Efficient Sparsification for Gaussian Process Regression

Article *in* Neurocomputing · March 2016

DOI: 10.1016/j.neucom.2016.02.032

3 authors, including:

Duy Nguyen-Tuong
Bosch Corporate Research
42 PUBLICATIONS **829** CITATIONS
SEE PROFILE

Marc Toussaint
Universität Stuttgart
147 PUBLICATIONS **2,231** CITATIONS
SEE PROFILE

Neurocomputing 192 (2016) 29–37

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Efficient sparsification for Gaussian process regression ☆

Jens Schreiter [a,b,*], Duy Nguyen-Tuong [a], Marc Toussaint [b]

[a] Cognitive Systems Group, Corporate Sector Research and Advance Engineering, Robert Bosch GmbH, 70465 Stuttgart, Germany
[b] Machine Learning and Robotics Laboratory, Institute for Parallel and Distributed Systems, University of Stuttgart, 70569 Stuttgart, Germany

ARTICLE INFO

Article history:
Received 29 June 2015
Received in revised form
17 November 2015
Accepted 1 February 2016
Available online 3 March 2016

Keywords:
Gaussian processes
Sparse approximations
Greedy subset selection
Learning robot inverse dynamics

ABSTRACT

Sparse Gaussian process models provide an efficient way to perform regression on large data sets. Sparsification approaches deal with the selection of a representative subset of available training data for inducing the sparse model approximation. A variety of insertion and deletion criteria have been proposed, but they either lack accuracy or suffer from high computational costs. In this paper, we present a new and straightforward criterion for successive selection and deletion of training points in sparse Gaussian process regression. The proposed novel strategies for sparsification are as fast as the purely randomized schemes and, thus, appropriate for applications in online learning. Experiments on real-world robot data demonstrate that our obtained regression models are competitive with the computationally intensive state-of-the-art methods in terms of generalization and accuracy. Furthermore, we employ our approach in learning inverse dynamics models for compliant robot control using very large data sets, i.e. with half a million training points. In this experiment, it is also shown that our approximated sparse Gaussian process model is sufficiently fast for real-time prediction in robot control.

1. Introduction

Nowadays, Gaussian processes (GPs) are a widely used non-parametric Bayesian modeling technique [2]. In contrast to other kernel approaches such as support vector machines (SVMs), see [3] for more details, GPs offer a probabilistic framework. This leads to predictive distributions for test points and model selection is easy to achieve with standard Bayesian procedures. However, the applicability of full Gaussian process regression (GPR) to large scale problems with a high number of training points n is limited due to the unfavourable scaling in training time and memory requirements. The dominating factors are usually $O(n^3)$ cost for solving a linear system and computing a logarithmic determinant with respect to a dense covariance matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ between all available training points and the $O(n^2)$ space required to store it in memory. Furthermore, the full GPR model needs $O(n^2)$ cost per predictive test point variance, as well as $O(\mathrm{dn})$ for predicting the mean, if a kernel evaluation costs $O(d)$, where d is the data dimension.

To overcome these limitations in computational cost and storage requirements, many approximations to full GPR have been proposed. In Fig. 1, an illustration showing different approximation approaches is presented. Local GPR approaches, e.g. as proposed in Nguyen-Tuong et al. [4], can be used to increase modeling performance. These methods are based on a partition of the input space, where for each region a local GP model is trained. On the other hand, more sophisticated GPR approximation techniques consider either approximations of the dense covariance matrix $\boldsymbol{K}$ or focus on sparse likelihood approximations. For example, covariance matrix approximations such as the Nyström method [6] can be employed to reduce modeling effort. Moreover, Fourier kernel approximations [7] like the sparse spectrum GPR scheme [8] directly consider an approximation of the specified covariance function to increase computational speed. Additionally, various sparse likelihood approximations have emerged recently, whose relations have been formalized in the unifying framework [9]. The fully independent training conditional (FITC) approximation [10] uses a flexible subset of virtual training points to generate a sparse GPR model and optimizes the virtual training points along with all other hyperparameters. In contrast, the deterministic training conditional (DTC) approximation selects a representative subset of real training points, the so-called active points, that induces the sparse likelihood approximation. A variational formalism for both sparse approximation techniques, which leads to a regularized log marginal likelihood for hyperparameter learning and the additional optimization of virtual training points with respect to the FITC approximation plus a new greedy selection method for the DTC approximation, is presented in [11]. Here, greedy schemes are

☆This paper is a longer and more detailed version of the publication [1].

* Corresponding author at: Cognitive Systems Group, Corporate Sector Research and Advance Engineering, Robert Bosch GmbH, 70465 Stuttgart, Germany.
E-mail address: jens.schreiter@de.bosch.com (J. Schreiter).

http://dx.doi.org/10.1016/j.neucom.2016.02.032
0925-2312/

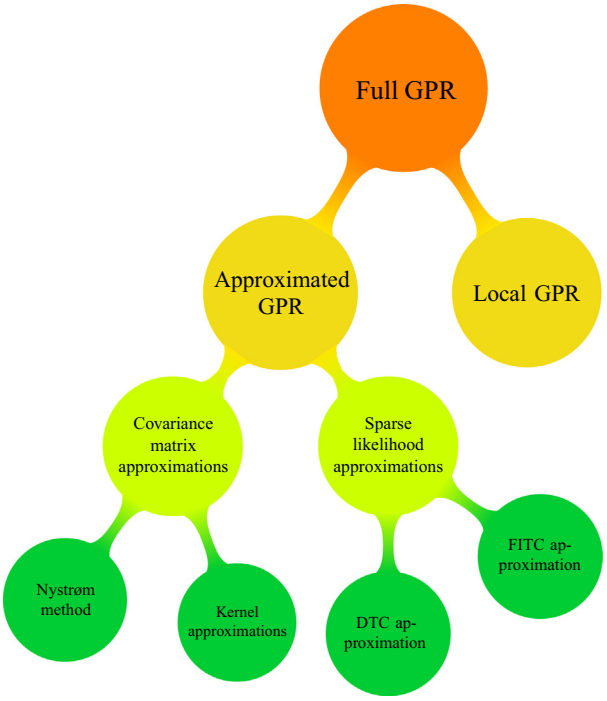


Fig. 1. Relation between different approximation techniques for Gaussian process regression. In this paper, we focus on the insertion and deletion strategies for the deterministic training conditional (DTC) approximation. Note that this illustration is by far not complete. For example, Gaussian process models based on network architectures [5] are not considered in this overview.

employed for the DTC approximation due to the high computational complexity of the optimal subset selection problem. A fast information gain criterion for insertion of training points to the active set is proposed in [12]. Smola and Bartlett, cf. [13], use a computationally costly selection heuristic which approximates the logarithmic marginal posterior probability. The same formalism as the previous criterion is used in [14], while improving computational performance by using a simpler approximation of the posterior probability. Quiñonero-Candela's [15] selection is based on the increase in the log marginal likelihood of the sparse GP by the insertion of a training point in the active subset. In [16], Csató and Opper measure the projection-induced error in the reproducing kernel Hilbert space (RKHS) and select the point which maximally extends the spanned subspace of the RKHS. Based on this idea, they also introduce a heuristic for deletion of training points from the active set. They show that removing active points can considerably reduce the prediction times for test points with only slightly decreasing generalization accuracy. All of the insertion and deletion methods mentioned above either lack computational speed, have high memory requirements, or lack of modeling accuracy. Moreover, if the regression model generation is based on a purely randomized selection or on a method with a small randomly selected subset of remaining training points for criteria evaluation, e.g. as done in [11,13–15], the performance in hard regression tasks deteriorates.

Our proposed novel sparsification method is closely related to the inclusion heuristic by Smola and Bartlett [13]. However, by employing some reasonable assumptions, we are able to significantly reduce the computational costs to the level of randomized selection, without a huge loss in model accuracy. Compared to the deletion criterion by Csató and Opper [16], our approach offers nearly the same prediction performance with lower computing time.

The remainder of the paper is organized as follows. In the following section, we introduce the sparse GPR setting and review the state-of-the-art sparsification criteria for inclusion and deletion. Our novel strategies for fast greedy insertion and deletion are presented in Section 3. We also describe an efficient way for learning the resulting hyperparameters with a generalized expectation maximization (EM) algorithm in our specific setup. In Section 4, we report on the results of our comprehensive comparison on several benchmark data sets for learning inverse dynamics models. Furthermore, we demonstrate the real-time applicability of our learned inverse dynamics models for a compliant robot control task. Finally, in Section 5, we discuss the results of our method and give directions for future work.

2. Sparse Gaussian process regression

Let $\mathcal{D}=(\boldsymbol{y},\boldsymbol{X})$ be the training data set, where $\boldsymbol{y}\in\mathbb{R}^n$ is a vector of noisy realizations y_i of the underlying regression function $f:\mathbb{R}^d\to\mathbb{R}$ with $f(\boldsymbol{x}_i)=f_i$, obeying the relationship $y_i=f_i+\varepsilon_i$ with Gaussian noise $\varepsilon_i\sim\mathcal{N}(0,\sigma^2)$. Furthermore, the n training inputs $\boldsymbol{x}_i\in\mathbb{R}^d$ are row-wise summarized in $\boldsymbol{X}\in\mathbb{R}^{n\times d}$. Our goal is the construction of a sparse GPR model for the underlying regression function. Csató [17] and Seeger [18] laid the foundation for this sparse GPR model under the DTC approximation which is presented below. We adopt the notation by Seeger et al. [12] to facilitate the comparability of the different criteria. Let I be the index set of size m of all active points $\boldsymbol{x}_i$ with $i\in I$, i.e. training points that represent the sparse approximation, and R be the index set of all remaining points, such that $I\cup R=\{1,\ldots,n\}$. The centered prior distribution over the latent function values $\boldsymbol{f}_I\in\mathbb{R}^m$ corresponding to the active subset is then given by

$$P(\boldsymbol{f}_I|\boldsymbol{X}_I)=\mathcal{N}(\boldsymbol{f}I|\boldsymbol{0},\boldsymbol{K}I)\quad\text{with}\quad\boldsymbol{K}I=\left(k(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)_{i,j\in I}\in\mathbb{R}^{m\times m}. \tag{1}$$

Here, $\boldsymbol{K}_I$ is the covariance matrix over the active training points determined through the specified covariance function $k_{ij}=k(\boldsymbol{x}_i,\boldsymbol{x}_j)$. The sparseness of this method is introduced via a likelihood approximation $Q_I(\boldsymbol{y}|\boldsymbol{f}_I,\boldsymbol{X})$ that is optimized with respect to the Kullback–Leibler divergence (KL-divergence) and induced through the active training points which leads to

$$Q_I(\boldsymbol{y}|\boldsymbol{f}_I,\boldsymbol{X})=\mathcal{N}\left(\boldsymbol{y}|\boldsymbol{P}_I^T\boldsymbol{f}_I,\sigma^2\boldsymbol{I}\right). \tag{2}$$

The projection matrix $\boldsymbol{P}_I=\boldsymbol{K}_I^{-1}\boldsymbol{K}_{I,\cdot}\in\mathbb{R}^{m\times n}$, where $\boldsymbol{K}_{I,\cdot}\in\mathbb{R}^{m\times n}$ comprises the covariance function values between all training points ($\cdot$ notation) and the active subset of training points, maps $\boldsymbol{f}_I$ to the prior conditional mean $\mathrm{E}[P(\boldsymbol{f}|\boldsymbol{f}_I)]=\boldsymbol{K}_{I,\cdot}T\boldsymbol{K}_I^{-1}\boldsymbol{f}_I\in\mathbb{R}^n$. With Bayesian inference we get the approximated posterior distribution

$$Q_I(\boldsymbol{f}_I|\boldsymbol{y},\boldsymbol{X})=\mathcal{N}\left(\boldsymbol{f}_I|\boldsymbol{L}\boldsymbol{M}^{-1}\boldsymbol{V}\boldsymbol{y},\sigma^2\boldsymbol{L}\boldsymbol{M}^{-1}\boldsymbol{L}^T\right), \tag{3}$$

which is proportional to the product of the prior in Eq. (1) and the approximated likelihood in Eq. (2). Here $\boldsymbol{L}\in\mathbb{R}^{m\times m}$ is the lower Cholesky factor of $\boldsymbol{K}_I$, $\boldsymbol{V}=\boldsymbol{L}^{-1}\boldsymbol{K}_{I,\cdot}\in\mathbb{R}^{m\times n}$, and $\boldsymbol{M}=\sigma^2\boldsymbol{I}+\boldsymbol{V}\boldsymbol{V}^T\in\mathbb{R}^{m\times m}$ for fixed I of size m. The approximated marginal likelihood directly follows from the integration over the same product about the active function values $\boldsymbol{f}I$ and results in

$$Q_I(\boldsymbol{y}|\boldsymbol{X})=\mathcal{N}\left(\boldsymbol{y}|\boldsymbol{0},\sigma^2\boldsymbol{I}+\boldsymbol{V}^T\boldsymbol{V}\right). \tag{4}$$

The predictive distribution for a test point $\boldsymbol{x}_*\in\mathbb{R}^d$ is given by

$$Q_I(\boldsymbol{f}_*|\boldsymbol{x}_*,\boldsymbol{y},\boldsymbol{X})=\mathcal{N}\Big(\boldsymbol{f}_*|\boldsymbol{k}_{I,*}^T\boldsymbol{L}^{-T}\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I,k_{**}-\|\boldsymbol{L}^{-1}\boldsymbol{k}_{I,*}\|^2+\sigma^2\|\boldsymbol{L}_M^{-1}\boldsymbol{L}^{-1}\boldsymbol{k}_{I,*}\|^2\Big) \tag{5}$$

with the Cholesky decomposition $\boldsymbol{M}=\boldsymbol{L}_M\boldsymbol{L}_M^T$, $\boldsymbol{\beta}_I=\boldsymbol{L}_M^{-1}\boldsymbol{V}\boldsymbol{y}\in\mathbb{R}^m$, and the covariance vector $\boldsymbol{k}_{I,*}\in\mathbb{R}^m$ between the test input and the active points. If only the predicted mean values are of interest, the prediction vector $\boldsymbol{\alpha}_I=\boldsymbol{L}^{-T}\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I$ can be precomputed to perform computations of mean values with only $\mathcal{O}(md)$ cost. Note that this cost depends on the calculation of the vector $\boldsymbol{k}_{I,*}$ and, thus, on the

specified covariance function, and is typically proportional to the input dimension d. The predictive variance is feasible in $\mathcal{O}(\Updownarrow^2)$ if $d < m$. The approximated posterior distribution for all training points $Q_I(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})$ induced through the active subset indexed by I is given by

$$Q_I(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X}) = \mathcal{N}\left(\boldsymbol{f}|\,\boldsymbol{V}^T\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I,\ \boldsymbol{K}-\boldsymbol{V}^T\boldsymbol{V}+\sigma^2\boldsymbol{V}^T\boldsymbol{M}^{-1}\boldsymbol{V}\right) \tag{6}$$

with the estimated mean vector $\boldsymbol{\mu}_I = \mathrm{E}\left[Q_I(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})\right] = \boldsymbol{V}^T\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I \in \mathbb{R}^n$. Due to the matrix–matrix multiplications, the training complexity of this sparse GPR model is $\mathcal{O}(nm^2)$. Predicting the mean for one test point is feasible in $O(\mathrm{dm})$.

2.1. State-of-the-art strategies for sparsification

Before the various state-of-the-art insertion and deletion schemes are presented, the most important symbols of the sparse GPR technique are summarized in Table 1. Most of the GP approximation techniques differ in the way, how the active set $\boldsymbol{X}_I$ is selected [9]. Sparsification deals with the insertion and deletion of training points to or from the active set. Usually, the remaining point $\boldsymbol{x}_i$ with $i \in R$ that has maximum gain with respect to an insertion criterion Δ_i will be selected. Analogously, we will remove the active point from the current posterior model given by Eq. (6) that has minimal loss with respect to a deletion criterion ∇_i. In the following, let $I' = I \cup \{i\}$. One of the simplest and fastest point selection and deletion methods is to randomly select one of the training points. This approach provides the baseline strategy to which we compare all other methods.

Currently, one of the best selection methods with respect to modeling accuracy is proposed by Smola and Bartlett. Their greedy scheme [13] select the remaining point that maximizes the posterior likelihood

$$P(\boldsymbol{\alpha}|\boldsymbol{y},\boldsymbol{X}) \propto P(\boldsymbol{y}|\boldsymbol{\alpha},\boldsymbol{X})P(\boldsymbol{\alpha}|\boldsymbol{X}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{K}\boldsymbol{\alpha},\sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{\alpha}|\boldsymbol{0},\boldsymbol{K}) \tag{7}$$

for the admission of the prediction vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ of the full GP model under the given data set $\mathcal{D}$. This likelihood approach is based on the transformation of $\boldsymbol{\alpha} = \boldsymbol{K}^{-1}\boldsymbol{f}$ and leads to the equivalent formulation

$$\tau_I = \min_{\boldsymbol{\alpha}_I}(\tau(\boldsymbol{\alpha}_I)) = \min_{\boldsymbol{\alpha}_I}\left(\frac{1}{2}\boldsymbol{\alpha}_I^T\boldsymbol{L}\boldsymbol{M}\boldsymbol{L}^T\boldsymbol{\alpha}_I - \boldsymbol{\alpha}_I^T\boldsymbol{L}\boldsymbol{y}\right) = -\frac{1}{2}\boldsymbol{\beta}_I^T\boldsymbol{\beta}_I \tag{8}$$

in the sparse sense as pointed out in [12], i.e. $\boldsymbol{\alpha}R = \boldsymbol{0}$. The decrease in the sparse posterior likelihood derived from Eq. (7) defines the selection criterion of Smola and Bartlett (SB), i.e.

$${}_{\mathrm{SB}}\Delta_i = \tau_I - \tau_{I'} = \frac{1}{2}\beta_{I',i}^2, \tag{9}$$

for each remaining point and with the new component $\beta_{I',i}$ of the updated vector $\boldsymbol{\beta}_{I'} \in \mathbb{R}^{m+1}$. Due to the high computational cost of $O(\mathrm{nm})$ for the criterion calculation per remaining point, the criterion is only evaluated for a randomly chosen subset of cardinality κ. The authors of [13] recommend $\kappa = 59$, which they justify with a probabilistic argument. Nevertheless, they end up with high computational cost of $O(\kappa \mathrm{nm}^2)$ for the whole DTC approximation. The conjugation of this selection heuristic defines the corresponding deletion criterion

$${}_{\mathrm{SB}}\nabla_i = {}_{\mathrm{SB}}\Delta_i \tag{10}$$

which leads to cost of $O(m^2)$ per active point.

To increase the performance of the former selection heuristic from Eq. (9), a matching pursuit approach (MPA) which reduces the computational effort but not the memory requirements is presented in [14]. Here, the authors fix $\boldsymbol{\alpha}_I \in \mathbb{R}^m$ in the minimization of $\tau(\boldsymbol{\alpha}_{I'})$ in Eq. (8) and only vary $\alpha_{I',i}$ with $i \in R$. This yields the insertion criterion

$${}_{\mathrm{MPA}}\Delta_I = \tau_I - \min_{\alpha_{I',i}}(\tau(\boldsymbol{\alpha}_{I'})) = \frac{\left(\boldsymbol{k}_{.,i}^T(\boldsymbol{y}-\boldsymbol{\mu}_I) - \sigma^2\mu_{I,i}\right)^2}{2\left(\sigma^2 k_{ii} + \boldsymbol{k}_{.,i}^T\boldsymbol{k}_{.,i}\right)} \tag{11}$$

with covariance vector $\boldsymbol{k}_{.,i} \in \mathbb{R}^n$. However, despite the lower computational cost of $\mathcal{O}(dn)$ per remaining point, on large data sets or under high input dimensions they also select a randomized subset of size κ for criteria evaluation to boost efficiency. An additional matrix cache that contains for example κ rows of the full covariance matrix $\boldsymbol{K}$ can help to speed up the criterion evaluations, but increases the memory requirements considerably.

In [12], a very fast greedy criterion with computational complexity of $\mathcal{O}(1)$ per remaining point is proposed. Here, the information gain (IG)

$${}_{\mathrm{IG}}\Delta_i = \mathrm{KL}[\tilde{Q}_{I'}(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})\,||\,Q_I(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})] \tag{12}$$

is measured by the increase in the KL-divergence between the current posterior distribution $Q_I(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})$ following from Eq. (6) and an approximated one $\tilde{Q}_{I'}(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X})$ after inclusion of the remaining point $\boldsymbol{x}_i$. Thereby, couplings between the latent function value f_i and the targets $\boldsymbol{y}_{\setminus i}$, i.e. without the i-th element, are ignored to guarantee low computational costs.

The intention of the following greedy selection criterion by Quiñonero–Candela (QC) is to increase the logarithmic marginal likelihood $\varphi_I(\boldsymbol{\theta})$ obtained from Eq. (4) by the inclusion of a remaining point. This leads to the equivalent criterion

$${}_{\mathrm{QC}}\Delta_i = \varphi_{I'}(\boldsymbol{\theta}) - \varphi_I(\boldsymbol{\theta}) = \frac{{}_{\mathrm{SB}}\Delta_i}{\sigma^2} - \log(l_{M,ii}) + \log(\sigma), \tag{13}$$

where only the change induced through the inclusion is considered, i.e. the hyperparameters $\boldsymbol{\theta}$ are fixed during the selection process, cf. [15]. More details for the adaption of hyperparameters are presented in Section 3.2. As pointed out in Eq. (13), this heuristic is closely related to the criterion in [13], since $\beta_{I',i}^2 \propto l_{M,ii}^{-2}$, where $l_{M,ii}$ is the i-th diagonal element of $\boldsymbol{L}_M$. Consequently, this criterion leads to the same computational cost of $\mathcal{O}(nm)$ per remaining point and is also calculated for only a small randomly selected remaining subset of size κ.

Table 1
This table summarizes the most important symbols for the presented sparse GPR approximation and the considered insertion and deletion strategies. A short description provides additional information about the used notations.

Symbol	Format	Description
$\boldsymbol{K}$	$n \times n$	Full covariance matrix
$\boldsymbol{K}_I$	$m \times m$	Covariance matrix induced by the active set of training points
$\boldsymbol{K}_{I,\cdot}$	$m \times n$	Covariance matrix induced by the active set points and all other training points
$\boldsymbol{P}_I$	$m \times n$	Projection matrix $\boldsymbol{P}_I = \boldsymbol{K}_I^{-1}\boldsymbol{K}_{I,\cdot}$ which induces the sparse approximation
$\boldsymbol{L}$	$m \times m$	Lower Cholesky factor of $\boldsymbol{K}_I$
$\boldsymbol{V}$	$m \times n$	Nyström factor for approximating the full covariance matrix $\boldsymbol{K}$
$\boldsymbol{M}$	$m \times m$	Short-hand notation for the matrix $\boldsymbol{M} = \sigma^2\boldsymbol{I} + \boldsymbol{V}\boldsymbol{V}^T$
$\boldsymbol{L}_M$	$m \times m$	Lower Cholesky factor of $\boldsymbol{M}$
$\boldsymbol{k}_{I,*}$	$m \times 1$	Covariance vector induced by the active set points and the test point $\boldsymbol{x}_*$
$\boldsymbol{k}_{I,i}$	$m \times 1$	Covariance vector induced by the active set points and the remaining point $\boldsymbol{x}_i$
$\boldsymbol{k}_{.,i}$	$n \times 1$	Covariance vector induced by all training points and the remaining point $\boldsymbol{x}_i$
$\boldsymbol{\alpha}_I$	$m \times 1$	Prediction vector $\boldsymbol{\alpha}_I = \boldsymbol{L}^{-T}\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I$
$\boldsymbol{\beta}_I$	$m \times 1$	Short-hand notation for the vector $\boldsymbol{\beta}_I = \boldsymbol{L}_M^{-1}\boldsymbol{V}\boldsymbol{y}$
$\boldsymbol{\mu}_I$	$n \times 1$	Estimated mean vector $\boldsymbol{\mu}_I = \boldsymbol{V}^T\boldsymbol{L}_M^{-T}\boldsymbol{\beta}_I$ of the sparse GPR model

The idea in [11] is the same as in [15], but a regularized logarithmic marginal likelihood given through a variational (VAR) framework is increased. This results in the criterion

$$_{\mathrm{VAR}}\Delta_i = {}_{\mathrm{QC}}\Delta_i + \frac{\| \boldsymbol{k}_{\cdot,i} - \boldsymbol{V}^T \boldsymbol{L}^{-1} \boldsymbol{k}_{I,i} \|^2}{2\sigma^2 \left(k_{ii} - \| \boldsymbol{L}^{-1} \boldsymbol{k}_{I,i} \|^2 \right)} \tag{14}$$

for active point selection. The relation to the criterion in [15] induces the same computational complexity of $\mathcal{O}(nm)$ per remaining point.To increase performance, the disadvantageous sub-sampling on a randomly chosen subset of remaining training points is again required.

The last insertion criterion which we discuss here is introduced in [17]. Csató (CS) defined his selection heuristic over the projection-induced error in the RKHS specified by the covariance function, see [3] for more details, which leads to

$$_{\mathrm{CS}}\Delta_i = k_{ii} - \| \boldsymbol{L}^{-1} \boldsymbol{k}_{I,i} \|^2 \tag{15}$$

with covariance vector $\boldsymbol{k}_{I,i} \in \mathbb{R}^m$. Due to its relatively low computational cost of $\mathcal{O}(m)$ per remaining point, it is possible to evaluate this criterion for all remaining points, which slightly increases the overall complexity of the DTC approximation to $O(\mathrm{nm}^3)$. Furthermore, in [16] Csató defined a greedy deletion criterion given by

$$_{\mathrm{CS}}\nabla_i = |\, {}_{\mathrm{CS}}\Delta_i \alpha_{I,i} \,| \,. \tag{16}$$

The main difference between the deletion and selection heuristic by Csató lies in the influence of the respective element of the prediction vector $\boldsymbol{\alpha}_I$. Since $\boldsymbol{\alpha}_I$ is known, we also need $O(m^2)$ arithmetic operations per active point, which is equally costly as the removal heuristic by Smola and Bartlett in Eq. (10). Note that the active point of the posterior model in Eq. (6) with minimal loss in terms of a deletion criterion is always removed.

3. Maximum error criterion for sparse Gaussian process regression

All the insertion and deletion methods discussed above either lack computational speed, have high memory requirements, or lack modeling accuracy. Moreover, if the regression model is generated based on a purely randomized selection or on a method with a small randomly selected remaining subset for criteria evaluation, e.g. as done in [13–15], the performance in hard regression tasks deteriorates. Our novel approach aims to provide a favorable compromise between modeling accuracy, computational cost, and memory requirements.

3.1. Maximum error criterion for fast greedy insertion and deletion of training points

In this section, we first discuss the successive inclusion of training points into the active subset. To include a remaining point $\boldsymbol{x}_i$ with $i \in R$ in the active subset, we have to update the Cholesky factors $\boldsymbol{L}$, $\boldsymbol{LM}$ and the matrix $\boldsymbol{V}$, respectively $\boldsymbol{K}_{I,\cdot}$, the vector $\boldsymbol{\beta}_I$, and the mean $\boldsymbol{\mu}_I$ of the posterior distribution given in Eq. (6), as shown in [12]. Thus, the cost for the sequential insertion in the m-th iteration is $O(\mathrm{nm})$.

Similar to the method of Smola and Bartlett in [13], our approach maximizes the posterior probability given by Eqs. (7) and (8). The greedy scheme in Eq. (9) successively maximizes the Euclidean norm of the vector $\boldsymbol{\beta}_{I'}$. This task is equivalent to iteratively minimizing $\| \boldsymbol{y} - \boldsymbol{\mu}_{I'} \|$ for the normalized vector $\boldsymbol{y}$ and, thus, approximately normalized $\boldsymbol{\mu}_{I'}$, since we have $\| \boldsymbol{\beta}_{I'} \|^2 = \boldsymbol{\beta}_{I'}^T \boldsymbol{\beta}_{I'} = \boldsymbol{y}^T \boldsymbol{\mu} I'$ after an inclusion. Due to the equivalence of norms in finite dimensional spaces, it holds true that $\| \boldsymbol{y} - \boldsymbol{\mu}_{I'} \| \leq n \max_{\forall j} | y_j - \mu_{I',j} |$. In the limit, i.e. with increasing m, we will approximately have $\boldsymbol{\mu}_{I'} \approx \boldsymbol{\mu}_I$. Since this convergence assumption holds only true for large m, we yield not necessarily to an upper bound for the model error after an inclusion. Nevertheless, we define

$$_{\mathrm{ME}}\Delta_i = | y_i - \mu_{I,i} | \tag{17}$$

as our new insertion criterion and select the remaining point that has the maximal error (ME) under the current posterior model in Eq. (6). This computationally efficient approach has $\mathcal{O}(1)$ cost for criterion calculation per remaining point. The convergence assumption obviates the update of the posterior model for each remaining point as needed for other selection criteria, e.g. in [11,13,15].

In the following, we present our maximum error deletion criterion for the removal of active points. Typically, the maximum number of active points m is predefined, since it influences computing time quadratically and memory requirements linearly. If a stopping criterion for m is used, for example, by monitoring the averaged square training error as in [12], deletion of appropriate active points improves the predictive performance without significantly deteriorating the existing model quality. The deletion also provides a way to reduce redundancy in the greedily selected active subset. Similar to the presented insertion strategy, we opt for a greedy criterion to successively delete active points. Note that deleting an active point does not necessarily lead to a state that was previously encountered when iteratively inserting training points. The reason is that the underlying assumptions for greedy insertion and deletion differ considerably. While for the inclusion strategies Cholesky updates are sufficiently fast and stable, QR-downdates based on the factorization $\boldsymbol{QR} = \boldsymbol{LML}^T$ are used for deleting active points since they offer higher numerical stability. This advantageous behavior is discussed in [19]. For our proposed technique, the cost for deletion of one point is equal to its insertion cost, i.e. $O(\mathrm{nm})$ in the m-th iteration of the DTC approximation. Inspired by the criterion in [17], we define our new deletion criterion as follows. Beginning with an already selected subset determined by I, we remove the active point with minimal value regarding the deletion criterion

$$_{\mathrm{ME}}\nabla_i = |\, {}_{\mathrm{ME}}\Delta_i \alpha_{I,i} \,| \,, \tag{18}$$

where $\boldsymbol{\alpha}_I = \boldsymbol{R}^{-1} \boldsymbol{Q}^T \boldsymbol{K}_{I,\cdot} \boldsymbol{y} \in \mathbb{R}^m$. Note that we use the maximum error ${}_{\mathrm{ME}}\Delta_i$ instead of the expensive projection-induced error ${}_{\mathrm{CS}}\Delta_I$. Thus, we obtain the same low complexity for a deletion criterion evaluation of $\mathcal{O}(1)$ per active point. Here, we coupled the error of an active training point in the current sparse model given by Eq. (6) with its importance under prediction in relation to the behavior in Eq. (7). So, our deletion criterion controls the current model accuracy and the generalization capability.

Finally, for a better comparison the complexity of each presented insertion criterion is presented in Table 2. Thereby, the row regarding to the memory effort describes only the storage amount which is needed for the criterion calculation for only one remaining training point. Since the FITC approximation depends not on a selection criterion, the respective entries in the presented table are empty.

3.2. Generalized expectation maximization for learning hyperparameters

The presented sparse GPR model depends not only on the active points $\boldsymbol{x}_i$ with $i \in I$, but also on the hyperparameters of the specified covariance function and the variance σ^2 of the Gaussian noise model. Let the vector $\boldsymbol{\theta}$ denote the collection of all hyperparameters. For the sake of notational simplicity the dependency of the above formulas on $\boldsymbol{\theta}$ was neglected. The adaptation of the hyperparameters can be realized by gradient based optimization

Table 2
This table summarizes the computational complexity and storage requirements of the discussed sparse GPR approximation methods. The memory row describes only the additional needed storage amount for criterion calculation based on the already determined predictive model (6). Nevertheless, the practical memory effort and computational performance of each method depends very strongly on their implementation, especially for the last four intelligent selection schemes.

Complexity	FITC	DTC insertion criterion							
		ME	IG	CS	MPA	SB	QC	VAR	Random
Criterion	–	$O(1)$	$O(1)$	$O(m)$	$O(dn)$	$O(nm)$	$O(nm)$	$O(nm)$	$O(1)$
Complete	$O(nm^2)$	$O(nm^2)$	$O(nm^2)$	$O(nm^3)$	$O(nm^2)$	$O(\kappa nm^2)$	$O(\kappa nm^2)$	$O(\kappa nm^2)$	$O(nm^2)$
Memory	–	$O(1)$	$O(1)$	$O(m)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(1)$

algorithms that maximize the logarithmic marginal likelihood

$$\varphi_I(\boldsymbol{\theta}) = \log\left(Q_I(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})\right) = -(n-m)\log(\sigma) - \sum_{i=1}^{m}\log(l_{M,ii}) - \frac{1}{2\sigma^2}\left(\boldsymbol{y}^T\boldsymbol{y} - \boldsymbol{\beta}_I^T\beta_I\right) - \frac{n}{2}\log(2\pi) \tag{19}$$

obtained from Eq. (4). The values $l_{M,ii}$ are the diagonal entries of the Cholesky factor $\boldsymbol{L}_M$. In [11], the author uses a regularized version of the above logarithmic marginal likelihood, i.e.

$${}_{\mathrm{VAR}}\varphi_{\mathrm{I}}(\boldsymbol{\theta}) = {}_{\varphi\mathrm{I}}(\boldsymbol{\theta}) - \frac{1}{2\sigma^2}\operatorname{trace}\left(\boldsymbol{K} - \boldsymbol{V}^T\boldsymbol{V}\right), \tag{20}$$

described through his variational (VAR) approach. This additional regularization term leads to correcting the Nyström approximation of the full covariance matrix in the hyperparameter learning process and gives a lower bound to the true logarithmic marginal likelihood in Eq. (19) of the DTC approximation, cf. [11]. One problem encountered when maximizing ${}_{\varphi\mathrm{I}}(\boldsymbol{\theta})$ or ${}_{\varphi\mathrm{I}}(\boldsymbol{\theta})$ in the variational framework, respectively, is their dependence on the active subset of training points determined by I. To solve this problem, we take alternating constrained optimization steps in an expectation maximization manner employing the theory of [20]. Then, the expectation step for estimating the new posterior distribution $Q_{I_{\mathrm{new}}}(\boldsymbol{f}_{I_{\mathrm{new}}}|\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta})$ from Eq. (3) with fixed hyperparameters $\boldsymbol{\theta}$ is given by

$$Q_{I_{\mathrm{new}}}(\boldsymbol{f}_{I_{\mathrm{new}}}|\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta}) = \underset{Q_I(\boldsymbol{f}_I|\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta}) \in Q_I(\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta})}{\operatorname{argmin}} \mathrm{KL}\left[Q_I(\boldsymbol{f}_I|\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta}) \,\|\, Q_I(\boldsymbol{y},\boldsymbol{f}_I|\boldsymbol{X},\boldsymbol{\theta})\right]. \tag{21}$$

Here, the posterior distribution in the expectation step in Eq. 21 regarding the KL-divergence is conditioned on the probability distribution $Q_I(\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta})$. That is the approximated posterior, which is induced by an active subset of size m. This condition is handled with a fixed final size m of the active subset in the greedy selection process. Furthermore, the maximization step results in

$$\boldsymbol{\theta}_{\mathrm{new}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \mathrm{E}_{\boldsymbol{f}_I|\boldsymbol{y},\boldsymbol{X},\boldsymbol{\theta}}\left[\log\left(Q_I(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})\right)\right] \tag{22}$$

to determine an updated set of hyperparameters $\boldsymbol{\theta}_{\mathrm{new}}$. The maximization step in Eq. (22) is realized with only few gradient ascents on the logarithmic marginal likelihoods from Eq. (19) or (20) with a fixed active point set $\boldsymbol{X}_I$, respectively. In this case, the repeated alternating computation of the E- and M-steps leads to a generalized EM algorithm, since we only increase the logarithmic marginal likelihood. Since the generalized EM algorithm converges to local maxima, the choice of the active training points is important in order to obtain a good set of hyperparameters $\boldsymbol{\theta}$. For the selection of the active subset we employ our efficient maximum error criterion to keep the hyperparameter learning fast and stable.

4. Evaluations

In this section, we compare our maximum error insertion and deletion criteria with other methods for the DTC approximation presented in Section 2. Furthermore, we also consider the FITC approximation given in [10] to present an extensive comparison. For all experiments, we use the stationary squared exponential covariance function

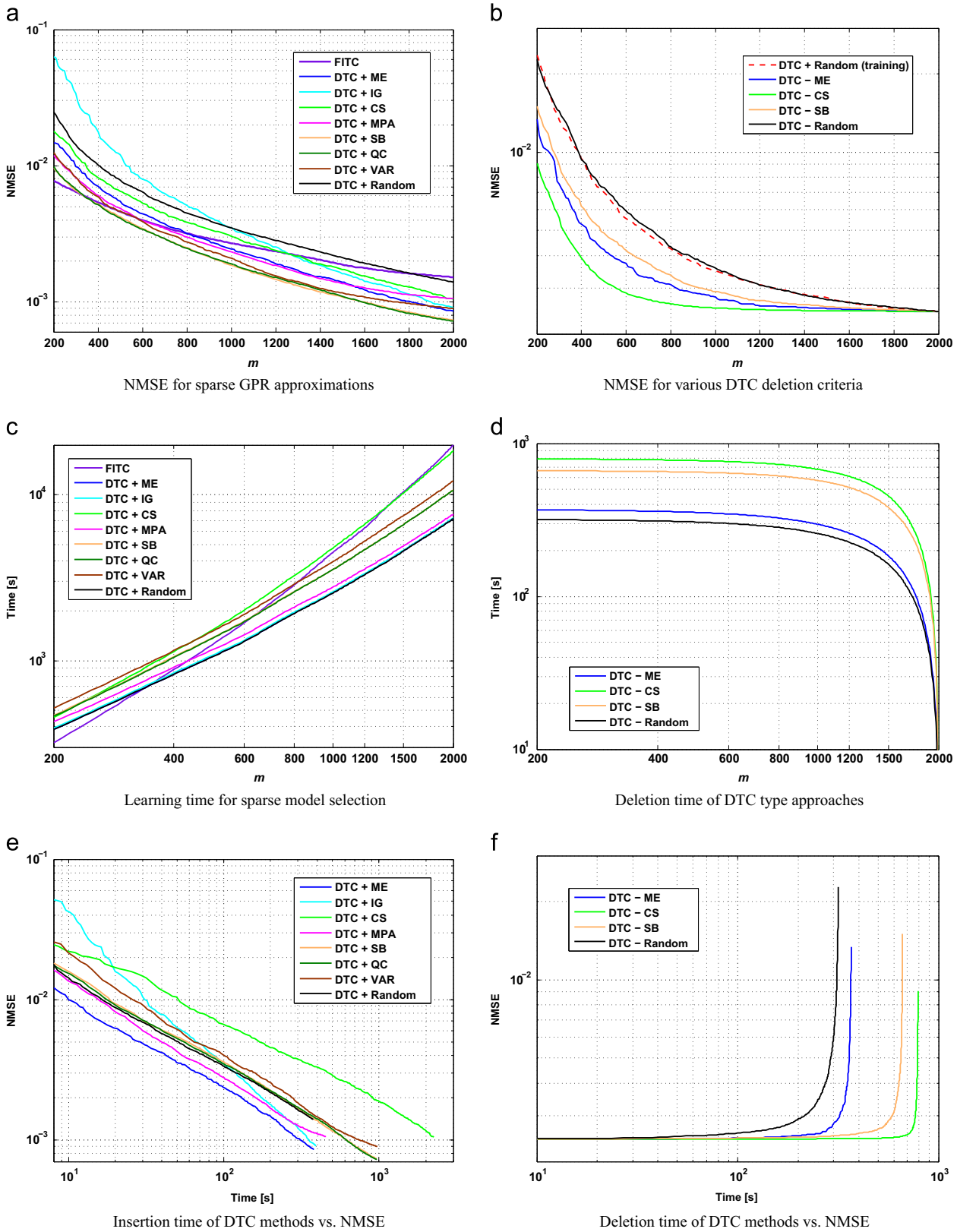
$$k(\boldsymbol{x}_i,\boldsymbol{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T\boldsymbol{\Lambda}^{-2}(\boldsymbol{x}_i - \boldsymbol{x}_j)\right) \tag{23}$$

with magnitude σ_f^2 and automatic relevance determination, i.e. with one lengthscale parameter λ_l per input dimension l in the diagonal matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{d\times d}$, see [2]. The accuracy of the methods under consideration is measured by the normalized mean square error (NMSE), cf. [4].

4.1. Comparisons on learning inverse dynamics models

Here, a real benchmark data set from the 7 Degree of Freedom (DoF) SARCOS master arm (13922 training and 5569 test points), and a simulation data set from the SARCOS model (14904 training and 5520 test points) are used for validation, see [4,21]. Each point of the data sets has 21 input dimensions, i.e. position, velocity and acceleration in joint space, and 7 targets, i.e. one torque for each joint of SARCOS robot arm, see Fig. 3(c) . The goal is to learn the inverse dynamics models, i.e. the mapping from position, velocity and acceleration to corresponding torque for each joint. Both robot data sets contain independent training and test points for all DoF's.

The convergence trends with respect to the NMSE for all discussed sparse GP approximations on the first DoF from the real SARCOS test data are shown in Fig. 2(a) and (b). Here, for all DTC type approaches, the hyperparameters were learned with ten EM steps and randomized active point selection up to a final set size of $m=2000$. For a fair comparison, we adapt the number of gradient steps in the FITC approximation linearly with increasing virtual training points, i.e. we use $150+\frac{m}{4}$ optimization steps, as the number of hyperparameters in the FITC approximation also grows linearly with m. The NMSE results for randomized selection in the DTC approximation are averaged over ten runs. The learning times and training times for the different sparsifications are further shown in Fig. 2(c) and (d), respectively. As shown by the results, the costs for insertion are similar for Smola and Bartlett [13] and Quiñonero–Candela [15], as pointed out in Section 2.1. The variational framework in [11] leads to constantly higher effort in the learning process, e.g. compared to the curves in [13,15], since the regularization term increase the cost of gradient based optimization techniques. We always use a remaining set size of $\kappa = 59$ to speed up the criteria calculation of the methods above. Our maximum error approach outperforms all DTC selection criteria with respect to training times with low NMSE on test data, see Fig. 2(e). For large active set sizes, we nearly reach the same accuracy as the more costly selection heuristics in [13,15] and outperform the matching pursuit approach from [14], see Fig. 2(a). But our selection criterion performs also very well for small active set sizes, which empirically justify the adopted assumptions in the derivation of the maximum error (ME) strategy. As shown by the right column in Fig. 2, we outperform the DTC deletion criterion from Smola and Bartlett [13] and the randomized version in term of generalization accuracy. Our approach also yields the best



NMSE for sparse GPR approximations

NMSE for various DTC deletion criteria

Learning time for sparse model selection

Deletion time of DTC type approaches

Insertion time of DTC methods vs. NMSE

Deletion time of DTC methods vs. NMSE

Fig. 2. Learning results as NMSE and learning times for the first DoF. The evaluation is performed on the SARCOS test data for different sparse GPR approximations, i.e. by Smola and Barlett (SB) [13], Seeger (IG) [18], Titsias (VAR) [11], Keerthi and Chu (MPA) [14], Quiñonero-Candela (QC) [15], and Csató (CS) [16]. The right column shows performances of different deletion criteria of the DTC approximation. Our novel strategy (ME) gives the best trade-off between low computing times and accurate prediction. As shown in Fig. 2(e), our approach yields the lowest learning curve and, thus, provides the best trade-off between model accuracy and computation time.

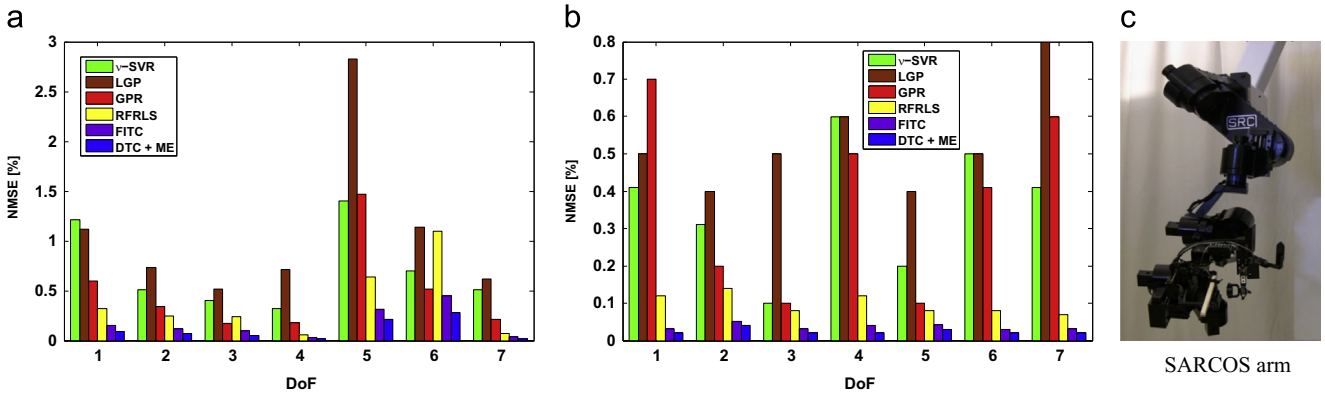


Fig. 3. Prediction errors as NMSE (in percent) for each degree of freedom (DoF) after prediction on the test sets with real SARCOS data 3(a) and simulated robot data 3 (b) from the SARCOS master arm 3(c). Overall, the DTC approximation with the novel and fast maximum error (ME) criterion performs best, closely followed by the FITC approximation.

compromise between low computational effort and high prediction precision, as shown in Fig. 2(f). In Fig. 2(b), DTC + Random (training) describes the convergence trend dependent on the randomized sparse GP model generation which builds the base for the subsequent evaluated deletion criteria. Thus, in Fig. 2 (d) presented deletion times depend on the current set size of $m=2000$ and it would be helpful to read all plots in the right column from right to left. Note that all deletion schemes yield better NMSE results than the simple, randomized deletion.

In Fig. 3, we compare our maximum error selection method for the sparse DTC approximation and the FITC approximation [10] with other established regression procedures on all DoF's for real and simulated SARCOS test data. The results for the other methods, i.e. local Gaussian processes (LGP), ν-SVR, GPR, and random Fourier regularized least squares (RFRLS), are taken from [4] and [22]. For comparison, we also employ a final set size of $m=2000$ active or virtual training points for both sparse GP approximations. We again use ten generalized EM steps for hyperparameter learning of the DTC approximation and 650 gradient ascents for optimization of virtual training points with subsequent prediction. The higher errors for the fifth and sixth DoF on the real SARCOS test data are due to more complex non-linearities in these DoF, e.g. induced by their joint inertia. Compared to these regression algorithms, our approach is efficient and one of the best performing methods. Compared to the FITC approximation, we have much less effort for sparse model selection with higher generalization accuracy.

4.2. Compliant and real-time tracking control

In this section, we employ learned inverse dynamics models for a real-time tracking control task [23] on a PR2 robot, as shown in Fig. 4. Here, the model-based tracking control law determines the joint torques $\boldsymbol{y}$ for each of the seven DoF's necessary for following a desired joint trajectory $\boldsymbol{x}_d$, $\dot{\boldsymbol{x}}_d$, $\ddot{\boldsymbol{x}}_d$, where $\boldsymbol{x}$, $\dot{\boldsymbol{x}}$, $\ddot{\boldsymbol{x}}$ are joint angles, velocities and accelerations of the robot, as shown in Fig. 5. This control paradigm uses a dynamics model, while employing feedback in order to stabilize the system. Here, the dynamics model of the robot can be used as a feed-forward model that predicts the joint torques $\boldsymbol{y}_{ff}$ required to perform the desired trajectory, while a feedback term $\boldsymbol{y}_{fb}$ ensures the stability of the tracking control with a resulting control law of $\boldsymbol{y}=\boldsymbol{y}_{ff}+\boldsymbol{y}_{fb}$. The feedback term can be a linear control law such as $\boldsymbol{y}_{fb}=\boldsymbol{G}_P\boldsymbol{e}+\boldsymbol{G}_D\dot{\boldsymbol{e}}$, where $\boldsymbol{e}=\boldsymbol{x}_d-\boldsymbol{x}$ denotes the tracking error and $\boldsymbol{G}_P$ as well as $\boldsymbol{G}_D$ position-gain and velocity-gain, respectively. If an accurate inverse dynamics model can be obtained, the feed-forward term $\boldsymbol{y}_{ff}$ will largely cancel the robot's non-linearities [24]. In this case, the gains $\boldsymbol{G}_P$ and $\boldsymbol{G}_D$ can be chosen to have small values enabling compliant control performance [25].

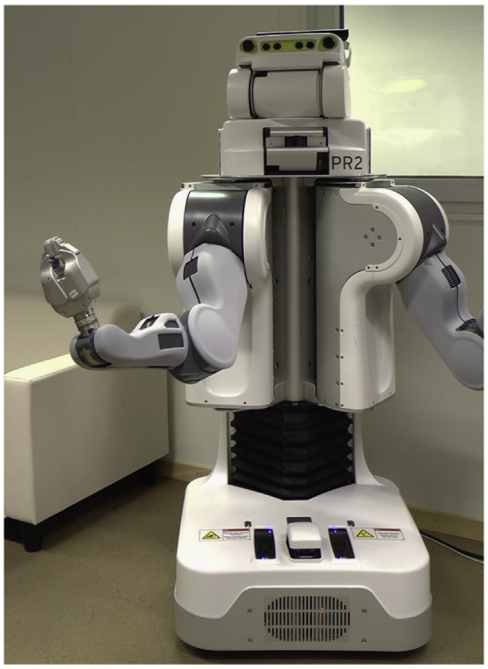


Fig. 4. The PR2 robot.

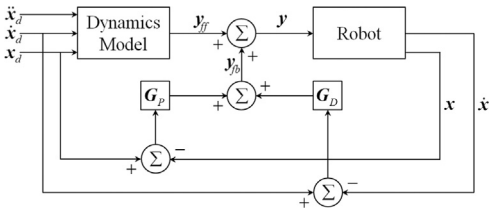
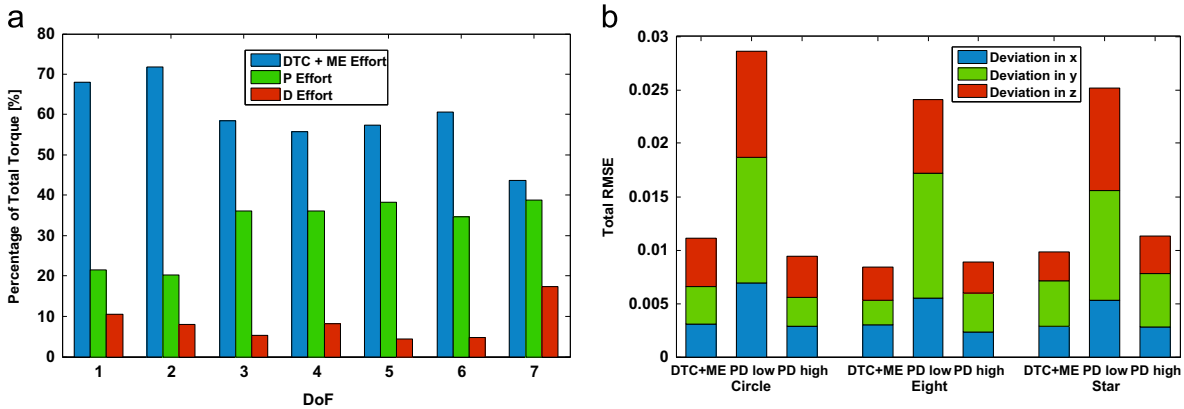


Fig. 5. Feed-forward control with inverse dynamics models.

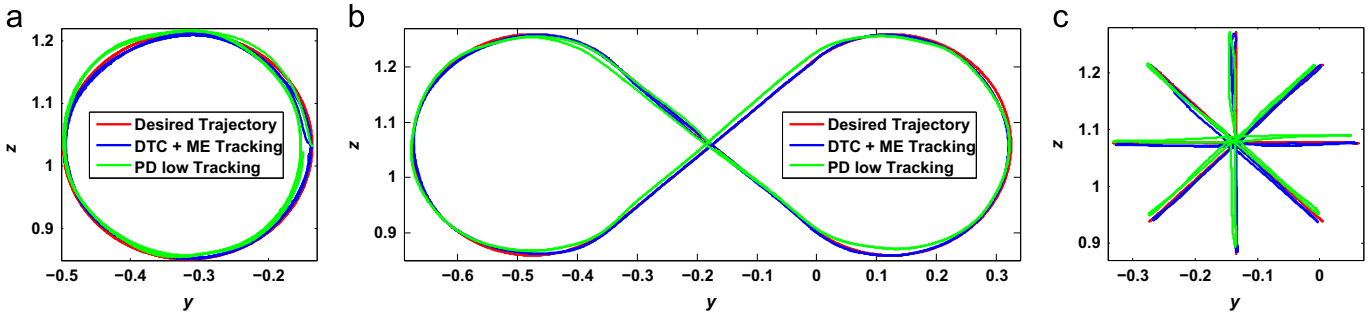
To obtain a global and precise dynamics model, we sample 517.783 data points with a frequency of 100 Hz from the right arm of the PR2 robot. Furthermore, we train a sparse GP model given through the DTC approximation with our maximum error (ME) criterion for each of the seven DoF's. Thereby, the hyperparameters are always learned with 10 generalized EM steps, as explained in Section 3.2. We choose a final active set size of $m=1000$, which is sufficient to reach a good model quality while



Torque percentage for each DoF

Tracking error on three test trajectories

Fig. 6. The Torque percentage for each DoF of the right PR2 arm is shown in Fig. 6(a). The higher the torque percentage, the more contributions have the corresponding parts. Here, DTC + ME dynamics models usually have far over 50% torque contribution which enables compliant control. Plot 6(b) shows the tracking errors in task-space (x, y, z) for the three test trajectories, i.e. circle-, eight- and star-shape. The RMSE value of each dimension is computed for 3 different control schemes, i.e. low-gain DTC + ME model-based control, PD-control with low gains, and PD-control with high gains, i.e. about four times higher gains as in the low gain control schemes.



Circle-shapes

Eight-shapes

Star-shape

Fig. 7. Tracking performance on the three test trajectories using low-gain DTC + ME model-based controller and PD-controller with low gains.

yielding prediction times of less than 3 ms for all seven DoF's. Hereby, we are able to perform tracking control in real-time at 100 Hz. In Fig. 6(a), the percentage on total torque for each DoF of the PR2 robot arm is shown, where the gains for feedback control are chosen very small in order to enable compliant control. The contribution of our sparse GPR model, i.e. approximated with DTC + ME, to the control effort is usually far over 50%. Here, a high contribution to the control effort indicates a well approximated model, since the feedback control loop does not need to strongly intervene during the control task. The corresponding tracking performance in task-space is presented in Fig. 7 for three test trajectories, e.g. circle-, eight- and star-shape. Here, we compare the low gain feed-forward control using the learned dynamics model with the standard PD-control scheme. The results with respect to the root mean square error (RMSE) are shown in Fig. 6 (b). It can be seen that, applying learned dynamics models, we obtain compliant tracking control, and, at the same time, have tracking accuracy comparable to the high gain PD-control scheme.

5. Conclusion

In this paper, we proposed a very fast greedy insertion and deletion scheme for sparse GPR or, more precisely, for the DTC approximation. Our criterion is based on the maximum error between model and training data, inspired by [13] and [16]. It leads to a stable and efficient way for automatic sparse model selection. The primary advantage of our maximum error greedy selection is the combination of high accuracy with low computational costs for criterion calculation for all remaining points. In contrast, the insertion methods in [11,13–15] have to select a small random subset of remaining points for criterion evaluation. This random restriction can lead to poorer results, especially on harder regression tasks. Even without caching, we are nearly as fast as a randomized insertion. For the removal of active points our approach almost reaches the accuracy of Csató's deletion method, outperforms the deletion criterion in [13], and is still nearly as fast as a randomized removal. Compared to the FITC approximation [10], all DTC methods lead to higher prediction accuracy and lower learning times. Further results on a real PR2 robot show that the proposed method can be employed for compliant, real-time robot control.

Acknowledgments

The authors would like to thank A. Gijsberts, Idiap Research Institute, for providing his results on the SARCOS data sets. We also like to thank S. Schaal, University of Southern California and Max-Planck-Institute for Intelligent Systems in Tübingen, for providing the image of the SARCOS master arm.

References

[1] J. Schreiter, D. Nguyen-Tuong, H. Markert, M. Hanselmann, M. Toussaint, Fast greedy insertion and deletion in sparse Gaussian process regression, in: M. Verleysen (Ed.), Proceedings of the 23rd European Symposium on Artificial Neural Networks, 2015, pp. 101–106.

[2] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, The MIT Press, Cambridge, Massachusetts, 2006.

[3] B. Schölkopf, A.J. Smola, Learning with Kernels-Support Vector Machines, Regularization, Optimization, and Beyond, The MIT Press, Cambridge, Massachusetts, 2002.
[4] D. Nguyen-Tuong, J. Peters, M. Seeger, Local Gaussian process regression for real time online model learning and control, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems, 21, 2009, pp. 1193–1200.
[5] A.C. Damianou, N.D. Lawrence, Deep Gaussian processes, in: C.M. Carvalho, P. Ravikumar (Eds.), Proceedings of the 16th International Conference on Artificial Intelligence and Statistics, JMLR: W&CP, vol. 31, 2013, pp. 207–215.
[6] C.K.I. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, 2001, pp. 682–688.
[7] E.G. Băzăvan, F. Li, C. Sminchisescu, Fourier kernel learning, in: A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, C. Schmid (Eds.), Proceedings of the 12th European Conference on Computer Vision, 2012, pp. 459–473.
[8] M. Lázaro-Gredilla, J. Quiñonero-Candela, C.E. Rasmussen, A.R. Figueiras-Vidal, Sparse spectrum Gaussian process regression, in: T. Jaakkola (Ed.), J. Mach. Learn. Res. 11 (2010) 1865–1881.
[9] J. Quiñonero-Candela, C.E. Rasmussen, A Unifying view of sparse approximate Gaussian process regression, in: R. Herbrich (Ed.), J. Mach. Learn. Res. 6 (2005) 1939–1959.
[10] E.L. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems, vol. 18, 2006, pp. 1257–1264.
[11] M.K. Titsias, Variational learning of inducing variables in sparse Gaussian processes, in: D. van Dyk, M. Welling (Eds.), Proceedings of the 12th International Conference on Artificial Intelligence and Statistics, JMLR: W&CP, vol. 5, 2009, pp. 567–574.
[12] M. Seeger, C.K.I. Williams, N.D. Lawrence, Fast forward selection to speed up sparse Gaussian process regression, in: C.M. Bishop, B.J. Frey (Eds.), Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, 2003, pp. 205–212.
[13] A.J. Smola, P.L. Bartlett, Sparse greedy Gaussian process regression, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, 2001, pp. 619–625.
[14] S.S. Keerthi, W. Chu, A matching pursuit approach to sparse Gaussian process regression, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems, vol. 18, 2006, pp. 643–650.
[15] J. Quiñonero-Candela, Learning with uncertainty – Gaussian processes and relevance vector machines (Ph.D. thesis), Technical University of Denmark, 2004.
[16] L. Csató, M. Opper, Sparse representation for Gaussian process regression, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, 2001, pp. 444–450.
[17] L. Csató, Gaussian processes–iterative sparse approximations (Ph.D. thesis), Aston University Birmingham, 2002.
[18] M. Seeger, Bayesian Gaussian process models–PAC-Bayesian generalisation error bounds and sparse approximations (Ph.D. thesis), University of Edinburgh, 2003.
[19] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, M.J. Way, P. Gazis, A. Srivastava, Stable and efficient Gaussian process calculations, in: C.K.I. Williams (Ed.), J. Mach. Learn. Res., vol. 10, 2009, pp. 857–882.
[20] J.V. Graça, K. Ganchev, B. Taskar, Expectation maximization and posterior constraints, in: J. Platt, D. Koller, Y. Singer, S.T. Roweis (Eds.), Advances in Neural Information Processing Systems, vol. 20, 2008, pp. 569–576.
[21] D. Nguyen-Tuong, J. Peters, Incremental sparsification for real-time online model learning, in: A. Bouchachia, N. Nedjah, C.-S. Leung (Eds.), Neurocomputing, vol. 74, 2011, pp. 1859–1867.
[22] A. Gijsberts, Incremental learning for robotics with constant update complexity (Ph.D. thesis), University of Genoa, 2011.
[23] J. Schreiter, P. Englert, D. Nguyen-Tuong, M. Toussaint, Sparse Gaussian Process Regression for Compliant, Real-Time Robot Control, in: A. Okamura, S. Hutchinson (Eds.), Proceedings of the IEEE International Conference on Robotics and Automation, 2015, pp. 2586–2591.
[24] M.W. Spong, S. Hutchinson, M. Vidyasagar, Robot Dynamics and Control, vol. 2, John Wiley & Sons, Hoboken, New Jersey, 2004.
[25] D. Nguyen-Tuong, J. Peters, M. Seeger, Computed torque control with nonparametric regression models, in: Proceedings of the American Control Conference, 2008, pp. 212–217.

Jens Schreiter received his Bachelor and Master degrees in mathematics from the University of Applied Sciences in Mittweida, Germany, in 2010 and 2012, respectively. Since 2012, he is working on his Doctoral degree at the Bosch Corporate Research facility in Renningen, located in the west of Stuttgart, Germany. The Machine Learning and Robotics Laboratory at the University of Stuttgart, Germany, supports his industrial Ph.D. studentship. His research interests include Gaussian process modeling techniques for transient systems and active learning methods for optimal design of experiments, both with automotive applications.

Duy Nguyen-Tuong has been with Bosch Corporate Research in Renningen since 2011, where he works on machine learning with focus on robotics and automotive applications. Before joining Bosch Research, he studied control and automation engineering at the University of Stuttgart and the National University of Singapore. From 2007 to 2011, he was a member of the Robot Learning Laboratory at the Max Planck Institute for Biological Cybernetics, where he completed his Ph. D. studies. His main research interest is the application of machine learning techniques for model learning. Recently, he also becomes interested in control and policy learning, which is useful for numerous automotive and industrial applications.

Marc Toussaint is a full Professor for Machine Learning and Robotics at the University of Stuttgart since 2012. Before that, he was an Assistant Professor and leading an Emmy Noether research group at FU & TU Berlin. His research focuses on the combination of decision theory and machine learning, motivated by fundamental research questions in robotics. Specific interests are combining geometry, logic and probabilities in learning and reasoning, and appropriate representations and priors for real-world robot learning. He is a Coordinator of the German research priority program *Autonomous Learning*, Reviewer for the German Research Foundation, and Program Committee Member of several top conferences (UAI, RSS, ICRA, AISTATS, ICML).