Incremental learning of EMG-based control commands using Gaussian Processes

Felix Schiel[1] **Annette Hagengruber**[1] **Jörn Vogel**[1] **Rudolph Triebel**[1,2]

[1]Institute of Robotics and Mechatronics
German Aerospace Center (DLR)
Germany
Firstname.Lastname(at)dlr.de

[2] Department of Computer Science
Technical University of Munich (TUM)
Germany

Abstract: Myoelectric control is the process of controlling a prosthesis or an assistive robot by using electrical signals of the muscles. Pattern recognition in myoelectric control is a challenging field, since the underlying distribution of the signal is likely to change during the application. Covariate shifts, including changes of the arm position or different levels of muscular activation, often lead to significant instability of the control signal. This work tries to overcome these challenges by enhancing a myoelectric human machine interface through the use of the sparse Gaussian Process (sGP) approximation Variational Free Energy and by the introduction of a novel adaptive model based on an unsupervised incremental learning approach. The novel adaptive model integrates an interclass and intraclass distance to improve prediction stability under challenging conditions. Furthermore, it demonstrates the successful incorporation of incremental updates which is shown to lead to a significantly increased performance and higher stability of the predictions in an online user study.

Keywords: Sparse GP Regression, Incremental Learning, myoelectric Human Machine Interface

1 Introduction

Electromyography (EMG) gives the opportunity to realize interfaces based on muscular activity to command assistive devices in health care applications. Common applications following this approach are myoelectric prostheses, where the activity of residual muscles in the stump is used to control a mechatronic limb. EMG can also be used to control a robotic arm, in case, the user has no remaining functional movements of his own arms, but is still able to evoke muscular activity [1]. However, the usage of a myoelectric interfaces, e.g. when using a prosthesis, often leads to poor performance due to altered input data [2]. Emerging problems are instantaneous or slow changes of the recorded muscle signals over time. In EMG, instantaneous changes can be caused by electrode shift or lift, different contraction forces or variations of the position of the limb, all of which result in the so called *covariate shift* [3]. Slow changes, also called *concept drift* [4], can arise from external interference, changes of the impedance of the electrodes or muscle fatigue [2, 5].

The mapping from the EMG signals to the control output can be realized by various predictive models. Common machine learning models assume that the training data is drawn from the same distribution as the testing data (data should be identically distributed (i.d.)). However, caused by the previously named challenges, this assumption may not always be valid. Therefore, to regain a satisfying performance, the model has to be adapted as soon as the performance is not satisfying anymore.

In this paper a novel adaptive model based on an unsupervised incremental learning approach is introduced that provides improved performance under covariate shift by the use of the sparse Gaussian Process (sGP) approximation in combination with an incremental learning approach. The method shows an applied solution to the mentioned problem and is to our knowledge the first work on applying incremental learning with sparse GPs to EMG-based control. A pilot user study with six subjects was conducted to verify the developed concept.

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

2 Related work

In order to increase reliability and stability of predictive models to changes in the input space, two main strategies have emerged in research. One is creating models which are able to generalize. This is frequently made possible through the use of training data sets which cover the occurring variability. The other concept is the adaptation of the model to unexpected changes in the data during prediction. Adaptations can be done before usage (offline) or during usage (online). Online adaptive models have the advantage to be very flexible and to be able to compensate for concept drift and covariate shift. One field in which adaptive models have been explored are intracortical BCIs, where they are used to compensate for non-stationarities in the recorded neural signals [6, 7, 8]. Furthermore, various adaptive approaches have also been applied in EMG-based control of assistive devices.

Previous work in EMG — One approach that tries to find models that generalize better in EMG applications is *transfer learning*. The underlying assumption is that there exist common structures in the data, e.g. between usage on different days. Transfer learning tries to find these common structures and defines adjustments to transfer the newly arriving data (domain adaptation) [9, 10] or the model parameters (rule adaptation) [11], respectively.

An offline adaptation approach for EMG-data is given by Zhai et al. [12], where a convolutional neural network was retrained with unsupervised corrected prediction results of previous sessions. Higher performance could be achieved in upcoming sessions. Vidovic et al. [3] reached comparable results of an increasing day-to-day stability of the prediction. Instead of using previous model predictions, they used short supervised retraining to update the parameters of an LDA classifier. It could be shown, that the short training sessions can maintain the quality of the predictions over multiple days.

An evaluation of an unsupervised and supervised online adaptation has been performed by Sensinger et al. [13]. In this study, the entropy of the prediction of an LDA classifier was used as a query function, which commonly describes the uncertainty of the prediction. For unsupervised updates, class labels of closest neighbors as well as smoothness of the predictions (no rapid changes between classes) were used to guess the unknown label. Applying supervised updates always achieved higher performance, whereas unsupervised updates showed no significant effect. Huang et al. [14] recently proposed a fast supervised as well as an unsupervised adaptive learning method based on a combination of SVM and a particle classifier. The particle classifier was used as a query function to rate new incoming samples. Incremental supervised updates have also been applied by Gijsberts et al. [15], where predicted finger forces based on *Ridge Regression* were used in combination with the *Sherman-Morrison Formula* to perform fast incremental updates. Both studies confirmed that incremental updates can increase the performance of a model in an adaptive learning application. Other successful examples of adaptive learning approaches are given by [16, 17].

Adaptive learning and sparse GP — GPs have the convenient property that an uncertainty estimation is part of the inference and that they give more reliable uncertainty measures than other methods for data discrimination (see, e.g. [18]). This makes the model very eligible for adaptive learning [19]. Additionally, the property of being a non-parametric model allows GPs to adapt the model complexity with respect to the complexity of the seen data, whereas parametric models are independent of the data after the training procedure and therefore bounded in their complexity. The main limitation of GPs is that the time for retraining the model scales with $O(n^3)$ where n is the number of data points, which makes online retraining, sample by sample, infeasible for larger data sets. Various approaches counteract this limitation to reduce the time complexity for online updates [20, 21].

Kapoor et al. [19] used a supervised active learning approach for object categorization with a GP. A query function chose unlabeled data points based on the posterior variance (uncertainty estimation) as well as on the posterior mean of the GP. The approach showed that the properties of GPs guide to the data points which could increase performance for future predictions. This result got confirmed by another study which applied a supervised active learning approach using a sparse GP method called *Informative Vector Machine* for image segmentation [22]. Additionally, they could show that the time complexity to update the sparse GP could be significantly reduced, which strengthens the applicability of GPs in adaptive learning scenarios.

3 Converting EMG signals to control commands

The major concept that underlies our proposed robot control system is to record and interpret the muscular activity of a human operator and convert it into control commands for a robot. To achieve

this, we use sEMG sensors that are attached to the skin of the dominant arm of the user. The signals recorded from the sensors are then digitized at a sampling rate of 1kHz and preprocessed to generate four classical time-domain features [23] over a window size of 150ms: *sEMG-amplitude*, *Slope Sign Change* (SSC), *Zero Crossing Rate* (ZCR), and *Wave Length* (WL). To be able to clearly distinguish an intended muscular activity from the nominal case where no activity is assumed, a calibration step is performed. This is done by recording measurements while the user is in a resting state to find a DC-offset to be subtracted from the signal and by determining a threshold for the SSC and ZCR features, such that muscular activity can be detected based on their values. The resulting calibrated signal is then represented as a time-dependent feature vector $\mathbf{x} \in \mathcal{X}$. The purpose of this work is to map these feature vectors onto control vectors $\mathbf{v}$, representing the velocities of the robot's end effector. In this work $\mathbf{v}$ is a 2-dimensional velocity vector corresponding to planar robot motions, but we note that our formulation can equally be extended also to 3-dimensional motions.

3.1 Non-parametric representation

As already sketched in the introduction, dealing with EMG signals comes with a number of challenges. First, the input data is often corrupted by many outliers and sensor noise. Second, the data distribution can change drastically during operation, mainly due to above mentioned effects such as electrode shift, change of impedance or muscle fatigue. And third, data sets of hand-annotated training data are usually not available compared to, e.g., visual data, where often large ground-truth benchmark data sets exist. To address these challenges, we propose to use a probabilistic, non-parametric data representation based on a sparse GP regression scheme. This has the advantage that it provides a solid probabilistic formulation, it doesn't restrict the formulation to a specific functional classifier model, and it can be trained effectively even with very little training data (as opposed to most deep learning approaches). Our approach is based on our previous work on EMG-based control [24], but we extend that model so that it adapts itself to new incoming data samples online.

Concretely, we formulate the approach as a supervised learning problem, where a training data set $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^n$ is given, consisting of n training samples of features $\mathbf{x}_i$ and ground-truth velocities $\mathbf{v}_i$. We apply a GP regression, where for each output dimension d we use a separate GP trained on $\mathcal{D}_d = \{(\mathbf{x}_i, v_{d,i})\}_{i=1}^n$. We restrict the ground truth velocities so that $v_{d,i} \in \{-1, 0, 1\}$, while predicted velocities v_d^* can be non-integers. In the GP formulation (see Rasmussen and Williams [25] for more details), latent random variables $\mathbf{f}$ are introduced as function values of the data $\mathbf{x}_i$, and it is assumed that any finite collection of latents $\mathbf{f}_i$ is jointly Gaussian distributed. The similarity of two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ is measured using a symmetric covariance function or *kernel* $k(\mathbf{x}_i, \mathbf{x}_j)$. For the regression case, a Gaussian likelihood $p(\mathbf{v} \mid \mathbf{f})$ is used to model sensor noise, so that a latent posterior $p(\mathbf{f} \mid X, \mathbf{v})$ can be computed in closed form from the training data $(X, \mathbf{v})$. The normalizer of that posterior, namely $p(\mathbf{v} \mid X)$ (a.k.a. the *marginal likelihood*), can be used to find optimal hyperparameters of the kernel k from the training data. To predict a value v^* for a new test sample $\mathbf{x}^*$, two marginalization steps w.r.t. the latents $\mathbf{f}$ and f^* are performed, where the first computes the latent posterior $p(f^* \mid \mathbf{v}, X, \mathbf{x}^*)$, and the second results in the *predictive distribution*

$$p(v^* \mid X, \mathbf{v}, \mathbf{x}^*) = \mathcal{N}(v^* \mid \mu^*, \sigma^*). \tag{1}$$

Both computations can be made in closed form, resulting in the predictive mean μ^* and the predictive variance σ^*.

The main limitation of GPs is their time complexity of $\mathcal{O}(n^3)$, as well as the memory complexity of $\mathcal{O}(n^2)$. This complexity arises from the required inversion of the $n \times n$ covariance matrix, and it makes the model intractable for use with larger data sets. Commonly used approaches to overcome this drawback are the *Fully Independent Training Conditional* (FITC) introduced by Snelson and Ghahramani [26] and the *Variational Free Energy* (VFE) approach, which was first published by Titsias [27]. Both represent the dataset by a smaller sub set $X_u \subset X$ of so called *inducing points*. In our work, we use the VFE approach as superior properties of the VFE objective function favor the method over the FITC approach [28]. In the following the approach is summarized.

The main idea of VFE is to model the GP posterior distribution directly instead of approximating the prior distribution. To do so, Titsias [27] derived an *augmented true posterior* $p(\mathbf{f}, \mathbf{u}|\mathbf{v})$ as well as an *augmented variational posterior* $q(\mathbf{f}, \mathbf{u})$ and minimized the *Kullback-Leibler-Divergence* between them, i.e. $KL\big(q(\mathbf{f}, \mathbf{u})||p(\mathbf{f}, \mathbf{u}|\mathbf{v})\big)$. The minimization can be solved analytically and leads to the following predictive posterior:

$$q_{VFE}(\mathbf{f}, f^*|\mathbf{v}, X_u, \mathbf{x}^*) = \mathcal{N}\big(Q_{*n}[Q_{nn} + B]^{-1}\mathbf{v}, K_{**} - Q_{*n}[Q_{nn} + B]^{-1}Q_{n*}\big) \tag{2}$$

Here $B = \sigma_n^2 I$ and Q_{ab} is a low rank approximation of the full covariance matrix K_{ab} where $Q_{ab} := K_{au}K_{uu}^{-1}K_{ub}$. VFE reduces the computational time noticeably to $\mathcal{O}(nm^2)$ and the memory complexity to $\mathcal{O}(nm)$ where m is the number of inducing points. To find the hyperparameters of the covariance function, the log marginal likelihood is maximized, which in the VFE model is derived as

$$log\ q_{VFE}(\mathbf{v}|X_u) = -\frac{1}{2}\mathbf{v}^T[Q_{nn}+B]^{-1}\mathbf{v} - \frac{1}{2}\ log\ |Q_{nn}+B| - \frac{n}{2}\ log\ 2\pi - \frac{1}{2\sigma^2}\ Tr(\tilde{K}) \qquad (3)$$

where $\tilde{K} = K_{nn} - Q_{nn}$. The trace term serves as a *regularizer* of the *log marginal likelihood* (LML). Based on the variational derivation, Titsias [27] shows that the regularizer guarantees a lower bound of the true LML of the full GP. To initialize the inducing points X_u of the sGP, *k-means* is applied. More detailed explanations of recording, pre-processing and initialization are described in Section 5.

4 Design of the adaptive model

As mentioned above, a major problem when dealing with EMG data is that the input data are not stationary, i.e. their distribution can quickly change during operation. Therefore, we propose an adaptive approach, which updates the sGP representation when new samples are observed. To obtain robustness against outliers we operate on *batches* of data, i.e. new incoming samples $\mathbf{x}^*$ are collected in a batch X, and we distinguish between "active" batches X^* of detected muscular activity, and "resting" batches X^o containing samples of no muscular activity. As soon as a batch is completed and validated for update, we use it to update the sGP models, i.e. the training data $\mathcal{D}_d$ and the inducing points X_u. Two main steps are required to perform these updates online: first, update rules need to be defined to decide when a batch is valid for update and which label to choose for the batch; secondly, the actual update needs to be performed. A flowchart of the algorithm is depicted in Fig. 1 and explained in more detail as next[1].

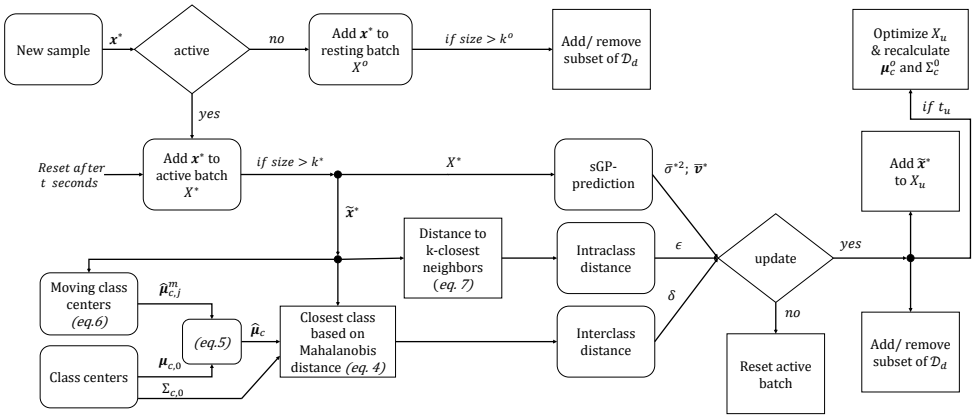


Figure 1: Flowchart of the final adaptive model in the online application.

4.1 Update rules

To be able to update the sGP, we need to create appropriate labels for the incoming data. While we use the sGP as a regressor in the online decoding of EMG signals, we will employ it as a classifier for the labeling task. As described in Section 3.1, the labels used in the training dataset are discrete, i.e. $v_{d,i} \in \{-1, 0, 1\}$. This allows us to consider the directions of motion decoded from the EMG as classes. Here, each degree of freedom of the output is represented by three classes of motion commands. In the case of 2D motion this means that there are the classes "left", "right" and "rest" for the horizontal axis, and the classes "upward", "downward" and "rest" for the vertical axis[2].

Since we are particularly interested in labeling samples for which the sGP has little confidence, we intend to stabilize the classification result by additionally employing the classical measures of inter- and intraclass distance. However, these metrics are not explicitly incorporated in the ensemble of the

[1] A formal description of the algorithm and the update rules is given in the supplementary materials.

[2] Note that "rest" in one DoF can actually relate to motion in the other DoF and vice versa.

sGPs, as their calculation depends on the hyperparameters as well as on the position of the inducing and training points. Therefore, and because we actually use independent sGPs for individual cartesian DoF, inter- and intraclass distance w.r.t. the data points can be inferred only indirectly, resulting in a relative distance measure. For each class c, we compute a mean vector $\boldsymbol{\mu}_{c,0}$ and a covariance matrix $\Sigma_{c,0}$ from all samples of the initial training dataset that correspond to this class c. Similarly, the batch X^* can be represented by its median $\tilde{\mathbf{x}}^*$. Here, the median was chosen over the mean, as the variance in the batch can be comparably large and the median is more robust against outliers.

Using the above variables, we can compute the interclass distance δ_c based on the *Mahalanobis distance* between the batch X^* and all available classes, except the resting class.

$$\delta_c(\tilde{\mathbf{x}}^*, \hat{\boldsymbol{\mu}}_c) = \sqrt{(\tilde{\mathbf{x}}^* - \hat{\boldsymbol{\mu}}_c)^T \Sigma_c^{-1} (\tilde{\mathbf{x}}^* - \hat{\boldsymbol{\mu}}_c)}; \qquad c = 1, ..., 4 \tag{4}$$

Here, we use $\hat{\boldsymbol{\mu}}_c$ as a modified version of the mean of each class $\boldsymbol{\mu}_{c,0}$, which is defined as:

$$\hat{\boldsymbol{\mu}}_c = \frac{\boldsymbol{\mu}_{c,0} + \boldsymbol{\mu}_{c,j}^m}{2} \tag{5}$$

Where $\boldsymbol{\mu}_{c,j}^m$ represents the moving class centers, which, after the jth batch has been processed, get pulled towards the median of that batch. Accordingly, it is initialized as $\boldsymbol{\mu}_{c,0}^m = \boldsymbol{\mu}_{c,0}$ and the update of the two closest classes ($\boldsymbol{\mu}_{1,j+1}^m$ and $\boldsymbol{\mu}_{2,j+1}^m$) is achieved by:

$$\boldsymbol{\mu}_{1,j+1}^m = \boldsymbol{\mu}_{1,j}^m + \alpha_1 \left(\tilde{\mathbf{x}}^* - \boldsymbol{\mu}_{1,j}^m\right) \qquad \textit{and} \qquad \boldsymbol{\mu}_{2,j+1}^m = \boldsymbol{\mu}_{2,j}^m + \alpha_2 \left(\tilde{\mathbf{x}}^* - \boldsymbol{\mu}_{2,j}^m\right). \tag{6}$$

Here, $1 > \alpha_1 > \alpha_2 > 0$ are hyperparameters of our method to adjust the adaptation speed of the model. The motivation of averaging the training data and the new data arises from the idea to integrate information of every batch, even though the batch is not used to update the training set in case none of the update rules is fulfilled. Now, since interclass-distance is used to express how close the data is to existing classes, it can be reduced to a single value by dividing the distance to the closest class δ_1 by the distance towards the second closest class δ_2, resulting in: $\delta = \frac{\delta_1}{\delta_2}$.

Accordingly, the intraclass distance ϵ is the mean of the distances between the closest k data points of the training set to the median of the batch. Here, ϵ is just calculated once and expresses how close data of the training set (including data of all classes) are to the median of the batch.

$$\epsilon(\tilde{\mathbf{x}}^*, X^k) = \frac{1}{k} \sum_{i=1}^{k} \exp\left(-\frac{1}{2} \frac{(\tilde{\mathbf{x}}^* - \mathbf{x}_i)^2}{l^2}\right) \tag{7}$$

Where X^k are the k closest neighbors and l corresponds to the optimized length-scale of one sGP.

Based on the metrics δ, ϵ, the mean output $\bar{\mathbf{v}}^*$ and the average standard deviation $\bar{\sigma}^{*2}$ of the batch update rules can be defined to determine if a batch is used for an update and to create suitable labels. Here, $\bar{\sigma}^{*2}$ is the average of $\bar{\boldsymbol{\sigma}}^{*2}$ which is a two-dimensional vector and expresses for each sGP the mean of the predicted standard deviations of one batch. The first and most obvious update rule is fulfilled if a batch is close to just a single class, resulting in small δ and high ϵ, in combination with high certainty of the sGPs (small $\bar{\sigma}^{*2}$ and one entry of $\bar{\mathbf{v}}^*$ predominates). Accordingly, the label is chosen based on the closest class. The next update rule handles batches in which the certainty of the sGP is low ($\bar{\sigma}^{*2}$ closer to one) but a low δ indicates that only one class is close. As a result, we can also use the closest class for labeling.

Besides these obvious cases, a value of δ close to one requires additional update rules. Usually, if a batch is located in between two classes, ϵ, as well as $\bar{\sigma}^{*2}$ result in medium sized values. Here, the resulting prediction $\bar{\mathbf{v}}^*$ can be used to select a valid label. In case $\bar{\mathbf{v}}^*$ shows a predominant entry, that entries class is used as a label (update rule 3). In case the resulting velocity is equal in two directions, this indicates that the user is trying to move in a diagonal direction. To be able to handle these cases, update rule 4 acts with an extended set of possible labels, i.e. $v_i \in \{-1,\ -0.707,\ 0,\ 0.707,\ 1\}$ and the batch is labeled as diagonal motion, accordingly. The diagonal labels were chosen such that their absolute velocity is equal to one. Including $\mathbf{v}$ as a metric has an important effect on the usability of the incremental learning approach, as the system updates correspond to the visual impression of the user. This way the model is less likely to make inaccurate updates.

While the first four update rules define the label of a batch, additional rules are added to decide whether new inducing points are needed. For one, an inducing point is added at the center of the batch, when one of the update rules is activated and $\bar{\sigma}^{*2}$ passes a rule specific threshold. Furthermore, inducing points are added if δ is small and $\bar{\sigma}^{*2} \approx 1$, as well as if ϵ and $\bar{\sigma}^{*2}$ are large. Adding inducing points is intended to support the sGPs to increase certainty in areas where labeled data points lie.

(a) (b) (c)

Figure 2: (a) and (b) are examples of position $P1$ and position $P2$, respectively; (c) shows the simulation environment including the EMG-controlled cursor and the target.

4.2 Updating the sGP

Once a batch is eligible for an update, a random subset of samples of the batch is selected and added to each $\mathcal{D}_d$ using the respective label. Additionally, to keep the amount of stored data constant, and the quantity of samples in each class consistent, samples from the same class as the batch are randomly removed from each sGPs data set[3]. It needs to be emphasized that each trained sGP has its own data and inducing points stored internally. This way the models differ continuously more from each other and let the observed space grow constantly. Furthermore, the metrics δ and ϵ only act on classes along the main axes, ignoring data which is labeled as *diagonal*. After $\mathcal{D}_d$ has been updated t_{u2} times, $\boldsymbol{\mu}_{c,0}$ and $\Sigma_{c,0}$ are recalculated, however, to reduce computational time and to further increase the influence of newly added points, just a subset ($\mathcal{D}_{sub}$) of $\mathcal{D}_d$ is taken, which contains more samples from the end of the datasets (i.e. data of the updates). Furthermore, the amount of inducing points is limited to u_{max} and optimization of them is performed after t_{u1} updates or if update rule 5 or 6 is activated.

A time dependent reset of the *active batch* (0.1 s) as well as a *resting batch* to update the zero-labeled data points of $\mathcal{D}_d$ are included. To avoid stagnating control, early stopping during the training process as well as constraining the length-scale to lie between 0.3 and 0.45 is applied. Finally, to guarantee the real time control (evaluation of the decoding algorithm runs at 100 Hz), all computations of the incremental update are moved to a separate thread, which runs in parallel to the decoding process and updates the sGP models as soon as calculations are finished.

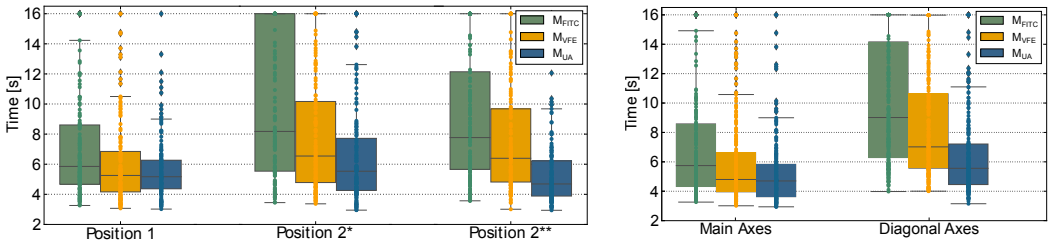
5 Experiments

To evaluate the adaptive model presented in this work, an online user study was conducted. Therefore, a 2D task on a screen was performed by six subjects using the EMG-based interface. Different conditions were used to evoke changes in the input data to challenge the decoding process.

5.1 Data recording, pre-processing and initialization

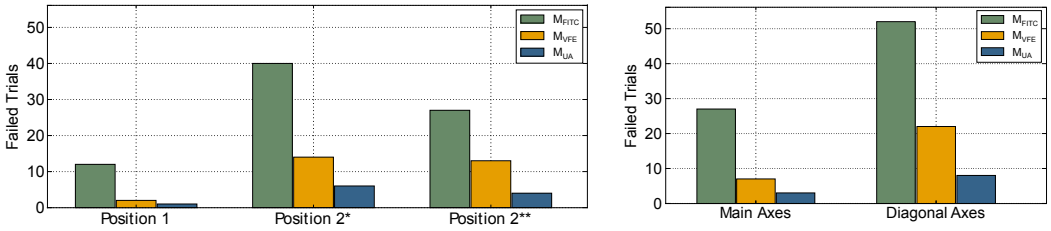
In this study eight DELSYS *Trigno Wireless* EMG sensors were attached to the skin of the user's dominant arm. To realize the commands, the sensors were attached to various muscles of the limb: four electrodes to extensors and flexors of wrist and fingers and four to biceps, triceps, pectoralis and deltoid, respectively. A GUI was utilized to guide the user through the supervised training procedure. Here, subjects were asked to execute unique patterns of constant contraction for each direction, which were labeled with $v_{d,i} \in \{-1, 0, 1\}$. The sGP regression allowed later for a continuous commanding when weaker contractions occurred. After testing different EMG features and the combinations of them, only the WL was utilized as it showed a good and stable performance using the adaptive model. To process the labeled data gathered from the training session, the Python library *Pyro* was used. This library offers an implementation of the VFE GP approximation, where a Gaussian likelihood as well as the Radial Basis Function as kernel function ($k(\mathbf{x_i}, \mathbf{x_j}) = \sigma_s \cdot \exp\left(-0.5(\mathbf{x_i} - \mathbf{x_j})^2(l^2)^{-1}\right)$) were selected. To optimize the hyperparameters signal variance (σ_s), length-scale (l) and noise variance (σ_n) an *Adam* optimizer was used and the mean function of the sGP was set to be constantly zero. Minor adaptations within the library ensure that structures which are computational expensive, e.g. the *Cholesky decomposition*, were only recomputed when an update of $\mathcal{D}$ or X_u was performed. Our initial approach, presented in [24], which was based on the FITC approximation and implemented in the *pyGPs* library [29], was used as a baseline in the user study. It is using the optimization algorithm

[3]Since the initial dataset does not contain any data points with diagonal labels, removal of samples of a diagonal class is omitted, until at least 500 samples of that class have been added to $\mathcal{D}_d$



(a) Positions (b) Directions

Figure 3: Completion time of the user study; (a) compares the results of three tested models in different Positions; (b) compares the results of the models for main and diagonal axes.



(a) Positions (b) Directions

Figure 4: Failed trials of the user study; (a) shows the failed trials out of 144 trials per position and model over all subjects; (b) shows the failed trials out of 216 trials per direction and model.

suggested by Rasmussen [30]. This baseline approach is further referred to as M_{FITC}, while our unsupervised adaptive model is referred to as M_{UA}.

5.2 User study

A user study should show how our novel adaptive model M_{UA} performs when covariate shift emerges, in comparison to the non-adapting model M_{VFE} and the baseline M_{FITC} in an online application. Six healthy subjects with almost no experience in EMG control (maximum one usage before) participated in the study. During the experiments subjects had to sit in front of the test setup, including a screen, as well as a handle to hold on to with their dominant hand with an angled elbow. Subjects were asked to generate forces against the handle to produce the needed muscular activity for the different directions. To test the algorithm's robustness to covariate shifts, different arm positions were realized. The starting position was a comfortable position in front of the setup, named position $P1$. To evoke changing input data, the subjects were asked to extend the elbow by moving the chair further away. The position was called $P2$. The arm configurations are shown in Figure 2(a) and (b)).

The experimental test was designed in a two-dimensional virtual environment, where the user was asked to navigate a cursor into a red target circle as fast as possible (Figure 2(c)). The eight different target locations were lying all at the same distance from the center of the simulation environment. To strain the model and to test its stability, the user had to match targets along the diagonals additionally to those lying along the initially trained coordinate axis. The targets appear in a random order and after the user reaches a target the cursor is automatically reset to the center.

One experimental session included three rounds structured as follows: each round starts with an initial training of the models on a dataset recorded in position $P1$, in order to initialize all models with identical training data and inducing points. Starting with a randomly selected model of M_{UA}, M_{FITC} or M_{VFE}, unknown to the subject, all targets had to be reached once in $P1$ and twice in $P2$ (called $P2^*$ and $P2^{**}$). Afterwards the same procedure was applied successively to the other models.

The main parameter to be evaluated is the time required to reach a target. If a subject is not able to reach a target, the trial terminates automatically after 15s and a penalty of one second is added for not completing the task. Next to the quantitative parameter, the subjective impression of the users w.r.t. each of the different models is obtained by a questionnaire.

6 Results

The boxplots in Figure 3 depict the time the subjects needed to finish a task using the different models w.r.t. the varying positions ($P1$, $P2^*$, and $P2^{**}$) shown in 3(a) and the navigation axes (main as well as diagonal axes) in 3(b). Each dot in the boxplot corresponds to one trial of one subject under the specified condition. The completion time of the three models during $P1$ differ only slightly, whereas the changed position $P2$ leads to a noticeable increase in completion time in all models. In $P2^{**}$ a slight increase of performance is notable compared to $P2^*$. Figure 3(b) shows further that the time to reach the targets on the main axes, on which the model is trained, is smaller than on the diagonals. Results for individual directions, positions and models are given in Figure 1 of the supplementary materials. The amount of trials which could not be completed for a specific position or direction are depicted in Figure 4. The changed Position $P2$ leads to higher failure rates, especially for the basis model M_{FITC}. Also, the diagonal axes evoke relatively more failures as the main axes for all models.

A statistical analysis was performed to show whether M_{UA} results in lower completion times than M_{VFE} or M_{FITC}. As the histograms of the distributions do not follow a normal distribution the Friedman test, a non-parametric test, was applied (all histograms are part of the supplementary materials). Since it is a rank based test, the amount of penalization of incomplete trials does not play a role. The Friedman test was performed on all positions ($P1$, $P2^*$, and $P2^{**}$) as well as on the directions (main axes and diagonal axis). All five statistical analysis show a significant difference of the distributions. The post hoc test from Nemenyi [31] confirmed that all three models differ significantly from each other in almost all cases. The only exception was found in $P1$ where M_{VFE} and M_{UA} do not differ significantly. The latter is no surprise as the models M_{VFE} and M_{UA} initially are identical. All results of the Friedman test can be found in Table 1 of the supplementary material.

The analysis of the questionnaire, (cf. Figure 1 of the supplementary material) reveals that the subjects were most satisfied with their performance during M_{UA}, which supports the results of the evaluation of the completion time. The *general control* of M_{UA} is rated on average more than one point better than all other models. Navigating on the diagonals compared to navigating on the main axes is less satisfying. The subjects also identified an improvement of the control using the M_{UA} model.

7 Discussion

The pilot study with six non-disabled subjects shows a significant increase in performance of the adaptive model M_{UA} over the non-adaptive models, which is verified in a quantitative and qualitative analysis of the user study. In the study, diagonal navigation as well as change of arm position confront the model with highly challenging conditions of covariate shift, which was handled reliably by our approach. The stabilization of the performance from $P2^*$ to $P2^{**}$ demonstrates the capability of the algorithm to handle the covariate shift also over a longer period of time. The update rules, diagonal labeling and incorporation of the model output $\bar{\mathbf{v}}^*$ close the loop between user and system and lead to stable incremental updates of the sGP in this use-case. Apart from the update rules, the whole structure of the adaptive model has contributed to its performance. For instance, the parallel threads, responsible for predictions and updates, guarantee the real time control and stability of the implementation.

The user study also reveals an enhanced performance of M_{VFE} compared to M_{FITC}. However, it is unclear whether the approximation approach is the only reason for the difference in performance. For one, we ran the VFE model with a different set of EMG-features. Additionally, the length-scale was constrained in our implementation of the VFE Model, which may have led to better generalization.

Future work will include a long term evaluation and extension of the approach to 3D, as well as testing with disabled subjects. Also, the complexity of the problem leaves further avenues for improvement, for instance proportional control, easier control of the frequency of the updates, rethinking the metrics, the influence of the batch-size, or increasing of computational efficiency (especially w.r.t. the optimization of inducing points).

Acknowledgments

We would like to thank Gabriel Quere for his valuable comments on the manuscript, as well as our subjects who attended the user study. A special thanks goes to the Reviewers, whose valuable comments helped us to improve the quality of the manuscript.

References

[1] A. Hagengruber and J. Vogel. Functional tasks performed by people with severe muscular atrophy using an sEMG controlled robotic manipulator. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1713–1718, 07 2018. doi:10.1109/EMBC.2018.8512703.

[2] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann. The extraction of neural information from the surface EMG for the control of upper-limb prostheses: emerging avenues and challenges. *IEEE Trans. on Neural Systems and Rehab. Engineering*, 22 (4):797–809, 2014. doi:10.1109/TNSRE.2014.2305111.

[3] M. M. Vidovic, H. Hwang, S. Amsüss, J. M. Hahne, D. Farina, and K. Müller. Improving the robustness of myoelectric pattern recognition for upper limb prostheses by covariate shift adaptation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 24(9): 961–970, 2016. ISSN 1534-4320. doi:10.1109/TNSRE.2015.2492619.

[4] A. Gepperth and B. Hammer. Incremental learning algorithms and applications. *European Sympoisum on Artificial Neural Networks (ESANN)*, pages 357–368, 2016. URL `https://hal.archives-ouvertes.fr/hal-01418129`.

[5] E. Scheme and K. Englehart. Electromyogram pattern recognition for control of powered upper-limb prostheses: State of the art and challenges for clinical use. *Journal of rehabilitation research and development*, 48, 2011. doi:10.1682/JRRD.2010.09.0177.

[6] W. Bishop, C. C. Chestek, V. Gilja, P. Nuyujukian, J. D. Foster, S. I. Ryu, K. V. Shenoy, and M. Y. Byron. Self-recalibrating classifiers for intracortical brain–computer interfaces. *Journal of neural engineering*, 11(2):026001, 2014.

[7] B. Jarosiewicz, A. A. Sarma, D. Bacher, N. Y. Masse, J. D. Simeral, B. Sorice, E. M. Oakley, C. Blabe, C. Pandarinath, V. Gilja, et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. *Science translational medicine*, 7(313): 313ra179–313ra179, 2015.

[8] D. Sussillo, S. D. Stavisky, J. C. Kao, S. I. Ryu, and K. V. Shenoy. Making brain–machine interfaces robust to future neural variability. *Nature communications*, 7:13749, 2016.

[9] J. Liu, X. Sheng, D. Zhang, N. Jiang, and X. Zhu. Towards zero retraining for myoelectric control based on common model component analysis. *IEEE Trans. on Neural Systems and Rehab. Engineering*, 24(4):444–454, 2016. ISSN 1534-4320. doi:10.1109/TNSRE.2015.2420654.

[10] J. Liu, D. Zhang, X. Sheng, and X. Zhu. Enhanced robustness of myoelectric pattern recognition to across-day variation through invariant feature extraction. *Annual Intern. Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015:7262–7265, 2015. doi:10.1109/EMBC.2015.7320068.

[11] V. Jayaram, M. Alamgir, Y. Altun, B. Schölkopf, and M. Grosse-Wentrup. Transfer learning in Brain-Computer Interfaces. *IEEE Computational Intelligence Magazine*, 11(1):20–31, 2016. doi:10.1109/MCI.2015.2501545. URL `http://arxiv.org/pdf/1512.00296v1`.

[12] X. Zhai, B. Jelfs, R. H. M. Chan, and C. Tin. Self-recalibrating surface EMG pattern recognition for neuroprosthesis control based on convolutional neural network. *Frontiers in Neuroscience*, 11:379, 2017. doi:10.3389/fnins.2017.00379. URL `https://www.frontiersin.org/article/10.3389/fnins.2017.00379`.

[13] J. W. Sensinger, B. A. Lock, and T. A. Kuiken. Adaptive pattern recognition of myoelectric signals: Exploration of conceptual framework and practical algorithms. *IEEE Trans. on Neural Systems and Rehab. Engineering*, 17(3):270–278, 2009. ISSN 1534-4320. doi:10.1109/TNSRE.2009.2023282.

[14] Q. Huang, D. Yang, L. Jiang, H. Zhang, H. Liu, and K. Kotani. A novel unsupervised adaptive learning method for long-term electromyography (EMG) pattern recognition. *Sensors (Basel, Switzerland)*, 17(6), 2017. doi:10.3390/s17061370.

[15] A. Gijsberts, R. Bohra, D. Sierra González, A. Werner, M. Nowak, B. Caputo, M. A. Roa, and C. Castellini. Stable myoelectric control of a hand prosthesis using non-linear incremental learning. *Frontiers in neurorobotics*, 8:8, 2014. ISSN 1662-5218. doi:10.3389/fnbot.2014.00008.

[16] X. Chen, D. Zhang, and X. Zhu. Application of a self-enhancing classification method to electromyography pattern recognition for multifunctional prosthesis control. *Journal of neuro-engineering and rehabilitation*, 10(1):44, 2013. doi:10.1186/1743-0003-10-44.

[17] J. M. Hahne, S. Dähne, H. Hwang, K. Müller, and L. C. Parra. Concurrent adaptation of human and machine improves simultaneous and proportional myoelectric control. *IEEE Trans. on Neural Systems and Rehab. Engineering*, 23(4):618–627, 2015. ISSN 1534-4320. doi:10.1109/TNSRE.2015.2401134.

[18] R. Paul, R. Triebel, D. Rus, and P. Newman. Semantic categorization of outdoor scenes with uncertainty estimates using multi-class Gaussian Process classification. In *Proc. of the International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[19] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Active learning with Gaussian Processes for object categorization. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. doi:10.1109/ICCV.2007.4408844.

[20] A. Freytag, E. Rodner, P. Bodesheim, and J. Denzler. Rapid Uncertainty Computation with Gaussian Processes and Histogram Intersection Kernels. In K. M. Lee, Y. Matsushita, J. M. Rehg, and Z. Hu, editors, *Computer Vision – ACCV 2012*, pages 511–524, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37444-9. doi:10.1007/978-3-642-37444-9_40.

[21] A. Ranganathan, M.-H. Yang, and J. Ho. Online sparse Gaussian process regression and its applications. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 20(2):391–404, 2011. doi:10.1109/TIP.2010.2066984.

[22] R. Triebel, J. Stühmer, M. Souiai, and D. Cremers. Active online learning for interactive segmentation using sparse gaussian processes. In *German Conference on Pattern Recognition*. Springer, 2014.

[23] B. Hudgins, P. Parker, and R. N. Scott. A new strategy for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering*, 40(1):82–94, 1993.

[24] J. Vogel and A. Hagengruber. An sEMG-based interface to give people with severe muscular atrophy control over assistive devices. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2136–2141, 2018.

[25] C. Rasmussen and C. K. I. Williams. *Gaussian processes for Machine Learning*. The MIT Press., 2006.

[26] E. Snelson and Z. Ghahramani. Sparse Gaussian Processes using pseudo-inputs. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 1257–1264, Cambridge, MA, USA, 2005. MIT Press.

[27] M. K. Titsias. *Variational Model Selection for Sparse Gaussian Process Regression*. PhD thesis, University of Manchester, 2009.

[28] M. Bauer, M. van der Wilk, and C. E. Rasmussen. Understanding probabilistic sparse gaussian process approximations, 2016. URL `http://arxiv.org/pdf/1606.04820v2`.

[29] M. Neumann, S. Huang, D. E. Marthaler, and K. Kersting. pyGPs – a python library for Gaussian Process Regression and Classification. *Journal of Machine Learning Research*, 16 (80):2611–2616, 2015. URL `http://jmlr.org/papers/v16/neumann15a.html`.

[30] C. E. Rasmussen. minimize.m: minimize a smooth differentiable multivariate function using LBFGS (Limited memory LBFGS) or CG (Conjugate Gradients). `http://www.cs.cmu.edu/~arunvenk/academics/neural/framework/src/GP/GP_utils/minimize.html`, 1996.

[31] P. Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Ph. D. Princeton University, 1963.