) = q(\boldsymbol{\theta}|\boldsymbol{c})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{R(\boldsymbol{\theta}, \boldsymbol{c})}{\eta+\omega}\right) \exp\left(-\frac{\eta+\omega-\gamma}{\eta+\omega}\right).$$

The Lagrange multipliers $\eta > 0$ and $\omega > 0$ are obtained by minimizing

$$g(\eta, \omega) = \eta\epsilon - \omega\beta + (\eta+\omega) \int \mu(\boldsymbol{c}) h(\boldsymbol{c}, \eta, \omega) \mathrm{d}\boldsymbol{c}, \quad (2)$$

where

$$h(\boldsymbol{c}, \eta, \omega) = \log \int q(\boldsymbol{\theta}|\boldsymbol{c})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{R(\boldsymbol{\theta}, \boldsymbol{c})}{\eta+\omega}\right) \mathrm{d}\boldsymbol{\theta}. \quad (3)$$

Evaluating $h(\boldsymbol{c}, \eta, \omega)$ is not trivial due to the integration over $q(\boldsymbol{\theta}|\boldsymbol{c})^{\frac{\eta}{\eta+\omega}}$, that cannot be approximated straightforwardly by sample averages. Below, we describe how to solve this issue and evaluate the dual function from data.

Dual Function Evaluation via the Quadratic Model

We assume that the reward function $R(\boldsymbol{\theta}, \boldsymbol{c})$ can be approximated by a quadratic model

$$\widehat{R}(\boldsymbol{\theta}, \boldsymbol{c}) = \boldsymbol{\theta}^\top \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{c}^\top \boldsymbol{B}\boldsymbol{c} + 2\boldsymbol{\theta}^\top \boldsymbol{D}\boldsymbol{c} + \boldsymbol{\theta}^\top \boldsymbol{r}_1 + \boldsymbol{c}^\top \boldsymbol{r}_2 + r_0, \quad (4)$$

where $\boldsymbol{A} \in \mathbb{R}^{d_{\boldsymbol{\theta}} \times d_{\boldsymbol{\theta}}}, \boldsymbol{B} \in \mathbb{R}^{d_{\boldsymbol{c}} \times d_{\boldsymbol{c}}}, \boldsymbol{D} \in \mathbb{R}^{d_{\boldsymbol{\theta}} \times d_{\boldsymbol{c}}}, \boldsymbol{r}_1 \in \mathbb{R}^{d_{\boldsymbol{\theta}}}, \boldsymbol{r}_2 \in \mathbb{R}^{d_{\boldsymbol{c}}}$, and $r_0 \in \mathbb{R}$ are the model parameters. Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are symmetric. We also assume the sampling distribution $q(\boldsymbol{\theta}|\boldsymbol{c})$ to be Gaussian of the form

$$q(\boldsymbol{\theta}|\boldsymbol{c}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{b} + \boldsymbol{K}\boldsymbol{c}, \boldsymbol{Q}). \quad (5)$$

Under these assumptions, the dual function in Eq. (2) can be expressed as

$$g(\eta, \omega) = \eta\epsilon - \omega\beta + \frac{1}{2}\Big(\boldsymbol{f}^\top \boldsymbol{F}^{-1} \boldsymbol{f} - \eta \boldsymbol{b}^\top \boldsymbol{Q}^{-1} \boldsymbol{b} + (\eta+\omega) \log|2\pi \boldsymbol{F}^{-1}(\eta+\omega)| - \eta \log|2\pi\boldsymbol{Q}|\Big) + \int \mu(\boldsymbol{c}) \left(\boldsymbol{c}^\top \boldsymbol{m} + \frac{1}{2}\boldsymbol{c}^\top \boldsymbol{M}\boldsymbol{c}\right) \mathrm{d}\boldsymbol{c}, \quad (6)$$

[1] All derivations are given in the supplementary material.

where

$$\begin{aligned} \boldsymbol{f} &= \eta\boldsymbol{Q}^{-1}\boldsymbol{b} + \boldsymbol{r}_1, \\ \boldsymbol{F} &= \eta\boldsymbol{Q}^{-1} - 2\boldsymbol{A}, \\ \boldsymbol{m} &= \boldsymbol{L}^\top \boldsymbol{F}^{-1} \boldsymbol{f} - \eta \boldsymbol{K}^\top \boldsymbol{Q}^{-1} \boldsymbol{b}, \\ \boldsymbol{M} &= \boldsymbol{L}^\top \boldsymbol{F}^{-1} \boldsymbol{L} - \eta \boldsymbol{K}^\top \boldsymbol{Q}^{-1} \boldsymbol{K}, \\ \boldsymbol{L} &= \eta\boldsymbol{Q}^{-1}\boldsymbol{K} + 2\boldsymbol{D}. \end{aligned}$$

Since the context distribution $\mu(\boldsymbol{c})$ is unknown, we approximate the expectation in Eq. (6) by sample averages. The dual function can be minimized by standard non-linear optimization routines such as IPOPT (Wächter and Biegler 2006). Finally, using Eq. (4) and Eq. (5) the new search distribution $p(\boldsymbol{\theta}|\boldsymbol{c})$ is computed in closed form as

$$p(\boldsymbol{\theta}|\boldsymbol{c}) = \mathcal{N}\Big(\boldsymbol{\theta}|\boldsymbol{F}^{-1}\boldsymbol{f} + \boldsymbol{F}^{-1}\boldsymbol{L}\boldsymbol{c}, \boldsymbol{F}^{-1}(\eta+\omega)\Big). \quad (7)$$

To ensure that the covariance $\boldsymbol{F}^{-1}(\eta+\omega)$ is positive definite, the matrix $\boldsymbol{A}$ of the quadratic model is constrained to be negative definite. C-MORE is summarized in Algorithm 1.

Learning the Quadratic Model

The performance of C-MORE depends on the accuracy of the quadratic model. For many problems, the reward function $R(\boldsymbol{\theta}, \boldsymbol{c})$ is not quadratic and the quadratic model is not suitable to approximate the entire reward function. However, the reward function is often smooth and it can *locally* be approximated by a quadratic model. Therefore, we locally approximate the reward function by learning a new quadratic model for each policy update. The quadratic model can be learned by regression methods such as ridge regression[2] (Bishop 2006). However, ridge regression is prone to error when the context is high-dimensional. Below, we address this issue by firstly showing that performing linear dimensionality reduction on the context variables yields a low-rank matrix of parameters. Secondly, we propose a nuclear norm minimization approach to learn a low-rank matrix without explicitly performing dimensionality reduction.

Dimensionality Reduction and Low-Rank Representation

Linear dimensionality reduction learns a low-rank matrix $\boldsymbol{W}$ and projects the data onto a lower dimensional subspace. Performing linear dimensionality reduction on the context variables yields the following quadratic model

$$\widehat{R}(\boldsymbol{\theta}, \boldsymbol{c}) = \boldsymbol{\theta}^\top \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{c}^\top \boldsymbol{W}^\top \widetilde{\boldsymbol{B}} \boldsymbol{W}\boldsymbol{c} + 2\boldsymbol{\theta}^\top \widetilde{\boldsymbol{D}} \boldsymbol{W}\boldsymbol{c} + \boldsymbol{\theta}^\top \boldsymbol{r}_1 + \boldsymbol{c}^\top \boldsymbol{W}^\top \widetilde{\boldsymbol{r}}_2 + r_0, \quad (8)$$

where $\boldsymbol{W} \in \mathbb{R}^{d_{\mathbf{z}} \times d_{\mathbf{c}}}$ denotes a rank-$d_{\mathbf{z}}$ matrix with $d_{\mathbf{z}} < d_{\mathbf{c}}$. The model parameters $\boldsymbol{A}, \widetilde{\boldsymbol{B}}, \widetilde{\boldsymbol{D}}, \boldsymbol{r}_1, \widetilde{\boldsymbol{r}}_2$ and r_0 can be learned by ridge regression. However, the matrix $\boldsymbol{B} =$

[2] After learning the parameters, $\boldsymbol{A}$ is enforced to be negative definite by truncating its positive eigenvalues. Subsequently, we re-learn the remainder parameters. An alternative approach is projected gradient descend, but it is more computationally demanding and requires step size tuning.

$\boldsymbol{W}^\top \widetilde{\boldsymbol{B}} \boldsymbol{W}$ is low-rank, i.e., $\mathrm{rank}(\boldsymbol{B}) = d_{\mathbf{z}} < d_{\mathbf{c}}$. Thus, performing linear dimensionality reduction on the contexts makes $\boldsymbol{B}$ low-rank. Note that the rank of $\boldsymbol{D} = \widetilde{\boldsymbol{D}} \boldsymbol{W}$ depends on $\boldsymbol{\theta}$ and is problem dependent. Hence, we do not consider the rank of $\boldsymbol{D}$ for dimensionality reduction.

There are several linear dimensionality reduction methods that can be applied to learn $\boldsymbol{W}$. Principal component analysis (PCA) (Jolliffe 1986) is a common method used in statistics and machine learning. However, being unsupervised, it does not take the regression targets into account, i.e., the reward. Alternative supervised techniques, such as KDR (Fukumizu, Bach, and Jordan 2009) and LSDR (Suzuki and Sugiyama 2013), do not take the regression model, i.e., the quadratic model, into account. On the contrary, in projection regression (Friedman and Stuetzle 1981; Vijayakumar and Schaal 2000) the model parameters and the projection matrix are learned simultaneously. However, applying this approach to the model in Eq. (8) requires alternately optimizing for the model parameters and the projection matrix and is computationally expensive.

In the original MORE, Bayesian dimensionality reduction (Gönen 2013) is applied to perform linear supervised dimensionality reduction on $\boldsymbol{\theta}$, i.e., the algorithm considers a projection $\boldsymbol{W}\boldsymbol{\theta}$. The matrix $\boldsymbol{W}$ is sampled from a prior distribution and the algorithm learns the model parameters using weighted average over the sampled $\boldsymbol{W}$. However, for high-dimensional $\boldsymbol{W}$, this approach requires an impractically large amount of samples $\boldsymbol{W}$ to obtain an accurate model, leading to computationally expensive updates.

Learning a Low-Rank Matrix with Nuclear Norm Regularization

The quadratic model in Eq. (4) can be re-written as

$$\widehat{R}(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{H} \boldsymbol{x},$$

where the input vector $\boldsymbol{x}$ and the parameter matrix $\boldsymbol{H}$ are defined as

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\theta} \\ \boldsymbol{c} \\ 1 \end{bmatrix}, \quad \boldsymbol{H} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{D} & 0.5\boldsymbol{r}_1 \\ \boldsymbol{D}^\top & \boldsymbol{B} & 0.5\boldsymbol{r}_2 \\ 0.5\boldsymbol{r}_1^\top & 0.5\boldsymbol{r}_2^\top & r_0 \end{bmatrix}.$$

Note that $\boldsymbol{H}$ is symmetric since both $\boldsymbol{A}$ and $\boldsymbol{B}$ are symmetric. As discussed in the previous section, we desire $\boldsymbol{B}$ to be low-rank. Unlike Eq. (8), we do not consider dimensionality reduction for the linear terms in $\boldsymbol{c}$, i.e., $2\boldsymbol{\theta}^\top \boldsymbol{D}\boldsymbol{c}$ and $\boldsymbol{c}^\top \boldsymbol{r}_2$. Instead, we learn $\boldsymbol{H}$ by solving the following convex optimization problem

$$\min_{\boldsymbol{H}} \left[\mathcal{J}(\boldsymbol{H}) + \lambda_* \|\boldsymbol{B}\|_* \right], \quad \text{s.t. } \boldsymbol{A} \text{ is negative definite}, \tag{9}$$

where $\mathcal{J}(\boldsymbol{H})$ denotes the differentiable part

$$\mathcal{J}(\boldsymbol{H}) = \frac{1}{2N} \sum_{n=1}^{N} \left(\boldsymbol{x}_n^\top \boldsymbol{H} \boldsymbol{x}_n - R(\boldsymbol{\theta}_n, \boldsymbol{c}_n) \right)^2 + \frac{\lambda}{2} \|\boldsymbol{H}\|_{\mathrm{F}}^2,$$

where $\lambda > 0$ and $\lambda_* > 0$ are regularization parameters. The Frobenius norm $\|\cdot\|_{\mathrm{F}}$ is defined as $\|\boldsymbol{H}\|_{\mathrm{F}} = \sqrt{\mathrm{tr}(\boldsymbol{H}\boldsymbol{H}^\top)}$. The nuclear norm of a matrix $\|\cdot\|_*$ is defined as the ℓ_1-norm of its singular values . This optimization problem can be explained as follows. The term $\mathcal{J}(\boldsymbol{H})$ consists of the mean squared error and the ℓ_2-regularization term. Thus, minimizing $\mathcal{J}(\boldsymbol{H})$ corresponds to ridge regression. Minimizing the nuclear norm $\|\boldsymbol{B}\|_*$ shrinks the singular values of $\boldsymbol{B}$. Thus, the solution tends to have sparse singular values and to be low-rank. The negative definite constraint further ensures that the covariance matrix in Eq. (7) is positive definite.

The convexity of this optimization problem can be verified by checking the following conditions. First, the convexity of the mean squared error can be proven following Boyd and Vandenberghe 2004 (page 74). Let $g(t) = \widehat{\mathcal{J}}(\boldsymbol{Z} + t\boldsymbol{V})$ be the mean squared error and $\boldsymbol{Z}$ and $\boldsymbol{V}$ are symmetric matrices. Then we have that $\nabla^2 g(t) = \frac{1}{N} \sum (\boldsymbol{x}_n^\top \boldsymbol{V} \boldsymbol{x}_n)^2 \geq 0$. Thus, the mean squared error is convex. Since the Frobenius norm is convex, $\mathcal{J}(\boldsymbol{H})$ is convex as well. Second, a set of negative definite matrices is convex since $\boldsymbol{y}^\top (a\boldsymbol{X} + (1-a)\boldsymbol{Y})\boldsymbol{y} < 0$ for any negative definite matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$, $0 \leq a \leq 1$, and any vector $\boldsymbol{y}$ (Boyd and Vandenberghe 2004). Third, the nuclear norm is a convex function (Recht, Fazel, and Parrilo 2010). Note that, since the gradient $\nabla \mathcal{J}(\boldsymbol{H})$ is symmetric, $\boldsymbol{H}$ is guaranteed to be symmetric as well given that the initial solution is also symmetric.

It is also possible to enforce the matrix $\boldsymbol{H}$ (rather than $\boldsymbol{B}$) to be low-rank, implying that both $\boldsymbol{\theta}$ and $\boldsymbol{c}$ can be projected onto a common low-dimensional subspace. However, this is often not the case, and regularizing by the nuclear norm of $\boldsymbol{H}$ did not perform well in our experiments. We may also directly constrain $\mathrm{rank}(\boldsymbol{B}) = d_{\mathbf{z}}$ in Eq. (9) instead of performing nuclear norm regularization. However, minimization problems with rank constraints are NP-hard. On the contrary, the nuclear norm is the convex envelop of the rank function and can be optimized more efficiently (Recht, Fazel, and Parrilo 2010). For this reason, the nuclear norm has been a popular surrogate to a low-rank constraint in many applications, such as matrix completion (Candès and Tao 2010) and multi-task learning (Pong et al. 2010).

Since the optimization problem in Eq. (9) is convex, any convex optimization method can be used (Boyd and Vandenberghe 2004). For our experiments, we use the *accelerated proximal gradient descend (APG)* (Toh and Yun 2009). The pseudocode of our implementation of APG for solving Eq. (9) is given in the supplementary material. Note that APG requires computing the SVD of the matrix $\boldsymbol{B}$. Since computing the exact SVD of a high-dimensional matrix can be computationally expensive, we approximate it by randomized SVD (Halko, Martinsson, and Tropp 2011).

Experiments

We evaluate the proposed method on three problems. We start by studying C-MORE behavior in a scenario where we know the true reward model and the true low-dimensional context. Subsequently, we focus our attention on two simulated robotic ball hitting tasks. In the first task, a toy 2-DoF planar robot arm has to hit a ball placed on a plane. In the second task, a simulated 6-DoF robot arm has to hit a ball placed in a three-dimensional space. In both cases, the robots

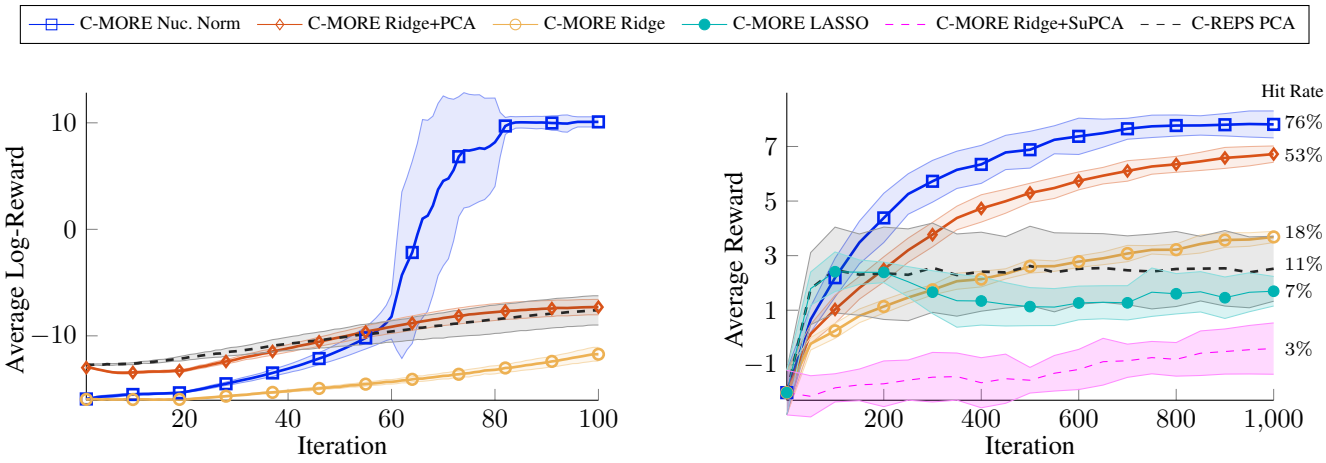


Figure 1: Average reward for the quadratic cost function problem. Shaded area denotes standard deviation (results are averaged over ten trials). Only C-MORE Nuc. Norm converges within 100 iterations to an almost optimal policy.

Figure 2: Averaged reward for the 2-DoF hitting task. C-REPS outperforms C-MORE early on. However, it prematurely converges to suboptimal solutions, while C-MORE continues to improve and soon outperforms C-REPS.

accomplish their task by using raw camera images as context variables. However, in the latter case we have limited data and therefore sample efficiency is of primary importance.

The evaluation is performed on three different versions of C-MORE, according to the model learning approach: using only ridge regression (*C-MORE Ridge*), aided by a low-dimensional context variables learned by PCA (*C-MORE Ridge+PCA*) and aided by nuclear norm regularization (*C-MORE Nuc. Norm*). We also use *C-REPS* (Deisenroth, Neumann, and Peters 2013) with PCA as baseline. For the ball hitting task with 2-DoF robot arm, we additionally evaluate C-MORE with model learned by LASSO (*C-MORE LASSO*), and ridge regression with low-dimensional context variables learned by supervised PCA (Li et al. 2016) (*C-MORE Ridge+SuPCA*). We also tried to preprocess the context space with an autoencoder. However, the learned representation performed poorly, possibly due to the limited amount of data, and therefore this method is not reported.

For each case study, first, the experiments are presented and then the results are reported and discussed. For additional details such as computation time of each method and sensitivity to the regularization parameters of C-MORE Nuc. Norm, we refer to the supplementary material.

Quadratic Cost Function Optimization

In the first experiment, we want to study the performance of the algorithms in a setup where we are able to analytically compute both the reward and the true low-dimensional context. To this aim, we define the following problem

$$R(\boldsymbol{\theta}, \boldsymbol{c}) = -(||\boldsymbol{\theta} - \boldsymbol{T}_1\tilde{\boldsymbol{c}}||_2)^2, \quad \tilde{\boldsymbol{c}} = \widetilde{\boldsymbol{I}}\boldsymbol{T}_2^{-1}\boldsymbol{c},$$
$$\boldsymbol{T}_1 \in \mathbb{R}^{d_{\boldsymbol{\theta}} \times d_{\tilde{\mathbf{c}}}}, \ \boldsymbol{T}_2 \in \mathbb{R}^{d_{\mathbf{c}} \times d_{\mathbf{c}}}, \ \widetilde{\boldsymbol{I}} \in \mathbb{R}^{d_{\tilde{\mathbf{c}}} \times d_{\mathbf{c}}}, \ d_{\tilde{\mathbf{c}}} < d_{\mathbf{c}},$$

where $\widetilde{\boldsymbol{I}}$ is a rectangular matrix with ones in its main diagonal and zeros otherwise, $\tilde{\boldsymbol{c}}$ is the true low-dimensional context, and $\boldsymbol{T}_1$ is to match the dimension of the true context and the parameter $\boldsymbol{\theta}$ in order to compute the reward. This setup is particularly interesting because only a subset of the observed context influences the reward. First, the observed context $\boldsymbol{c}$ is linearly transformed by $\boldsymbol{T}_2^{-1}$. Subsequently, thanks to the matrix $\widetilde{\boldsymbol{I}}$, only the first $d_{\tilde{\mathbf{c}}}$ elements are kept to compose the true context, while the remainder is treated as noise. Finally, the reward is computed by linearly transforming the true context by $\boldsymbol{T}_1$.

We set $d_{\tilde{\mathbf{c}}} = 3, d_{\boldsymbol{\theta}} = 10, d_{\mathbf{c}} = 25$, while the elements of $\boldsymbol{T}_1, \boldsymbol{T}_2$ are chosen uniformly randomly in $[0, 1]$. The sampling Gaussian distribution is initialized with random mean and covariance $\boldsymbol{Q} = 10,000\boldsymbol{I}$. For learning, we collect 35 new samples and keeps track of the samples collected during the last 20 iterations to stabilize the policy update. The evaluation is performed at each iteration over 1,000 contexts. Each context element is drawn from a uniform random distribution in $[-10, 10]$. Since we can generate a large amount of data in this setting, we pre-train PCA using 10,000 random context samples and fixed the dimensionality to $d_{\mathbf{z}} = 20$ (chosen by cross-validation). The learning is performed for a maximum of 100 iterations. If the KL divergence is lower than 0.1, then the learning is considered to be converged and the policy is not updated anymore.

As shown in Figure 1, C-MORE Nuc. Norm clearly outperforms all the competitors, learning an almost optimal policy and being the only one to converge within the maximum number of iterations. It is also the only algorithm correctly learning the true context dimensionality, as nuclear norm successfully regularizes $\boldsymbol{B}$ to have rank three. On the contrary, PCA does not help C-MORE much and yields only slightly better results than plain ridge regression. PCA cannot in fact determine task-relevant dimensions as non-relevant dimensions have equally-high variance.

Ball Hitting with a 2-DoF Robot Arm

In this task, a simulated planar robot arm (Figure 3) has to hit a green virtual ball placed on RGB camera images of size

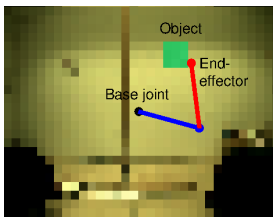
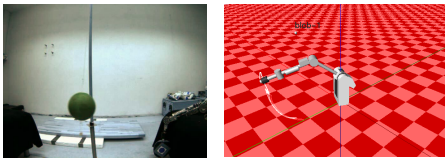


Figure 3: 2-DoF hitting task. The context (blue and red lines) consists of a virtual green ball and the background image.



(a) (b)

Figure 4: The 6-DoF robot as seen from the camera (Figure 4a, bottom right) and in simulation (Figure 4b). The goal is to control the robot to hit the green ball according to camera images, resized to 32×24.

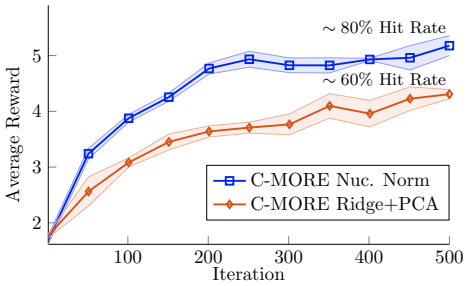


Figure 5: 6-DoF hitting task results (averaged over three trials). Nuclear norm regularization outperforms PCA, both in terms of reward and accuracy.

32×24. The observed pixels define the context, for a total of 2304 context variables. The ball is randomly and uniformly placed in the robot workspace. Noise drawn from a uniform random distribution in $[-30, 30]$ is added to the context to simulate different light conditions. The robot controls the joint accelerations at each time step by a linear-in-parameter controller with Gaussian basis functions, for a total of 32 parameters $\boldsymbol{\theta}$ to be learned. The reward $R(\boldsymbol{\theta}, \boldsymbol{c})$ is the negative cumulative joint accelerations plus the negative distance between the end-effector and the ball at the final time step.

For learning, the agent collects 50 samples at each iteration and keeps samples from the last four previous iterations. The evaluation is performed at each iteration over 500 contexts. Pixel values are normalized in $[-1, 1]$. The sampling Gaussian distribution is initialized with random mean and identity covariance. For C-MORE Nuc. Norm, C-MORE LASSO and C-MORE PCA, we perform 5-fold cross-validation every 100 policy updates to choose the values of regularization parameter for nuclear norm, regularization parameter for ℓ_1 norm, and dimension $d_{\mathbf{z}}$, respectively. The decision is based on the mean squared error between the collected returns and the model-predicted ones. Due to high computation time of C-MORE SuPCA for high values of $d_{\mathbf{z}}$, we tried different values of $d_{\mathbf{z}} \in \{5, 7, 10\}$ and selected $d_{\mathbf{z}} = 10$ which gave the best result[3]. Similarly for C-REPS PCA, we tried different values of $d_{\mathbf{z}} \in \{10, 20, 30, 40\}$ and selected $d_{\mathbf{z}} = 10$ which gave the best result.

Figure 2 shows the averaged reward against the number of iterations. Once again, C-MORE aided by nuclear norm regularization performs the best, achieving the highest average reward. At the 1000th iteration, the learned controller hits the ball with 76% accuracy. The rank of its learned matrix $\boldsymbol{B}$ is approximately 31, which shows that the algorithm successfully learns a low-rank model representation. The model learned by LASSO performs very poorly and it is even outperformed by plain ridge regression. However, this is unsurprising since the context variables are highly correlated and LASSO is known to not work well for such variables. On the contrary, preprocessing the context space through PCA still helps C-MORE (the rank of its learned $\boldsymbol{B}$ is approximately 25), but yields poor results for C-REPS, which suffers of premature convergence. Lastly, preprocessing the context space through SuPCA does not seem work well. This may be due to $d_{\mathbf{z}}$, which could be too small for this task.

[3]SuPCA with $d_{\mathbf{z}} = 15$ took approximately 5 minutes/iteration.

Ball Hitting with a 6-DoF Robot Arm

Similarly to the previous task, here a 6-DoF robotic arm has to hit a ball placed on a three-dimensional space, as shown in Figure 4. The context is once again defined by the vectorized pixels of RGB images of size 32×24, for a total of 2304 context variables. Note that Figure 4a shows an image before we rescale it to size 32×24. However, unlike the 2-DoF task, the ball is directly recorded by a real camera placed near the physical robot, and it is not virtually generated on the images. Furthermore, the robot is controlled by dynamic motor primitives (Ijspeert, Nakanishi, and Schaal 2002) (DMPs), which are non-linear dynamical systems. We use one DMP per joint, with five basis functions per DMP. We also learn the goal attractor of the DMPs, for a total of 36 parameters $\boldsymbol{\theta}$ to be learned. The reward $R(\boldsymbol{\theta}, \boldsymbol{c})$ is computed as the negative cumulative joint accelerations and minimum distance between the end-effector and the ball as well.

The image dataset is collected by taking pictures with the ball placed at 50 different positions. To increase the number of data points, we add a uniform random noise in $[-30, 30]$ to the context to simulate different light conditions. Therefore, although some samples determine the same ball position, they are considered different due to the added noise. The search distribution is initialized by imitation learning using 50 demonstration samples. For learning, the agent collects 50 samples at each iteration and always keeps samples from the last four previous iterations.

We only evaluate C-MORE with nuclear norm and PCA since they performed well in the previous evaluation. Figure 5 shows that nuclear norm again outperforms PCA. At the 500th iteration, the robot hits the ball with 80% accuracy. Considering that the robot is not able to hit the ball in some contexts due to physical constraints and can achieve a maximum accuracy of 90%, this accuracy is impressive for the task. The averaged rank of matrix $\boldsymbol{B}$ learned by the nuclear norm is approximately 25, which shows that minimizing the nuclear norm successfully learns a low-rank matrix. For PCA, the averaged rank of $\boldsymbol{B}$ is approximately 30.

Conclusion

Learning with high-dimensional context variables is a challenging and prominent problem in machine learning. In this paper, we proposed C-MORE, a novel contextual policy search method with integrated dimensionality reduction. C-MORE learns a reward model that is locally quadratic in the policy parameters and the context variables. By enforcing the model representation to be low-rank, we perform supervised linear dimensionality reduction. Unlike existing techniques relying on non-convex formulations, the nuclear norm allows us to learn the low-rank representation by solving a *convex* optimization problem, thus guaranteeing convergence to a global optimum. The main disadvantage of the proposed method is that it demands more computation time due to the nuclear norm regularization. Although we did not encounter severe problems in our experiments, for very large dimensional tasks this issue can be mitigated by using more efficient techniques, such as active subspace selection (Hsieh and Olsen 2014).

In this paper, we only focused on linear dimensionality reduction techniques. Recently, non-linear techniques based on deep network has been showing impressive performance (Bengio 2009; Watter et al. 2015). In future work, we will incorporate deep network into C-MORE, e.g., by using a deep convolutional network to represent the reward model.

Acknowledgement

This work was partially funded by KAKENHI and a DFG grant (priority program "Autonomous learning", SPP1527).

References

Abdolmaleki, A.; Lioutikov, R.; Peters, J.; Lau, N.; Reis, L. P.; and Neumann, G. 2015. Model-based relative entropy stochastic search. In *Advances in Neural Information Processing Systems*, 3537–3545.

Baxter, J., and Bartlett, P. L. 2000. Reinforcement learning in POMDP's via direct gradient ascent. In *International Conference on Machine Learning*, 41–48.

Bengio, Y. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2:1–127.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. New York, NY, USA: Cambridge University Press.

Candès, E. J., and Tao, T. 2010. The power of convex relaxation: near-optimal matrix completion. *IEEE Transactions on Information Theory* 56:2053–2080.

da Silva, B. C.; Konidaris, G.; and Barto, A. G. 2012. Learning parameterized skills. In *International Conference on Machine Learning*.

Deisenroth, M. P.; Neumann, G.; and Peters, J. 2013. A survey on policy search for robotics. *Foundations and Trends in Robotics* 2:1–142.

Friedman, J. H., and Stuetzle, W. 1981. Projection pursuit regression. *Journal of the American Statistical Association* 76:817–823.

Fukumizu, K.; Bach, F. R.; and Jordan, M. I. 2009. Kernel dimension reduction in regression. *The Annals of Statistics* 37:1871–1905.

Gönen, M. 2013. Bayesian supervised dimensionality reduction. *IEEE Transanscation on Cybernetics* 43:2179–2189.

Halko, N.; Martinsson, P.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review* 53:217–288.

Hansen, N.; Müller, S. D.; and Koumoutsakos, P. 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation* 11:1–18.

Hsieh, C., and Olsen, P. A. 2014. Nuclear norm minimization via active subspace selection. In *International Conference on Machine Learning*, 575–583.

Ijspeert, A. J.; Nakanishi, J.; and Schaal, S. 2002. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems*, 1523–1530.

Jolliffe, I. T. 1986. *Principal Component Analysis*. Springer Verlag.

Kober, J.; Oztop, E.; and Peters, J. 2011. Reinforcement learning to adjust robot movements to new situations. In *International Joint Conference on Artificial Intelligence*, 2650–2655.

Kullback, S., and Leibler, R. A. 1951. On information and sufficiency. *The Annals of Mathematical Statistics* 22:79–86.

Li, G.; Yang, D.; Nobel, A. B.; and Shen, H. 2016. Supervised singular value decomposition and its asymptotic properties. *Journal of Multivariate Analysis* 146:7–17.

Neumann, G. 2011. Variational inference for policy search in changing situations. In *International Conference on Machine Learning*, 817–824.

Pong, T. K.; Tseng, P.; Ji, S.; and Ye, J. 2010. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization* 20:3465–3489.

Recht, B.; Fazel, M.; and Parrilo, P. A. 2010. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review* 52:471–501.

Rosenstein, M. T., and Barto, A. G. 2001. Robot weightlifting by direct policy search. In *International Joint Conference on Artificial Intelligence*, 839–846.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the game of go with deep neural networks and tree search. *Nature* 529:484–503.

Suzuki, T., and Sugiyama, M. 2013. Sufficient dimension reduction via squared-loss mutual information estimation. *Neural Computation* 25:725–758.

Toh, K., and Yun, S. 2009. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. In *International Symposium on Mathematical Programming*.

Vijayakumar, S., and Schaal, S. 2000. Locally weighted projection regression: Incremental real time learning in high dimensional space. In *International Conference on Machine Learning*, 1079–1086.

Wächter, A., and Biegler, L. T. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106:25–57.

Watter, M.; Springenberg, J. T.; Boedecker, J.; and Riedmiller, M. A. 2015. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, 2746–2754.