Extracting Low-Dimensional Control Variables for Movement Primitives

Extrahieren niedrig dimensionaler Kontrollvariablen für Bewergungsprimitive

Master-Thesis von Jan Mundo

Tag der Einreichung:

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Dr. Elmar Rückert
3. Gutachten: Prof. Dr. Gerhard Neumann

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Intelligent Autonomous Systems

Betreuer: Dr. Elmar Rückert, Prof. Dr. Gerhard Neumann, Prof. Dr. Jan Peters

Extracting Low-Dimensional Control Variables for Movement Primitives
Extrahieren niedrig dimensionaler Kontrollvariablen für Bewergungsprimitive

Vorgelegte Master-Thesis von Jan Mundo

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Dr. Elmar Rückert
3. Gutachten: Prof. Dr. Gerhard Neumann

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 29. Oktober 2014

_____

(Jan Mundo)

Abstract

In robotics we often want to solve a multitude of different, but related tasks. Movement primitives (MPs) provide a powerful framework for data driven movement generation that has been successfully applied for learning from demonstrations and robot reinforcement learning. As the parameters of the primitives are typically high dimensional, a common practice for the generalization of movement primitives to new tasks is to adapt only a small set of control variables, also called meta parameters, of the primitive. Yet, for most MP representations, the encoding of these control variables is pre-coded in the representation and can not be adapted to the considered tasks. In this thesis, we want to learn the encoding of task-specific control variables also from data instead of relying on fixed meta-parameter representations. We use hierarchical Bayesian models (HBMs) to estimate a low dimensional latent variable model for probabilistic movement primitives (ProMPs), which is a recent movement primitive representation. We show on two real robot datasets that ProMPs based on HBMs outperform standard ProMPs in terms of generalization and learning from a small amount of data and also allows for an intuitive analysis of the movement. We also extend our HBM to a mixture model, such that we can model different movement types in the same dataset.

Zusammenfassung

In der Robotik betrachten wir oft eine Vielzahl von verschiedenen aber verwandten Aufgaben. Bewegungsprimitive bieten ein mächtiges Framework für datengesteuertes Erzeugen von Bewegungen, welches erfolgreich auf Lernen von Demonstrationen und Reinforcement Learning angewandt wurde. Da die Parameter der Bewergungsprimitiven typischerweise hochdimensional sind, ist es gängig für die Generalisierung von Bewegungsprimitiven nur eine kleine Zahl von Kontrollvariablen - auch Metaparameter genannt - anzupassen. Allerdings sind für die meisten Repräsentationen von Bewegungsprimitiven diese Kontrollvariablen vorab kodiert und können nicht an die betrachtete Aufgabe angepasst werden. In dieser Arbeit wollen wir die Kodierung von aufgabenspezifischen Kontrollvariablen von den Daten lernen, anstatt vorab kodierte Metaparameter zu verwenden. Wir verwenden bayessche Netze um ein Modell mit niederdimensionalen, latenten Variablen für probabilistische Bewegungsprimitive zu lernen. Probabilistische Bewegungsprimitive sind eine kürzlich entwickelte Repräsentation von Bewegungsprimitiven. Wir zeigen anhand von zwei Robotik-Datensätzen dass unsere Methode die Standardformulierung der probabilistischen Bewegungsprimitiven im Hinblick auf Generalisierung und Lernen von wenigen Daten übertrifft. Außerdem ermöglicht uns das vorgestellte Modell eine intuitive Analyse von Bewegungen. Wir erweitern unser Modell zu einem Mixture Modell, sodass wir verschiedene Bewegungstypen in demselben Datensatz lernen können.

Contents

1 Introduction

Robots with high-dimensional joint spaces are widely used and find more and more fields of application, because they have the capability to perform very complex tasks. With increasing complexity of tasks the ability of learning from demonstrations is crucial, since it is very challenging to define the tasks by hand. In the pasts reinforcement learning and imitation learning have risen to great success in this area. When learning or analyzing movement data, we have to deal with several challenges. These challenges can be high-dimensional data, missing data or partial observations, multi-modal data or noise. Another requirement is that the robot can generalize from the learned tasks as it is hard to learn configuration of all possible tasks. Such tasks can be a grasping movement, lifting objects, motions that avoid an obstacle or even complex tasks as table tennis strokes or golf movements. For representing such tasks a suitable choice are movement primitives (MPs). Movement primitives are a compact parametric description of a movement [20, 9, 11, 6]. They provide a powerful framework for data driven movement generation as they can be learned from demonstrations as well as by reinforcement learning. They can generalize to a new task by adapting a given set of meta-parameters [29, 13, 16]. Such parameters can be the final joint positions or the execution speed of the movement [9]. Yet, for most movement primitive representations, the set of meta-parameters is pre-coded into the movement primitive representation and can not be adapted. However, for most tasks, a different encoding of the meta-parameters might be more appropriate than the pre-coded parameters of the primitive representation. We believe that this shortcoming has also hindered the application of movement primitives for more complex multi-task learning applications. In this work we want to learn the encoding of the meta-parameters also from data. Therefore we propose an approach which extracts a low-dimensional manifold in the MP parameter space. Each point on this manifold is described by a small set of control variables. Hence, our underlying assumption is that, while the parametrization of movements might be high-dimensional, useful parameter vectors for a given set of tasks typically share a lot of structure. For instance they lie on a lower dimensional manifold. Each demonstration can now be characterized by the corresponding control variables that can be seen as a compact description of the task considered in this demonstration. For example, in a table tennis scenario, these control variables could specify the location of the hitting point or the desired return direction for the ball. Hence, our model can not only be applied for efficient generalization in multi-task learning with movement primitives but is also well suited for analyzing the movements of human demonstrators. We implement the latent manifold model by a Hierarchical Bayesian Model (HBM). The control variables for each demonstrations are treated as latent variables and are also inferred from the data. The model is extended by a mixture model such that we can learn the control variables of multiple types of movements. We will use Probabilistic Movement Primitives (ProMPs) as underlying movement primitive representation as they can be naturally integrated in the HBM representation as they ProMPs already define a simple hierarchical Bayesian model.

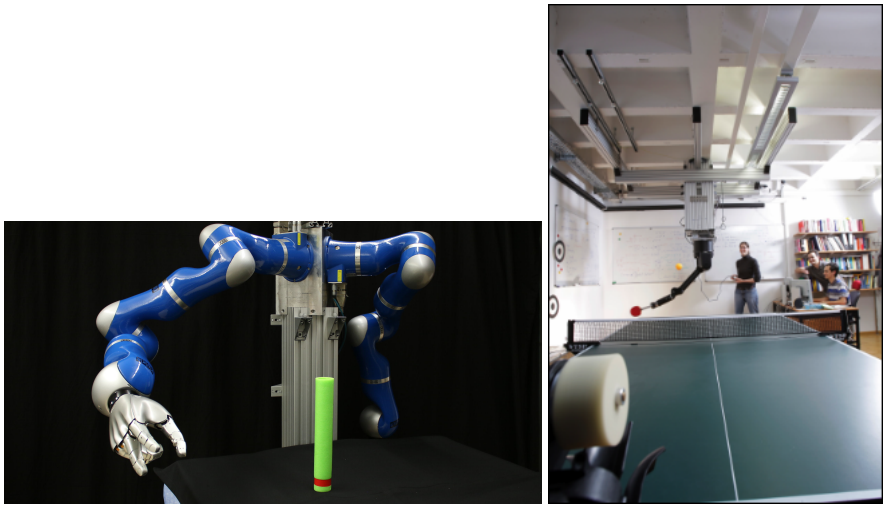

Figure 1.1.: The robots used in the main experiments to learn trajectory distributions. Shown onn the left panel is Darias consisting of two Kuka light weight robot arms. On the right the table tennis setup from IAS with a Barrett WAM robot and highspeed cameras is illustrated. We thank Katharina Muelling for providing the table tennis data and Axel Griesch for creating this picture.

1.1 Outlook of thesis

In Section 2 we discuss the related work containing movement primitives and multi-task learning. In Section 3 we discuss variational inference and Probabilistic Movement Primitives as they provide the foundation of this thesis. Afterwards we extend the formulation of ProMPs by a hierarchical prior to encode low-dimensional latent control variables to obtain a more accurate model of the prior distribution in Section 4. Results are represented in Section 5 where we evaluate our proposed model on two kinestethic teaching datasets. We will illustrate the improved generalization properties of our approach compared to the standard ProMP approach in the case of a small amount of training data, noise and partial observations. In our experiments, we use high-dimensional robots in complex setups such as table tennis as shown in Figure 1.1. We also show how demonstrations can be easily analyzed and characterized by the extracted latent control variables.

2 Related Work

In this section we give an overview on movement primitives and multi-task learning methods as we extract control variables for movement primitives inspired from models used on multi-task learning.

2.1 Movement Primitives

Movement primitives can be categorized into trajectory-based [9, 26, 20] and state-based representations [11]. In this thesis we will focus on trajectory based approaches as they are more commonly used and easier to scale up to higher dimensions. A common trajectory-based approach are the dynamical movement primitives (DMPs). DMPs [9] are represented by a parametrized dynamical system that is given by a linear point-attractor that is perturbed by a non-linear time dependent forcing function. The forcing function can be used to encode an arbitrary shape of the trajectory and the weights of the forcing function can be easily obtained from demonstrations by linear regression. One of the benefits of the DMP approach is that it specifies a small set of meta-parameters. These meta-parameters include the final position of the movement, which is given by the point attractor, the final velocities, the execution speed, or the amplitude of the movement [12, 23, 9]. In multi-task learning with DMPs [16, 13], it is a common strategy to only adapt the meta-parameters due to the high dimensionality of the weights of the forcing function. While DMPs have several more benefits such as stability, and the ability to represent stroke based and rhythmic movements, they also have several limitations, such as that they can not represent optimal behavior in stochastic systems, the adaptation of the trajectory due to the meta-parameters is based on heuristics and it is unclear how to combine DMPs simultaneously or to continuously switch from one DMP to another DMP.

These issues have been fixed by the recently proposed Probabilistic Movement Primitives approach (ProMPs)[20, 21]. ProMPs estimate a distribution of trajectories instead of single trajectories. The main benefit of the probabilistic representation is that we can use probabilistic operators such as conditioning for adaptation and a product of distribution for simultaneous combination. A distribution over trajectories also contains information on which time points are relevant for the movement, e.g., time points with small variance in the Cartesian end-effector space could denote task relevant via-points or targets. More details on ProMPs are given in Section 3.1.

However, in difference to DMPs, ProMPs are lacking meta-parameters that can be used to adapt the trajectories with a small amount of control variables. It would be easy to pre-specify such control variables by conditioning the trajectory distribution for a fixed set of time points. However such an approach would again require a lot of manual tuning and is lacking flexibility. In this thesis we face this issue by extending the prior distribution of the original ProMP formulation with a hierarchical prior which contains the desired control variables. This approach is part of my thesis and discussed in Section 4.

2.2 Multi-task learning

The approach developed for this thesis automatically extracts a small amount of control variables from a given set of demonstrations in the ProMP framework. We use a hierarchical Bayesian approach to model prior distributions, which is inspired by techniques from multi-task learning (MTL) [4, 31, 18, 15, 25, 27]. multi-task Learning has received a lot of intention over the last two decades and still is an important research area [31, 18, 15, 25, 27]. It is not only used on robotics but also in research areas as computer vision or medical science. It first is defined by Caruana in [4] as

> *"Multi-task Learning (MTL) is an inductive transfer mechanism whose principle goal is to improve generalization performance (...) by leveraging the domain-specific information contained in the training signals of related tasks."*.

The standard methodology in machine learning dividing complex problems into subproblems, which are solved independent from one another suffer from the fact that the rich information shared across these subproblems is completely lost. In MTL the underlying assumption is that multiple tasks (or trajectories) share a common structure, and, hence, with an increasing number of related tasks that have been already learned, the number of needed training samples for generalizing to a new task decreases [2]. This property is highly desired in robotics, where the data is often high dimensional and obtaining training samples is costly. Different approaches exist to model the shared information across tasks. They can be roughly separated into two different categories The first category describes methods where parameters of the model are close to each other in a geometric sense [8, 27]. In these methods often an objective function is defined which is to be optimized. Since the objective function is assumed to be convex or approximately convex one can use Lagrangian multiplier and dual theory. Some work also is done on non-convex objective functions [1]. The second category contain models where parameters of the model share a common structure [31, 30, 5, 18, 24, 22]. This structure can be a clustering assumption [30], a (Gaussian) prior distribution for the parameters of all tasks [31, 18] or some advanced structure like the Kingman's coalescent [5], which is a continuous time, partitioned valued Markov process. While we concentrate on a continuous latent spaces there exists some approaches considering discrete prior distributions [10]. In the case where the tasks share a common prior distribution in continuous or discrete [10] settings. Also a conventional approach is to model the relatedness of the tasks in the covariance function of the prior distribution. In high-dimensional parameter spaces this is prone to overfitting if only a small amount of training data is available. Also it is harder to analyze such prior distributions in an intuitive fashion. Another problem of multi-task learning is negative transfer, where learning task simultaneously slows down the performance [28].

Our approach is highly related to the Bayesian MTL approach presented in [22], where a prior distribution over parameters is learned. The prior distribution is assumed to have a low-dimensional, latent structure that is represented by a linear factor model. In order to represent several modes (or non-linearities) in the data, the model is extended to a mixture model of linear factor models. For both, the number of mixture components and the number of factors, a non-parametric Dirichlet prior has been used. All parameters of the model are integrated out by the use of a combination of sampling and variational inference. We will use a simplification of this model, assuming a fixed number of mixture components, without the Dirichlet priors, allowing a much more efficient algorithm without the need for expensive

sampling methods. We extend the model of Passos et al. by an additional hyper-prior and show that this hyper-prior significantly increases the robustness of the Bayesian model.

3 Methods

As Probabilistic Movement Primitives provide the foundation of our proposed method we will introduce them in this section in detail. Additionally we will give a short overview on variational methods for latent variable models, because it we use variational inference as our model learning approach.

3.1 Probabilistic Movement Primitives

Movement Primitives (MP) in general are used as building blocks in highly complex scenarios with high-dimensional compliant robots. In Section 2.1 we stated that switching between movement primitives or coactivate multiple movement primitives are some of the limitations of DMPs. Facing these problems a probabilistic formulation of movement primitives are developed, called Probabilistic Movement Primitives (ProMPs) [20]. The framework of the ProMPs encode all desirable properties of a MP as coactivation, modulation, temporal scaling and learning. Additionally it provides a stochastic time varying feedback controller to reproduce a given trajectory distribution. Overall ProMPs build a powerful framework for representing basic elementary movements, such as hitting or grasping, implementing a lot of desired properties of MPs.

ProMPs represent a movement by a distribution $p(\boldsymbol{\tau})$ over trajectories $\boldsymbol{\tau} = \boldsymbol{y}_{1:T}$, where $\boldsymbol{y}_t$ specifies the joint positions (or any other quantities, such as a Cartesian coordinates of a ball) at time step t, while T denotes the final time step. ProMPs use a linear basis function model with J basis functions to represent a single trajectory

$$p(\boldsymbol{y}_t|\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{y}_t \middle| \boldsymbol{\Psi}_t \boldsymbol{w}, \beta^{-1}\boldsymbol{I}_S\right) \text{ and } p(\boldsymbol{\tau}) = \prod_{t=1}^{T} p\left(\boldsymbol{y}_t \middle| \boldsymbol{w}\right),$$

where β denotes the precision of the S-dimensional data. The weight vector $\boldsymbol{w}$ is a compact representation of the trajectory. The basis functions $\boldsymbol{\Psi}_t$ only depend on the time or, alternatively, on the phase of the movement. For a single Degree of Freedom (DoF), $\boldsymbol{\Psi}_t$ is just given by a vector of normalized Gaussian basis functions with

$$b_{t,i} = \exp\left(\frac{-0.5(t-c_i)^2}{2h}\right),$$

which are normalized afterwards by $\psi_{t,i} = \frac{b_{t,i}}{\sum_j b_{t,j}}$ Here c_i denotes the center of the ith basis function. Typically the the centers of the basis functions are spread linearly between 0 and

1, and each basis function has bandwidth given by $h = (c_{i+1} - c_i)^2$. For multi-dimensional systems with D DoFs, the basis function matrix is represented by a block-diagonal matrix, i.e,

$$\mathbf{\Psi}_t = \begin{bmatrix} \boldsymbol{\psi}_t{}^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\ \mathbf{0}^T & \boldsymbol{\psi}_t{}^T & \dots & \mathbf{0}^T \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}^T & \mathbf{0}^T & \mathbf{0}^T & \boldsymbol{\psi}_t{}^T \end{bmatrix}.$$

Due to this encoding of the basis function matrix, the trajectories of all DoFs can still be represented as a single weight vector $\boldsymbol{w}^T = [\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, \dots, \boldsymbol{w}_D^T]$ that is given by a concatenation of all weight vectors for each degree of freedom.

Still, a single weight vector $\boldsymbol{w}$ only represents a single trajectory $\boldsymbol{\tau}$. In order to represent a distribution over trajectories $p(\boldsymbol{\tau})$, we can estimate a distribution $p(\boldsymbol{w})$ over the weight vectors and, subsequently, integrate out the weight vectors. In the original ProMP approach, a multivariate Gaussian distribution is used to model the prior distribution

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w). \tag{3.1}$$

As such, the distribution over trajectories is also Gaussian and can be computed in closed form

$$\begin{aligned} p(\boldsymbol{\tau}) &= \int_w p(\boldsymbol{\tau}|\boldsymbol{w}) p(\boldsymbol{w}) d\boldsymbol{w}, \\ &= \int_w \mathcal{N}\left(\boldsymbol{y}_{1:T} \middle| \mathbf{\Psi}_{1:T}\, \boldsymbol{w}, \beta^{-1} \boldsymbol{I}_S\right) \mathcal{N}\left(\boldsymbol{w} \middle| \boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w\right) d\boldsymbol{w}, \\ &= \mathcal{N}\left(\boldsymbol{y}_{1:T} \middle| \mathbf{\Psi}_{1:T}\, \boldsymbol{\mu}_w, \mathbf{\Psi}_{1:T} \boldsymbol{\Sigma}_w \mathbf{\Psi}_{1:T}^T \beta^{-1} \boldsymbol{I}_S\right), \end{aligned}$$

where $\mathbf{\Psi}_{1:T}$ is a $TD \times DJ$ matrix containing the basis function matrices for all time steps and $\boldsymbol{w}$ is a DJ dimensional column vector. In the following we use the abbreviations $S = TD$ and $d = DJ$ to keep the notation uncluttered.

3.1.1 Learning from Demonstrations with ProMPs

A ProMP already defines a simple hierarchical Bayesian model in a similar fashion as a Bayesian linear regression model. The mean $\boldsymbol{\mu}_w$ and the covariance matrix $\boldsymbol{\Sigma}_w$ can be learned from data by maximum likelihood using the Expectation Maximization (EM) algorithm [7]. A simpler solution that works well in practice is to compute first the most likely estimate of $\boldsymbol{w}^{[i]}$ for each trajectory $\boldsymbol{\tau}^{[i]}$ independently, where the index i denotes the i-th demonstration. Given a trajectory $\boldsymbol{\tau}_i$, the corresponding weight vectors $\boldsymbol{w}^{[i]}$ can be estimated by a straight forward least squares estimate. Subsequently, mean and covariance of $p(\boldsymbol{w})$ can be estimated by the sample mean and sample covariance of the $\boldsymbol{w}^{[i]}$'s. One advantage of the EM based approach in comparison to the more direct approach is that the EM algorithm can also be used for learning from incomplete data where, for instance some segments of the trajectories might be missing due to occlusions in vision based recordings.

However, the training of ProMPs also suffers from a severe disadvantage. As the model has a lot of parameters due to the high-dimensional covariance matrix, ProMPs suffer from overfitting if we have little training data and noisy trajectories. The more sophisticated hierarchical Bayesian model for ProMPs introduced in this thesis alleviates this problem.

3.1.2 Predictions with ProMPs by Conditioning

ProMPs can also be used to predict the behavior of the demonstrator once we have seen an initial part of a new trajectory. Lets assume that we have observed a human demonstrator at $m = 1, 2, ..., M$ different time points. Note that these time points do not need to be sampled in uniform time intervals. t_1 to t_M at the positions $\boldsymbol{y}_{t_1}$ to $\boldsymbol{y}_{t_M}$. Let us further denote $\boldsymbol{\Psi_o}$ as the concatenation of the basis function matrices for these time points and $\boldsymbol{o}$ as concatenation of the $\boldsymbol{y}_{t_m}$ vectors. Given these observations, we can obtain a conditioned distribution $p(\boldsymbol{w}|\boldsymbol{o})$ over the weight vectors as Gaussian distribution with mean and variance

$$\boldsymbol{\mu_{w|o}} = \boldsymbol{\mu_w} + \boldsymbol{\Sigma_w \Psi_o}^T \left(\boldsymbol{\Sigma_o} + \boldsymbol{\Psi_o \Sigma_w \Psi_o}^T\right)^{-1} \left(\boldsymbol{o} - \boldsymbol{\Psi}_o \boldsymbol{\mu_w}\right), \tag{3.2}$$

$$\boldsymbol{\Sigma_{w|o}} = \boldsymbol{\Sigma_w} - \boldsymbol{\Sigma_w \Psi_o}^T \left(\boldsymbol{\Sigma_o} + \boldsymbol{\Psi_o \Sigma_w \Psi_o}^T\right)^{-1} \boldsymbol{\Psi_o \Sigma_w}. \tag{3.3}$$

The conditional distribution $p(\boldsymbol{w}|\boldsymbol{o})$ can be used to predict the behavior of the demonstrator for future time points $t > t_M$, i.e. we can determine the mean and covariance of $\boldsymbol{y}$ for future time points. Note that the same procedure can be applied for partial observations, where only a subset of the quantities in $\boldsymbol{y}_t$ is observed. The covariance matrix $\boldsymbol{\Sigma_o}$ can be used to control the importance of different dimensions. For example the diagonal elements of $\boldsymbol{\Sigma_o}$ might be set to low values for important features.

3.2 Variational inference in latent variable models

In variational inference we consider posterior distributions over unobserved or latent variables Z given some observed variables or data X and also some deterministic parameters θ. Variational inference is a technique for approximating intractable integrals through Bayesian inference.

We start our discussion with the marginal log-likelihood of the observed data, which is given by integrating out the latent random variables for the complete data log likelihood

$$\log p(X|\theta) = \int_Z \log p(X, Z|\theta) dZ. \tag{3.4}$$

Usually it is infeasible to solve this integral in closed form, because the latent variables are not known and in the case of continuous latent random variables there might be exponentially many hidden states so that exact calculation is prohibitively expensive. Therefore we need to resort to approximative variational Bayesian methods. These methods can compute closed form solutions by iteratively updating the latent random variables, similar to the EM-Algorithm [7].

Variational methods avoid computing the integral given in Equation (3.4) by introducing an approximative posterior distribution $q(Z)$ over the latent random variables. The true posterior

distribution $p(Z|X,\theta)$ is infeasible to evaluate or to compute expectations with respect to it to optimize the log-likelihood. The variational posterior $q(Z)$ only is an approximation. The log-likelihood $\log p(X|\theta)$ can be decomposed into a lower bound on the approximative posterior $\mathscr{L}(q,\theta)$ and the Kullback–Leibler divergence (KL)

$$\log p(X|\theta) = \mathscr{L}(q,\theta)\mathrm{KL}(q||p),$$

with

$$\begin{aligned}\mathscr{L}(q,\theta) &= \int_Z q(Z)\log\frac{p(X,Z|\theta)}{q(Z)}, \text{ and} \\ \mathrm{KL}(q||p) &= -\int_Z q(Z)\log\frac{p(Z|X,\theta)}{q(Z)}.\end{aligned} \tag{3.5}$$

We verify this decomposition by first using the product rule $\log p(X,Z|\theta) = \log p(Z|X,\theta) + \log p(X|\theta)$ and substituting this into Equation (3.5),

$$\begin{aligned}\mathscr{L}(q,\theta) &= \int_Z q(Z)\log\frac{p(X,Z|\theta)}{q(Z)}, \\ &= \int_Z q(Z)\log\frac{p(Z|X,\theta)p(X|\theta)}{q(Z)}, \\ &= \int_Z q(Z)(\log\frac{p(Z|X,\theta)}{q(Z)} + \log\frac{p(X|\theta)}{q(Z)}), \\ &= \underbrace{\int_Z q(Z)\log\frac{p(Z|X,\theta)}{q(Z)}}_{KL(q||p)} + \underbrace{\int_Z q(Z)\log\frac{p(X|\theta)}{q(Z)}}_{\log p(X|\theta)}.\end{aligned}$$

Taking into account that $q(Z)$ is a probability distribution and therefore sum to 1, the second term gives exactly the required marginal log-likelihood. The Kullback-Leibler divergence defines a metric on the similarity the true posterior $p(Z|X,\theta)$ and $q(Z)$ such that $KL(q||p) = 0$ if and only if $q(Z) = p(Z|X,\theta)$. This shows that $q(Z)$ is an approximation for the required true posterior of the latent random variables.

Instead of maximizing the log likelihood function we can maximize the lower bound with respect to the variational posterior. For a suitable choice of the variational posterior $q(Z)$ computing the lower bound $\mathscr{L}(q,\theta)$ is tractable. A common choice for $q(Z)$ is a complete factorization of each latent random variable

$$q(Z) = \prod_{i=1}^{M} q_i(Z_i), \tag{3.6}$$

which assumes the latent variables partition into M disjoint groups. Using a factorized variational posterior we can maximize the lower bound with respect to each variational distri-

bution. We now can compute the optimal solution $q_i^*(Z_i)$ by the following general update equation

$$\log q_i^*(Z_i) = < \log p(X, Z|\theta) >_{i \neq j} + const, \tag{3.7}$$

where $< \cdot >_{i \neq j}$ denotes the expectation w.r.t. all remaining variational distributions. Equation (3.7) can be used to compute incremental improvements for all latent variables until convergence to learn our model.

4 Extracting Control Variables with Hierarchical Priors

In this section we will extend the Probabilistic Movement Primitives (ProMPs) with a hierarchical prior distribution on the weight vector $\boldsymbol{w}$. This extension results in a more complex prior distribution which exploits the information shared among multiple different but related tasks by assuming that the weight vectors of the tasks lie on a low dimensional latent manifold. Therefore we refer to the model as *Latent Manifold ProMPs* (LMProMPs). As in Section 3.1 we will demonstrate how we can learn the latent manifold and the latent control variables from demonstrations using variational inference. Finally we show how movements can be predicted with only a small amount of test data. We start our discussion by considering only a single movement type and extend the model afterwards to multiple movement types in a natural fashion by using a mixture model.

4.1 Control Variables for a Single Movement Type

As in the framework of ProMPs we represent a trajectory by

$$p(\boldsymbol{y}_t|\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{y}_t \middle| \boldsymbol{\Psi}_t \boldsymbol{w}, \beta^{-1}\boldsymbol{I}_S\right) \text{ and } p(\boldsymbol{\tau}) = \prod_{t=1}^{T} p\left(\boldsymbol{y}_t \middle| \boldsymbol{w}\right).$$

Since our goal is to model a prior distribution that can be modulated by low dimensional latent control variables we propose the following hierarchical prior

$$p(\boldsymbol{w}^{[i]}) = \mathcal{N}\left(\boldsymbol{w}^{[i]} \middle| \boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, \alpha^{-1}\boldsymbol{I}\right). \tag{4.1}$$

We will shortly discuss this prior distribution and how it differs from the one given in the original ProMP framework. The vector $\boldsymbol{b}$ denotes an *offset term* and together with the *projection matrix* $\boldsymbol{M}$ it defines the mapping from the latent low-dimensional *control variable* $\boldsymbol{h}^{[i]}$ to the *weight vector* $\boldsymbol{w}^{[i]}$ for the trajectory of the i-th demonstration.

The control variable $\boldsymbol{h}^{[i]}$ models the adaptation of the movement to the current task by shifting the prior distribution along the hyperplane $\boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}$. So each task is not only characterized by its weight vector $\boldsymbol{w}^{[i]}$ but also by its control variable $\boldsymbol{h}^{[i]}$. The parameters of this hyperplane $\boldsymbol{b}$ and $\boldsymbol{M}$ are to be learned during training which is shown in Section 4.1.1. This prior distribution makes our model complex enough to model structures like different shapes of trajectories. On the other hand our model is simple enough which prevents from overfitting in high-dimensional parameter spaces. For instance we only use a single precision parameter α instead of a full covariance matrix. This also serves the goal of learning from only a few trajectories since more trajectories would be needed to learn a more complex model by fitting a full covariance matrix.

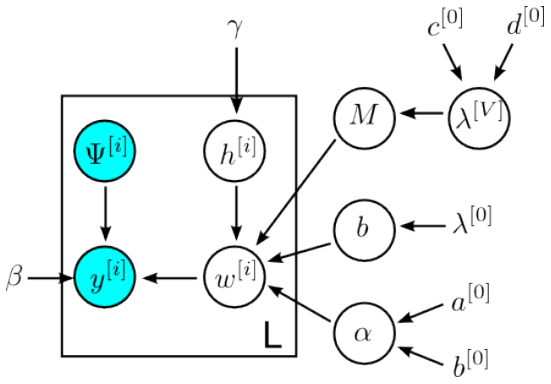

Figure 4.1.: The Latent Manifold ProMP Model used for single movement types. The original ProMP is extended by the offset vector $\boldsymbol{b}$, the projection matrix $\boldsymbol{M}$ and the control variable $\boldsymbol{h}^{[i]}$. The covariance is simplified to a single precision parameter α. All model parameters then are given hyper-priors as we follow a fully Bayesian approach.

If we set the control variable $\boldsymbol{h}^{[i]}$ to zero the mean of the hierarchical prior would simplify to $\boldsymbol{b}$ which corresponds to the mean $\boldsymbol{\mu}_w$ of the original formulation of ProMPs. So we basically extended the mean of the prior distribution of the weight vectors by introducing control variables and simplify the covariance of the prior distribution to prevent from overfitting. The combination of the latent variable $\boldsymbol{h}^{[i]}$ and the projection matrix $\boldsymbol{M}$ implements a more accurate model of the prior distribution. As we will demonstrate in Section 5, this hierarchical prior model is less sensitive to overfitting in the case of noisy observations or incomplete data.

All parameters of the model are unknown a priori. We follow a fully Bayesian approach, where we treat all parameters as random variables and introduce conjugate prior distributions for each of them

$$
\begin{aligned}
p(\boldsymbol{b}|\lambda^{[0]}) &= \mathcal{N}\left(\boldsymbol{b}\middle|\boldsymbol{0}, (\lambda^{[0]})^{-1}\boldsymbol{I}_d\right), \\
p(\boldsymbol{M}|\lambda^{[1:V]}) &= \prod_{v=1}^{V} \mathcal{N}\left(\boldsymbol{m}^{[v]}\middle|\boldsymbol{0}, (\lambda^{[v]})^{-1}\boldsymbol{I}_d\right), \\
p(\boldsymbol{h}^{[i]}|\gamma) &= \mathcal{N}\left(\boldsymbol{h}^{[i]}\middle|\boldsymbol{0}, (\gamma)^{-1}\boldsymbol{I}_V\right), \\
p(\alpha|a^{[0]}, b^{[0]}) &= \Gamma\left(\alpha\middle|a^{[0]}, b^{[0]}\right).
\end{aligned}
$$

Here $\boldsymbol{m}^{[v]}$ denotes the v-th column of the projection matrix $\boldsymbol{M} = [\boldsymbol{m}^{[1]}, \boldsymbol{m}^{[2]}, \ldots, \boldsymbol{m}^{[V]}]$ where V denotes the dimensionality of the latent control variable $\boldsymbol{h}^{[i]}$. The symbol Γ denotes the Gamma distribution. The graphical representation of our model is shown in Figure 4.1.

To enhance the numerical stability of the variational updates we also place an additional Gamma prior on the precision parameters of the columns of each of the projection matrix

$$
p(\lambda^{[v]}|c^{[0]}, d^{[0]}) = \Gamma\left(\lambda^{[v]}\middle|c^{[0]}, d^{[0]}\right).
$$

We evaluate the influence of this hyper-prior in Section 5. The technique to place a prior distribution on each column of the projection matrix instead of using on prior for the whole matrix is also used in Bayesian PCA. During the updates of the precision parameter $\lambda^{[v]}$ some

of them may be driven to infinity with its corresponding column vector $\boldsymbol{m}^{[\nu]}$ converging to zero. For the special case $c^{[0]} = 1$, this prior distribution approximates a Laplace prior which also is more peaked around its mean which also favor sparse solutions. This results in sparser solutions of the projection matrix $\boldsymbol{M}$ which are easier to analyze, since less columns are active. We consider all parameters of the hierarchical prior as latent random variables

$$Z = \{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}\},$$

while we treat the hyper-parameters as deterministic parameters

$$\theta = \{\beta, \lambda^{[0]}, \gamma, a^{[0]}, b^{[0]}, c^{[0]}, d^{[0]}\},$$

where L denotes the total number of demonstrations. We use variational inference to learn the posterior distribution of the latent random variables given some demonstrations in Section 4.1.1. The parameters can be optimized using Bayesian optimization or simply can be tuned by hand which worked well in our experiments. In our experiments we simplified the model by setting $\lambda^{[0]}$ and γ to one, since their impact on the posterior distributions we are interested in is infinitesimal small. $a^{[0]}$ and $b^{[0]}$ are set to small numbers like 10^{-5} to model an uninformative prior. With this we obtain a model with only the precision parameter β to tune. This parameter induces how much we trust the given data. More details on this is given in Section 5.3.

4.1.1 Learning from demonstrations by variational inference

For our model we obtain the following complete data likelihood which is used to compute the variational inference updates

$$\begin{aligned} &p(\boldsymbol{y}^{[1:L]}, \boldsymbol{\Psi}^{[1:L]}, \boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \lambda^{[1:V]}, \alpha), \\ &= \prod_{i=1}^{L} \{p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}) p(\boldsymbol{w}^{[i]} | \boldsymbol{b}, \boldsymbol{M}, \boldsymbol{h}^{[i]}) p(\boldsymbol{h}^{[i]})\} p(\alpha) p(\boldsymbol{b}) p(\boldsymbol{M} | \lambda^{[1:V]}) p(\lambda^{[1:V]}). \end{aligned} \tag{4.2}$$

As in Section 3.2 we also use a complete factorization of the variational posterior distribution given by

$$q(\boldsymbol{Z}) = q(\boldsymbol{b}) \prod_{\nu=1}^{V} \{q(\boldsymbol{m}^{[\nu]}) q(\lambda^{[1:V]})\} q(\alpha) \prod_{i=1}^{L} q(\boldsymbol{w}^{[i]}) q(\boldsymbol{h}^{[i]}). \tag{4.3}$$

Since we use conjugate prior distributions the variational posterior distributions reads

$$\begin{aligned} q(\boldsymbol{w}^{[i]}) &:= \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}), \\ q(\boldsymbol{h}^{[i]}) &:= \mathcal{N}(\boldsymbol{h}^{[i]} | \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}), \\ q(\boldsymbol{b}) &:= \mathcal{N}(\boldsymbol{b} | \boldsymbol{\mu}_{\boldsymbol{b}}, \sigma_{\boldsymbol{b}} \boldsymbol{I}), \\ q(\boldsymbol{m}^{[\nu]}) &:= \mathcal{N}(\boldsymbol{m}^{[\nu]} | \boldsymbol{\mu}_{\boldsymbol{m}^{[\nu]}}, \sigma_{\boldsymbol{m}^{[\nu]}} \boldsymbol{I}), \end{aligned}$$

$$
\begin{aligned}
q(\alpha) :=& \Gamma(\alpha|\bar{a}, \bar{b}), \text{ and} \\
q(\lambda^{[\nu]}) :=& \Gamma(\lambda^{[\nu]}|\bar{c}, \bar{d}).
\end{aligned}
$$

It is worth noting that the variational posterior distribution of $\boldsymbol{h}^{[i]}$ contains a full covariance matrix even though its prior only has a single precision parameter. The reason for this is that the different dimensions of the control variable are not assumed to be independent. So we learn a correlation between the different tasks we consider.

We now can compute each posterior distribution using the general update in Equation (3.7) by computing the expected value of the complete data log-likelihood function w.r.t to each remaining latent random variable. The solutions read

$$
\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} =& \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}(\beta \boldsymbol{\Psi}^{[i]^T} \boldsymbol{y}^{[i]} + \bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})), \\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} =& (\beta \boldsymbol{\Psi}^{[i]^T} \boldsymbol{\Psi}^{[i]} + \bar{\alpha} \boldsymbol{I}_d)^{-1}, \\
\boldsymbol{\mu}_{\boldsymbol{b}} =& \sigma_{\boldsymbol{b}}(\sum_{i=1}^{L} \{\bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})\}), \\
\sigma_{\boldsymbol{b}} =& (\sum_{i=1}^{L} \{\bar{\alpha}\} + \lambda^{[0]})^{-1} = (L\bar{\alpha} + \lambda^{[0]})^{-1}, \\
\boldsymbol{\mu}_{\boldsymbol{m}^{[\nu]}} =& \sigma_{\boldsymbol{m}^{[\nu]}}(\sum_{i=1}^{L} \{\bar{\alpha} \mu_{h^{[\nu,i]}}(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}})\}), \\
\sigma_{\boldsymbol{m}^{[\nu]}} =& (\sum_{i=1}^{L} \{\bar{\alpha}((\mu_{h^{[\nu,i]}})^2 + \sigma_{h^{[\nu,i]}})\} + \bar{\lambda}^{[\nu]})^{-1} = (\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}(\nu, \nu))^{-1}, \\
\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} =& \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}(\bar{\alpha} \boldsymbol{I}_V \bar{\boldsymbol{M}}^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}})), \\
\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}} =& (\bar{\alpha} \boldsymbol{I}_V (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) + \gamma \boldsymbol{I}_V)^{-1}, \\
\bar{c} =& c^{[0]} + \frac{d}{2}, \\
\bar{d} =& d^{[0]} + \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{m}^{[\nu]}}{}^T \boldsymbol{\mu}_{\boldsymbol{m}^{[\nu]}} + d\sigma_{\boldsymbol{m}^{[\nu]}}), \\
\bar{a} =& a^{[0]} + \frac{dL}{2}, \\
\bar{b} =& b^{[0]} + \frac{1}{2} \sum_{i=1}^{L} \{C + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + d\sigma_{\boldsymbol{b}} + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[Q],
\end{aligned}
$$

with

$$
\begin{aligned}
\bar{\boldsymbol{M}} =& [\boldsymbol{\mu}_{\boldsymbol{m}^{[1]}}, \ldots, \boldsymbol{\mu}_{\boldsymbol{m}^{[V]}}], \\
Q =& (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}, \text{ and} \\
C =& (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}).
\end{aligned}
$$

We also combine the precision parameters $\sigma_{\boldsymbol{m}^{[\nu]}}$ of each of the projection vectors $\boldsymbol{m}^{[\nu]}$ into one matrix

$$\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}} = \begin{bmatrix} \sigma_{\boldsymbol{m}^{[1]}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\boldsymbol{m}^{[V]}} \end{bmatrix}.$$

The inferred feature precision is denoted by $\bar{\alpha}$ and the scalar $\mu_{h^{[\nu,i]}}$ denotes the ν-th element in the vector $\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} = [\mu_{h^{[1,i]}}, \ldots, \mu_{h^{[V,i]}}]^T$. The derivations of the update equations are given in Appendix B.1.

By iteratively updating these equations they converge to a (local) optimum for the given data. As for the EM-algorithm the initialization is crucial, since the iterative updates might get stuck in local optima. Our method is closely related to Principle Component Analysis (PCA) as each dimension of the latent manifold can be seen as a principle component in parameter space. However we are not able to perform PCA directly on the given data, since the principle components must be determined in parameter space but only the trajectories in task space are given as observed data. Nevertheless we could use PCA to initialize our model properly after we obtained the required weight vectors by linear regression. This gives a good initial solution for our model, as shown in the results section.

4.1.2 Predictions by Conditioning the Hierarchical Prior

For the original formulation of ProMPs conditioning was performed directly on the weight vector to obtain a new Gaussian distribution for some observations $\boldsymbol{o}$. However with the LMProMPs the task does not only depend on the weight vector $\boldsymbol{w}$ but also on the latent control variable $\boldsymbol{h}$. We therefore condition typically on the lower-dimensional control variable and integrate out the weight vector, which then gives us the conditional distribution over the weight vector given the observed data. Note that conditioning directly on the weight vector is not possible, since the weight vector depends on the control variable. The conditional distribution over the control variables $\boldsymbol{h}$ given the observed data can be determined by first using Bayes rule and integrating out the weight vector $\boldsymbol{w}$ and perform marginalization and conditioning afterwards:

$$\begin{aligned}
p(\boldsymbol{h}|\boldsymbol{o}) \propto & p(\boldsymbol{o}|\boldsymbol{h})p(\boldsymbol{h}), \\
= & \int_{\boldsymbol{w}} p(\boldsymbol{o}|\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{h})d\boldsymbol{w}\, p(\boldsymbol{h}), \\
= & \int_{\boldsymbol{w}} \mathcal{N}\left(\boldsymbol{o}|\boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{w}, \boldsymbol{\Sigma}_{\boldsymbol{o}}\right)\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}, (\alpha)^{-1}\boldsymbol{I}_d)d\boldsymbol{w}\,\mathcal{N}\left(\boldsymbol{h}|\boldsymbol{0}, \gamma\boldsymbol{I}_V\right), \\
= & \mathcal{N}\left(\boldsymbol{o}|\boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}), \boldsymbol{\Sigma}_{\boldsymbol{o}} + \boldsymbol{\Psi}_{\boldsymbol{o}}(\alpha)^{-1}\boldsymbol{I}_d{\boldsymbol{\Psi}_{\boldsymbol{o}}}^T)\mathcal{N}(h|\boldsymbol{0}, \gamma\boldsymbol{I}_V\right), \\
= & \mathcal{N}\left(\boldsymbol{h}|\boldsymbol{0}, \gamma\boldsymbol{I}_V\right)\mathcal{N}(\boldsymbol{o}|\boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}), \boldsymbol{\Sigma}_{\boldsymbol{o}} + \boldsymbol{\Psi}_{\boldsymbol{o}}(\alpha)^{-1}\boldsymbol{I}_d{\boldsymbol{\Psi}_{\boldsymbol{o}}}^T\right), \\
= & \mathcal{N}\left(\begin{array}{c|cc} \boldsymbol{h} & \boldsymbol{0} & \gamma\boldsymbol{I}_V \quad \gamma\boldsymbol{I}_V^T\bar{\boldsymbol{M}}^T{\boldsymbol{\Psi}_{\boldsymbol{o}}}^T \\ \boldsymbol{o} & \boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}) & \boldsymbol{\Psi}_{\boldsymbol{o}}\bar{\boldsymbol{M}}\gamma\boldsymbol{I}_V \quad \boldsymbol{\Sigma}_{\boldsymbol{o}} + \boldsymbol{\Psi}_{\boldsymbol{o}}(\alpha^{-1}\boldsymbol{I}_d){\boldsymbol{\Psi}_{\boldsymbol{o}}}^T \end{array}\right), \\
= & \mathcal{N}\left(\begin{array}{c|cc} \boldsymbol{o} & \boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}) & A \quad \boldsymbol{\Psi}_{\boldsymbol{o}}\bar{\boldsymbol{M}}\gamma\boldsymbol{I}_V \\ \boldsymbol{h} & \boldsymbol{0} & \gamma\boldsymbol{I}_V^T\bar{\boldsymbol{M}}^T{\boldsymbol{\Psi}_{\boldsymbol{o}}}^T \quad \gamma\boldsymbol{I}_V^T \end{array}\right), \\
= & \mathcal{N}(\boldsymbol{o}|\boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}), A),
\end{aligned}$$

$$\mathcal{N}(\boldsymbol{h}|\mathbf{0}+\gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T A^{-1}(\boldsymbol{o}-\boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}}+\bar{\boldsymbol{M}}\mathbf{0}), \gamma \boldsymbol{I}_V - \gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T A^{-1} \boldsymbol{\Psi}_{\boldsymbol{o}} \bar{\boldsymbol{M}} \gamma \boldsymbol{I}_V),$$

where

$$A = \boldsymbol{\Sigma}_{\boldsymbol{o}} + \boldsymbol{\Psi}_{\boldsymbol{o}}(\alpha^{-1}\boldsymbol{I}_d + \bar{\boldsymbol{M}} \gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T) {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T.$$

So we obtain the conditioned distribution of the control variable $\boldsymbol{h}$ given the observed data $\boldsymbol{o}$, with its mean and covariance

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}} =& \mathbf{0} + \gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T A^{-1}(\boldsymbol{o} - \boldsymbol{\Psi}_{\boldsymbol{o}}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\mathbf{0}), \\
=& \gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T A^{-1}(\boldsymbol{o} - \boldsymbol{\Psi}_{\boldsymbol{o}} \boldsymbol{\mu}_{\boldsymbol{b}}), \qquad (4.4) \\
\boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}} =& \gamma \boldsymbol{I}_V - \gamma \boldsymbol{I}_V^T \bar{\boldsymbol{M}}^T {\boldsymbol{\Psi}_{\boldsymbol{o}}}^T A^{-1} \boldsymbol{\Psi}_{\boldsymbol{o}} \bar{\boldsymbol{M}} \gamma \boldsymbol{I}_V. \qquad (4.5)
\end{aligned}$$

Given the distribution over the latent control variables we are able to compute the distribution over the weight vectors by integrating out the control variables

$$\begin{aligned}
p(\boldsymbol{w}) =& \int_{\boldsymbol{h}} p(\boldsymbol{w}|\boldsymbol{h}) p(\boldsymbol{h}), \\
=& \int_{\boldsymbol{h}} \mathcal{N}\left(\boldsymbol{w}|\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}\right) \mathcal{N}(\boldsymbol{h}|\boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}}, \boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}}) \, d\boldsymbol{h}, \\
=& \mathcal{N}\left(\boldsymbol{w}|\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}}, (\alpha)^{-1}\boldsymbol{I}_d + \bar{\boldsymbol{M}} \boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}} \bar{\boldsymbol{M}}^T\right).
\end{aligned}$$

As such the conditional distribution over the weight vectors is also Gaussian with mean and variance

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}} =& \boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}}, \qquad (4.6) \\
\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}} =& (\alpha)^{-1}\boldsymbol{I}_d + \bar{\boldsymbol{M}} \boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}} \bar{\boldsymbol{M}}^T. \qquad (4.7)
\end{aligned}$$

It is illustrative to investigate the differences of the standard conditioning of the ProMPs in Equation (3.2) and Equation (3.3) to the conditioning with the hierarchical prior. The conditioning in the ProMP case requires a full-rank covariance matrix, which is hard to obtain given a small amount of training data. In contrast, the latent prior model only requires the projection matrix $\bar{\boldsymbol{M}}$ to perform the conditioning. Hence, the predictions of the latent prior model are less prone to overfitting and are therefore also applicable for a small amount of training data.

4.2 Extension to Multiple Movement Types

Until now we only have considered a single movement type. Since we want to learn multiple different tasks we now extend the proposed model to cope with multiple types of movements. If we inspect the parameter space of the weight vectors we see that different movement types often build clusters. Note that the LMProMP for a single movement type only uses a single Gaussian distribution as prior of the weight vectors, which is not able to model multiple clusters properly. The weight vectors $\boldsymbol{w}$ also might lie on a *non*-linear manifold which we cannot model with a single Gaussian prior distribution. An example of these different types

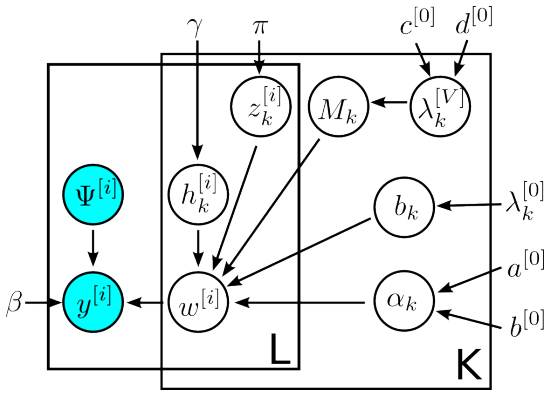

Figure 4.2.: The Latent Manifold ProMP Model used for multiple movement types. The model for a single movement types is extended with a mixture model. Additionally we introduce an multinomial variable $z^{[i]}$ which indicates to which mixture component a demonstration belongs.

of movements might by forehand and backhand strokes in a table tennis game. To model such nonlinearities we extend our prior distribution to a standard mixture model

$$p(\boldsymbol{w}^{[i]}) = \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{w}^{[i]} \middle| \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, \alpha_k^{-1} \boldsymbol{I}\right). \tag{4.8}$$

The extended model has K mixture components, each one having the same prior distribution as in Equation (4.1), which can model a single movement type. The vector $\boldsymbol{b}_k$ in Equation (4.8) denotes an offset term of component k and the projection matrix $\boldsymbol{M}_k$ defines the mapping from the low-dimensional control variables $\boldsymbol{h}_k^{[i]}$ to the weight vector $\boldsymbol{w}^{[i]}$, while the parameter α_k models the precision of each component of the proposed prior distribution. Additionally π_k denotes the mixing coefficients. We also add a multinomial variable to our probabilistic model, i.e. $z_k^{[i]} \in \{0,1\}$. We represent this multinomial variable as binary vector $\boldsymbol{z}^{[i]} = \{z_1^{[i]}, ..., z_K^{[i]}\}$, which indicates to which mixture component trajectory i belongs. We treat this mixing indices also as latent variable and therefore introduce a multinomial hyper-prior

$$p(\boldsymbol{z}) = \prod_{i=1}^{L} \prod_{k=1}^{K} (\pi_k)^{z_k^{[i]}}.$$

The extended graphical model containing the mixture model is shown in Figure 4.2.

4.2.1 Learning from demonstrations for mulitple movements

As for a single movement type we use variational inference to learn the latent variables from demonstrations. Given our model we obtain the following complete data likelihood function

$$\begin{aligned} &p(\boldsymbol{y}^{[1:L]}, \boldsymbol{\Psi}^{[1:L]}, \boldsymbol{w}^{[1:L]}, \boldsymbol{h}_{1:K}^{[1:L]}, z_{1:K}^{[1:L]}, \boldsymbol{b}_{1:K}, \boldsymbol{M}_{1:K}, \alpha_{1:K}, \lambda_{1:K}^{[1:V]}), \\ &\quad = \prod_{i=1}^{L} \{ p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}, \beta) \prod_{k=1}^{K} p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) p(\boldsymbol{h}_k^{[i]} | \gamma) \} \} \end{aligned}$$

$$\prod_{k=1}^{K}\{p(\boldsymbol{b}_k|\lambda_k^{[0]})p(\boldsymbol{M}_k|\lambda_k^{[1:V]})p(\alpha_k|a_k^{[0]},b_k^{[0]})p(\lambda_k^{[1:V]}|c_k^{[0]},d_k^{[0]})\}p(\boldsymbol{z}|\pi).$$

We again use a complete factorization of the variational posterior distribution over the latent variables Z

$$q(\boldsymbol{Z}) = \prod_{k=1}^{K}\{q(\alpha_k)q(\boldsymbol{b}_k)\prod_{v=1}^{V}\{q(\boldsymbol{m}_k^{[v]})q(\lambda_k^{[v]})\}\prod_{i=1}^{L}q(\boldsymbol{h}_k^{[i]})\}\prod_{i=1}^{L}\{q(\boldsymbol{w}^{[i]})\}q(\boldsymbol{z}).$$

The variational distributions are the same as for Equation (4.3). Additionally we define the variational distribution of the mixing indices as

$$q(\boldsymbol{z}) = \prod_{i=1}^{L}\prod_{k=1}^{K}(\mu_{z_k^{[i]}})^{z_k^{[i]}}.$$

The variational updates are the same as for the case with only a single component, with the difference that the trajectories are weighted by the responsibilities of the individual mixture components $\mu_{z_k^{[i]}}$,

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} &= \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}\left(\beta {\boldsymbol{\Psi}_{1:T}^{[i]}}^T \boldsymbol{y}_{1:T}^{[i]} + \sum_{k=1}^{K}\bar{\alpha}_k\mu_{z_k^{[i]}}\left(\boldsymbol{\mu}_{\boldsymbol{b}_k} + \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}\right)\right),\\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} &= \left(\beta {\boldsymbol{\Psi}_{1:T}^{[i]}}^T \boldsymbol{\Psi}_{1:T}^{[i]} + \sum_{k=1}^{K}\bar{\alpha}_k\mu_{z_k^{[i]}}\boldsymbol{I}\right)^{-1},\\
\boldsymbol{\mu}_{\boldsymbol{b}_k} &= \sigma_{\boldsymbol{b}_k}\left(\sum_{i=1}^{L}\{\mu_{z_k^{[i]}}\bar{\alpha}_k(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})\}\right),\\
\sigma_{\boldsymbol{b}_k} &= \left(\sum_{i=1}^{L}\{\mu_{z_k^{[i]}}\bar{\alpha}_k\} + \lambda_k^{[0]}\right)^{-1},\\
\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} &= \sigma_{\boldsymbol{m}_k^{[v]}}\left(\sum_{i=1}^{L}\{\mu_{z_k^{[i]}}\bar{\alpha}_k(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k})\mu_{h_k^{[v,i]}}\}\right),\\
\sigma_{\boldsymbol{m}_k^{[v]}} &= \left(\sum_{i=1}^{L}\{\mu_{z_k^{[i]}}\bar{\alpha}_k((\mu_{h_k^{[v,i]}})^2 + \sigma_{h_k^{[v,i]}})\} + \bar{\lambda}_k^{[v]}\right)^{-1} = (\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}(v,v))^{-1},\\
\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} &= \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}\left(\sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k\bar{\boldsymbol{M}}_k^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k})\right),\\
\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}} &= \left(\sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k(\bar{\boldsymbol{M}}_k^T\bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}) + \gamma_k\boldsymbol{I}_V\right)^{-1},\\
\bar{c}_k &= c_k^{[0]} + \frac{d}{2},\\
\bar{d}_k &= d_k^{[0]} + \frac{1}{2}({\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}}^T\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + d\sigma_{\boldsymbol{m}_k^{[v]}}),\\
\bar{a}_k &= a_k^{[0]} + \frac{d}{2}\sum_{i=1}^{L}\{\mu_{z_k^{[i]}}\},
\end{aligned}$$

$$
\begin{aligned}
\bar{b}_k &= b_k^{[0]} + \frac{1}{2}\sum_{i=1}^{L}\{C + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[Q], \\
\mu_{z_k^{[i]}} &= \frac{\rho_k^{[i]}}{\sum_{j=1}^{K}\rho_j^{[i]}}, \\
\rho_k^{[i]} &= \exp(\log\pi_k + \frac{d}{2}(\mathcal{K}(\bar{a}_k) - \log\bar{b}_k) - \frac{\bar{\alpha}_k}{2} \\
&\quad [C + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[Q]],
\end{aligned}
$$

with

$$
\begin{aligned}
\bar{\boldsymbol{M}}_k &= [\boldsymbol{\mu}_{\boldsymbol{m}_k^{[1]}}, \ldots, \boldsymbol{\mu}_{\boldsymbol{m}_k^{[V]}}], \\
Q &= (\bar{\boldsymbol{M}}_k^T d\bar{\boldsymbol{M}}_k + \boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}, \text{ and} \\
C &= (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\mu_{h_k^{[v,i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\mu_{h_k^{[v,i]}}).
\end{aligned}
$$

The symbol $\mathcal{K}$ denotes the digamma function, while the remaining notation is the same as in Section 4.1.1. The derivations of the update equations are given in Appendix B.2.

Again the initialization of the iterative update schema is crucial. We applied the same initialization from the model for a single movement type. Additionally, we perform k-means clustering on the weight vectors obtained by linear regression to determine to which mixture component an approximated weight vector belongs. Finally, for each component we perform PCA with the weight vectors assigned to the specific cluster. The problem with *K-means* is that the number of clusters need to be pre-specified. Taking inspiration from [3] we initialize the model with a large number of clusters. Similar to the projection matrix $\boldsymbol{M}$ some of the components only provide insufficient contribution to explaining the data and therefore their mixing coefficients π_k will be driven to zero during training. Such components whose mixing coefficients fall below a certain threshold in an iteration are removed. We therefore obtain an approximately optimal number of mixture components without the need of expensive techniques like cross-validation.

4.2.2 Predictions for multiple movements

Computing predictions with the mixture model is also straight forward. For each component we compute the conditioned distribution on the latent control variables $p^{[k]}(\boldsymbol{h}|\boldsymbol{o})$ using Equation (4.4) and Equation (4.5).The posterior over the weight vectors $p^{[k]}(\boldsymbol{w}|\boldsymbol{o})$ are computed using Equation (4.6) and Equation (4.7). Thereafter the posterior distributions are weighted by the responsibilities $z^{[k]}$ of each mixture component

$$
\begin{aligned}
z^{[k]} &= \frac{\pi_k\mathcal{N}\left(\boldsymbol{o}\middle|\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]}, \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]}\right)}{\sum_{j=1}^{K}\pi_j\mathcal{N}\left(\boldsymbol{o}\middle|\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}}^{[j]}, \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[j]}\right)}, \\
\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}} &= \sum_{k=1}^{K} z^{[k]}\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]},
\end{aligned}
$$

$$\mu_{w|o} = \sum_{k=1}^{K} z^{[k]} \mu_{w|o}^{[k]}.$$

We could also use the component with the maximal responsibility instead of our proposed method. But as we want to do predictions for unseen tasks weighting the different components is a more appropriate approach and as shown in the result section the responsibilities for one component are dominating if we want to predict task we have already seen in training.

5 Results

First we evaluate our model on a synthetic dataset and inspect the prior distributions of the ProMPs and of LMProMPs This gives some insight on how the different proposed prior distributions model the data. We afterwards evaluate our method on two real robot and one human task. In the first task the robot played a table tennis game and we use the Cartesian coordinates of a racket mounted at its end-effector and the Cartesian coordinates of the ball. A Barrett WAM anthropomorphic arm was used for this experiment [19]. The robot provides regular updates about its joint positions at a rate of 1KHz that are used by the forward kinematics to compute the Cartesian position of the racket. The ball is tracked by a high-speed, multi-camera vision system [17] that provides updates at a rate of 200Hz. The extracted dataset contains twenty ball and racket trajectories shown in Figure 5.1(B).

In the second task we placed an obstacle in front of a KUKA lightweight arm and demonstrated by kinesthetic teaching different ways to approach a desired target point in Cartesian space. During the demonstrations we avoided hitting the obstacle and we bypassed it either by moving to the left or to the right. The demonstrations are depicted in Figure 5.2. For this experiment we recored the Cartesian position and orientation of the end-effector. The state vector $\boldsymbol{y}_t$ for this experiment is seven dimensional, three dimensions for the position and four for the quaternion based orientation.

In the human task we use visual markers from a motion capturing system from humans playing golf. We especially focus on the three Cartesian coordinates of the *Club P* marker, which is placed at the top of the shaft. The extracted dataset contains 30 swings of novices and 30 swings of experts. We evaluate the differences between novices and experts in the latent control space. This experiment only contains first results as it primarily shows how the model can be used to analyze the data in different setups and can be used in other disciplines like sport science.

5.1 Comparing the proposed prior distributions

We consider a synthetic dataset with four different movement types, shown in Figure 5.3. We created two-dimensional weight vectors $\boldsymbol{w}^{[i]}$ that are sampled on of the four clusters. We learn the prior distribution for the original ProMP as well as for our proposed LMProMP model assuming a single and multiple movement types. Afterwards we plot the parameter space of the weight vectors as well as the prior distributions. One can see that the prior of the original ProMPs averages over all tasks. If we want to generate trajectories by sampling weight vectors from the prior distribution in most cases we would obtain trajectories which are quite different from the learned tasks since the mean of the prior distribution was not demonstrated during training. Also it has a high variance denoted by the black ellipse which also would result in trajectories which differ from the training set. The LMProMP model for a single movement type has already a tighter prior distribution. In fact it is a good approximation for the variance of one of the four clusters. We use a two-dimensional latent control variable drawn as red lines, which shifts the prior distribution along the hyperplane in the

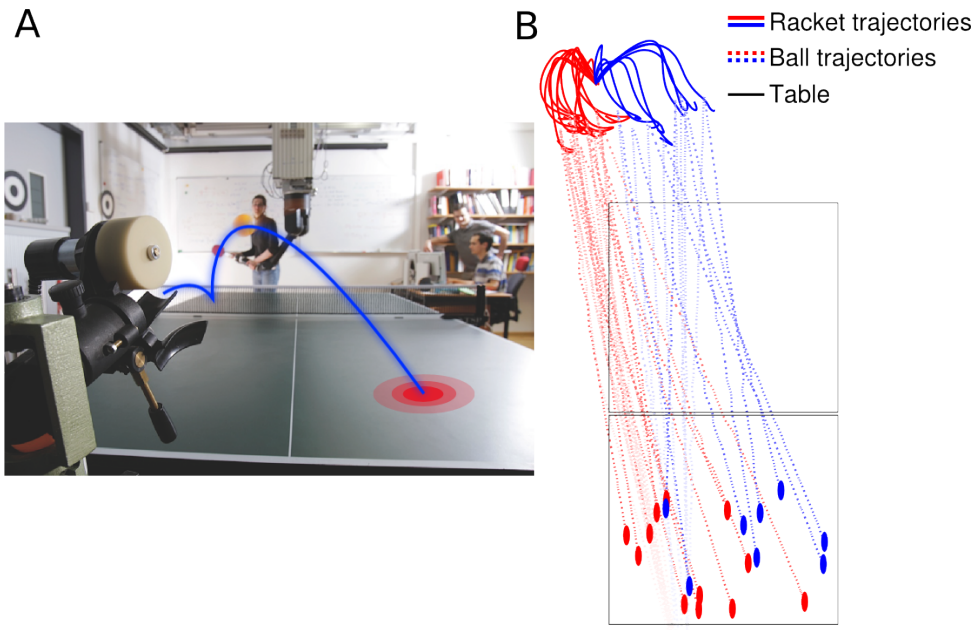

Figure 5.1.: Ping-Pong: (A) Trajectory prediction task on a table tennis dataset. The data consists of 20 end-effector and ball trajectories illustrated in (B). We thank Katharina Muelling and Axel Griesch (photographer) for providing the picture and the table tennis data.

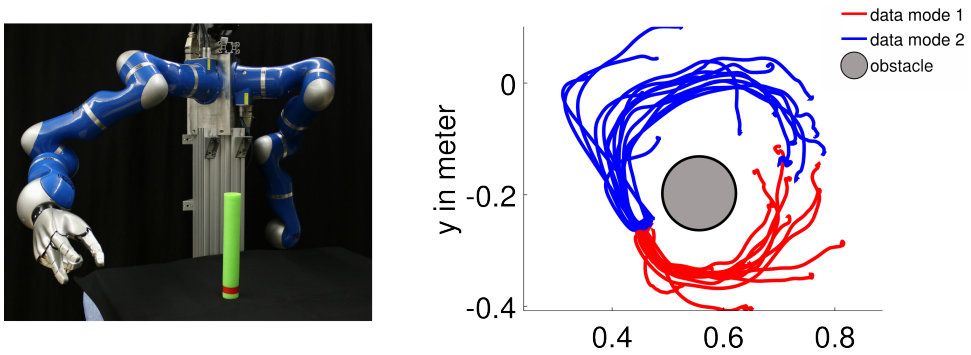

Figure 5.2.: Bi-Modal: Experimental setting and two dimensions out of the 7-dimensional dataset (three end-effector coordinates and the four dimensional quaternions). We thank Guilherme Maeda, Rudolf Lioutikov and Marco Ewerton for providing the data.

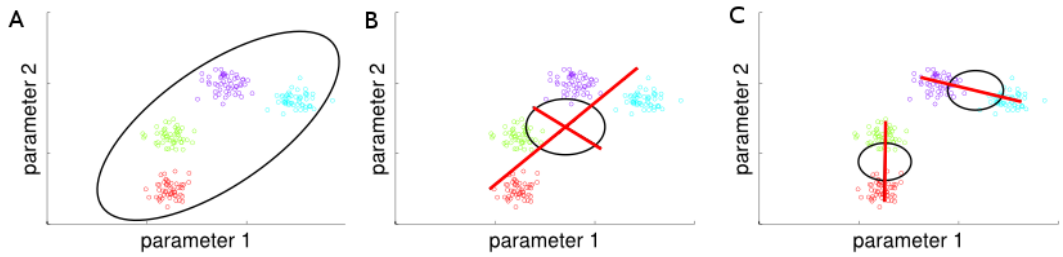

Figure 5.3.: Synthetic: The parameter space for a synthetic dataset with four different movement types denoted by the different colored circles. (A) The prior of the original ProMP denoted by the black ellipse averages over all task clusters. (B) The prior of the LMProMPs assuming only a single movement type and also averages over the data. The main difference is that the prior can be shifted along the 2-dimensional latent control variable $\boldsymbol{h}$ denoted by the red lines. (C) The prior of the LMProMPs for multiple movements is more accurate since two mixture components are used. The covariance is also very similar to the variance of the different movement types, which makes the model even more accurate.

parameter space. One can generate appropriate trajectories which are similar to the training data if the corresponding value of the two-dimensional latent control variable is given. However it might be very hard to obtain this value of the control variable. The LMProMP for multiple movements avoids this problem by introducing multiple mixture components. In this setup we initialized the model with two mixture components. Getting the value of the control variable to generate trajectories for one of the movement types is easier, since we first can distinguish between the mixture components and afterwards choose the value of the latent control variable. One also can see that the nonlinear latent manifold of the four different movement types is approximated best with the LMProMP model for multiple movement types.

5.2 The effect of noise and missing data

We use the table tennis setup to predict the final impact location of the ball at the opponent's court. We evaluate our prediction by computing the Euclidean distance in the x,y-plane to the true impact location. The dataset used for learning is shown in Figure 5.1. It should be noted that the colors (red and blue) in Figure 5.1 are only used for the visualization as no labels were used for modeling the data. For a baseline comparison we trained the ProMPs on the same data. The learned distributions over trajectories for ProMPs are illustrated for three Cartesian coordinates of the racket and the ball in Figure 5.4. We denote the mean of the trajectory distribution with a solid black line and the standard deviation by the shaded region.

In the collected dataset, the robot returns the ball within 550ms to 650ms in advance to the final ball impact. In our comparison, we analyze the prediction performance with respect to the time until the impact event, where we focus on the movement phase right after the stroke, $\approx$ 625ms before the end. We used leave-one-out cross-validation to compute the test error.

A fast multi-camera vision setup, good lighting conditions, and access to the opponents sensor readings are amenities we can not always afford. Therefore, we simulate the effect

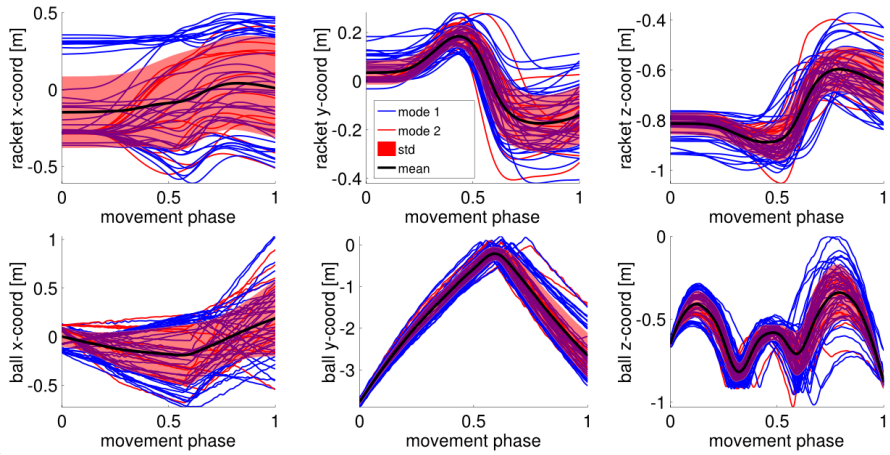

Figure 5.4.: Ping-Pong: Learned distributions over trajectories for all six dimensions using ProMPs. The six dimensions contains the 3 Cartesian coordinates of the racket as well as the ball.

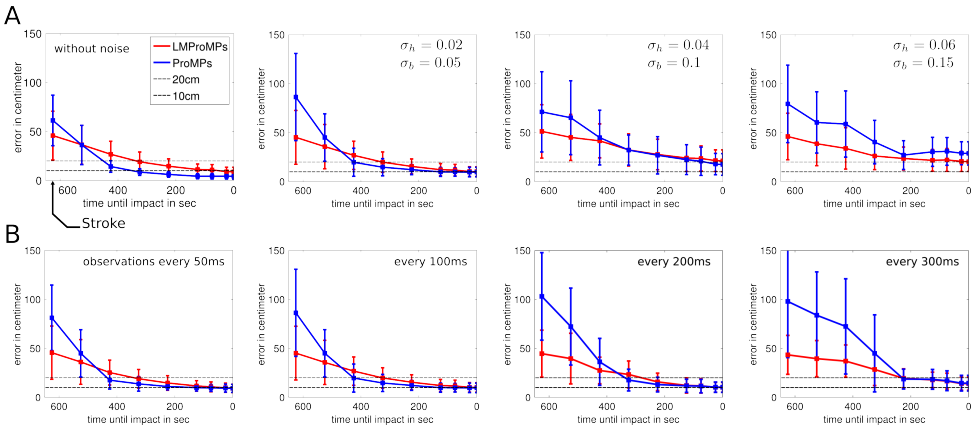

Figure 5.5.: Ping-Pong: The effect of noise (A) and missing data (B) on the prediction performance of ProMPs and LM-ProMPs. In (A), from left to right the amount of applied noise to the data is increased. In (B) four different frame rates of observations ($\in \{50, 100, 200, \text{and } 300\}$ms) are investigated.

of noisy and incomplete observations, and we evaluate their impact on the prediction performance. First, we add zero-mean Gaussian observation noise to the Cartesian coordinates of the racket and to the Cartesian coordinates of the ball. The standard deviation of the noise used in our evaluation is $\sigma_h \in 10^{-2}\{0, 2, 4, 6\}$ and $\sigma_b \in 10^{-2}\{0, 5, 10, 15\}$ for the racket and the ball, respectively. The results are illustrated in Figure 5.5(A), where we show the advantage of the learned prior distribution using latent variables. Additionally, we evaluate the effect of sparse observations using different sampling intervals, $\{50, 100, 200, \text{ and } , 300\}$ms. The proposed model is more robust with respect to sparse observations, whereas the standard ProMPs overfit to the training data, especially in the early phase of the movement. The performance comparison of the two approaches is illustrated in Figure 5.5(B).

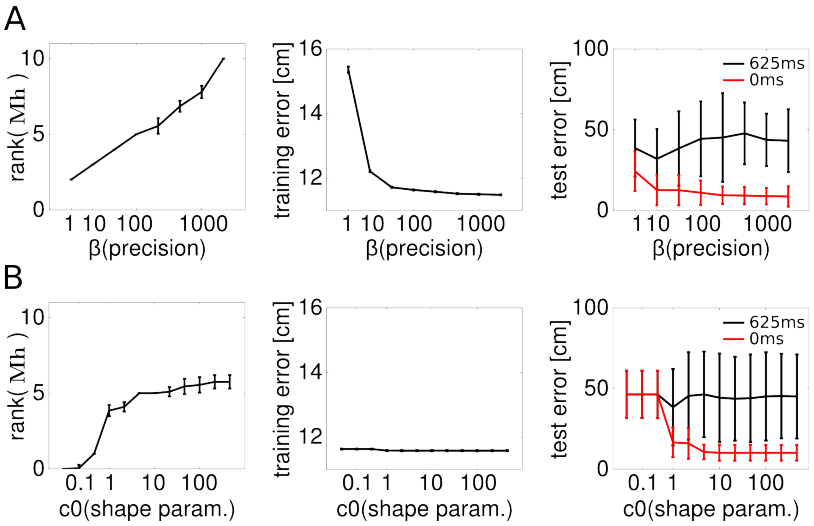

Figure 5.6.: Ping-Pong: (A) The parameter β denotes the data precision. It can be used to adapt the model complexity (first panel in A). With increasing β values the training error decreases and the model overfits to the training data. This is shown for two prediction horizons (625ms and 0ms until the ball impact) in the 3rd panel in (A). The numerical stability of the proposed model can be increased by adding an gamma prior on the precision parameters λ, which has little effect on the prediction performance (for $c_0 \geq 1$).

5.3 Analyzing the model parameters

As opposed to most movement primitive approaches, our model has only one free parameter to choose that is the precision of the data denoted by β. For large β values the number of contributing latent variables in the generative model is increased, and, at some point, the model will overfit to the training data. To analyze this effect, we approximate the complexity of the learned model by computing the rank of the linear feature weights denoted by $\boldsymbol{M}\boldsymbol{h}^{[i]}$ in Equation (4.1)

For values of $\beta \in \{1, 10, 50, 100, 200, 500, 1000, 5000\}$ we compute the training and test error. The prediction performance is shown in Figure 5.6(A). The lowest test error was achieved for $\beta = 10$. Note that the test error will not converge to zero due to noise introduced with $\sigma_h = 0.02$ and $\sigma_b = 0.05$, and the sparse observations at 50ms intervals.

The numerical stability of the LMProMPs can be increased with the addition of a gamma prior on the $\lambda^{[\nu]}$ parameters, discussed in Section 4.1. To investigate the influence of this regularization on the test error, we evaluated gamma priors with a constant mean ($c_0/d_0 = 100$) and increasing precision in the interval $c_0 \in [0.05, 500]$. For small values of c_0 the prior converges to a uniform distribution. For $c_0 \geq 1$ the variational updates were numerically stable and the gamma prior had only little influence on the test error, as shown in Figure 5.6(B).

Finally, we semantically analyze the table tennis dataset to evaluate how the latent variable affect the learned prior distribution. We trained the model with 10-dimensional latent variables $\boldsymbol{h}^{[i]}$ in Equation (4.8). The effect of the first two latent dimensions in the generative model is illustrated in Figure 5.7(A-B). The two latent dimensions of the model affect the final position and the waviness of the x-coordinate of the racket trajectories shown in Figure 5.7(B).

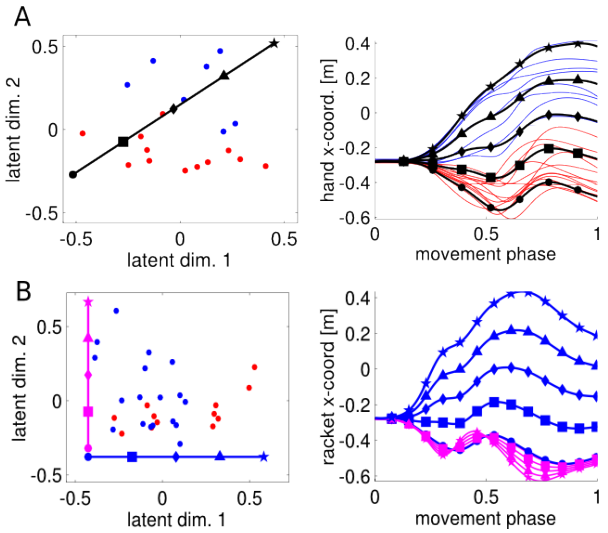

Figure 5.7.: Ping-Pong: (A) Semantic analysis of the data varying the first 2 dimensions of the latent control variable $\boldsymbol{h}$. (B) The first dimension of $\boldsymbol{h}$ describes the final position while the second dimension relates to the waviness of the trajectories.

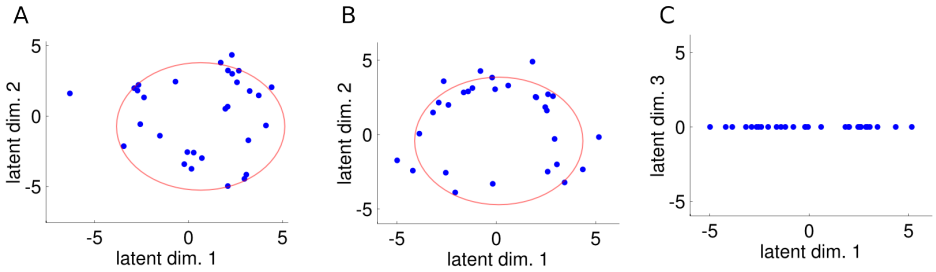

Figure 5.8.: Golf: The first two dimensions of the control variable $\boldsymbol{h}$ for the novices are denoted as the blue points in (A) while they are shown for the skilled player in (B). The red ellipse illustrates the structure contained in the data. (C) The other dimensions of the control variable do not have an impact on the learned manifold.

As shown in Figure 5.7(B) we can assign meaningful features to many of the dimensions of the extracted control variables $\boldsymbol{h}$. Here we have shown that the final position can be modulated by varying the first dimension of the control parameter, while changing the second dimension modulates the waviness of the trajectory. If we now want to improve they way we play table tennis we could analyze the strokes of experts and analyze the differences in their movements in the lower dimensional control parameter space, which is much easier than in the high-dimensional weight vector space or using raw trajectories.

We also applied our model to analyze the similarities of movements between novice and experts in golf. Therefore we use data from 30 swings from novices and 30 swings from experts, which are recorded using visual markers. We use the velocity profile for the *Club P* marker, which is placed at the top of the shaft. We trained the model with 10-dimensional latent control variables $\boldsymbol{h}^{[i]}$. In Figure 5.8(A-B) we show the first two dimensions of the learned control variables. While the control variables for experts form approximately an ellipse which is illustrated by the red line, the ones for novices does vanishes more from the intended structure. This is the only structure contained in the data since all other dimensions

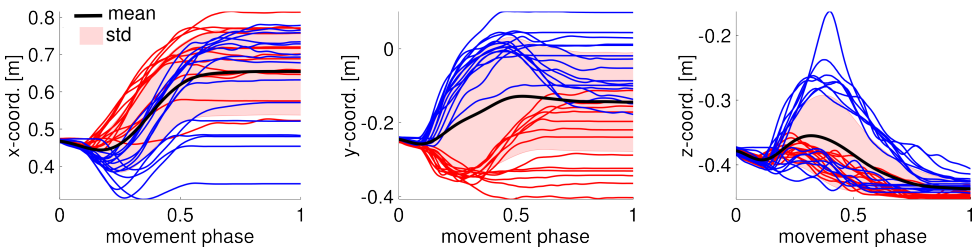

Figure 5.9.: Bi-Modal: Learned distributions using ProMPs. The mean is denoted by the black line and the standard deviation by the shaded region. ProMPs cannot represent the bi-modal distribution in the 2nd panel.

of the control variable decreased to zero. This shows that how our model is not bound to the pre-specified dimension of control variables but extracts the dimension to some extend from the data.

This experiment also demonstrates that our model can be applied to a multitude of different tasks like table tennis and golf without a lot of tuning to the specific task.

5.4 Learning bi-modal trajectory distributions

To demonstrate that LMProMPs can model multi-modal distributions, we study demonstrations of a bi-modal target-reaching task. A KUKA lightweight arm was used to reach for different target locations on a table while avoiding an obstacle. We used kinesthetic teaching and we demonstrated two different ways to approach the target.

For a comparison, we trained ProMPs to learn from the demonstrations, which were unable to represent the two modes. As a result, generalization by conditioning to not encountered target locations may result in trajectories that pass through the obstacle shown in Figure 5.10(B). The learned distributions and example trajectories are shown in Figure 5.9.

In contrast, the LMProMPs model is able to capture the two modes of the demonstrations, as shown in Figure 5.10. We initialized the experiment with *K-means* clustering method using two components. The learned prior distribution and the influence of the first two dimensions of the latent variable are illustrated in Figure 5.10(A-B).

Each mixture component specializes on one mode of the data. Using the learned bi-modal prior distribution, our model is able to generate trajectories to new target locations that avoid the obstacle as shown in Figure 5.10(C). The inferred trajectories are smooth and can be executed on the real robot using inverse kinematics to obtain a reference joint trajectory and inverse dynamics control to execute it. The resulting trajectories of the end-effector of the real robot are illustrated in Figure 5.10(D).

The LMProMP outperforms the original ProMPs as they avoid the obstacle due to the more appropriate prior distribution.

5.5 Summary of the investigated features

We compare the proposed LMProMP model to the standard ProMP approach in the two robotic setups and one synthetic dataset.

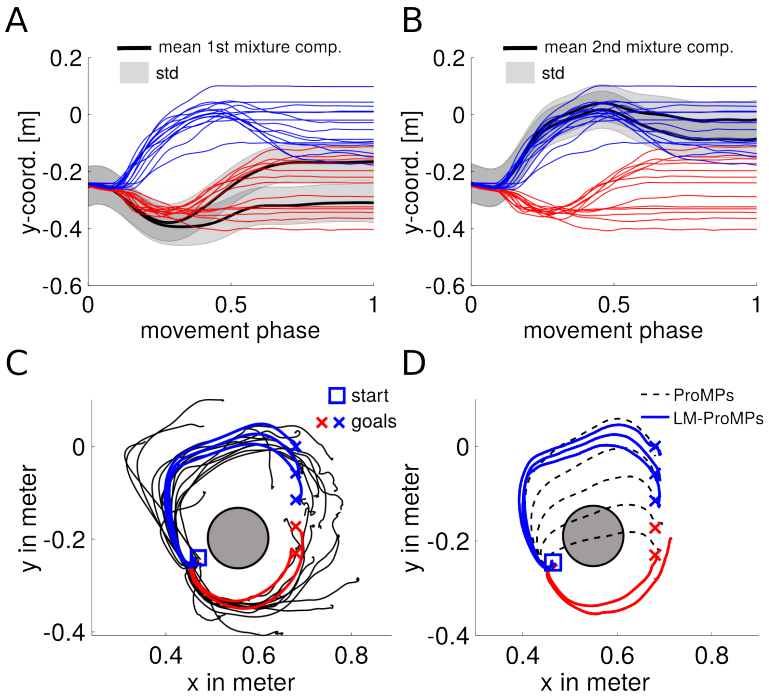

Figure 5.10.: Bi-Modal: Learned bi-modal distribution using the proposed mixture model with two mixture components (A-B). The latent variable is used to specialize on sub-regions within the distribution of the mixture component. This is illustrated for two dimensions of **h**, where solid black lines denote the mean. (C) Results for conditioning on unseen targets for each mode using LMProMPs. (D) Real robot results where LMProMPs avoid the obstacle, while the conditioned trajectories with ProMPs fail.

For the synthetic dataset we investigated how the different approaches model the data. We inspect the prior distributions of ProMPs and LMProMPs and their importance of generating trajectories which reproduces the training data.

In the table tennis scenario we investigate the effect of noise and missing data on predicting the final ball impact location at the opponent's side of the table and we demonstrate how the learned latent variables can be used to semantically analyze the data.

Additionally, we show in the golf setting using visual markers how the model can be used to semantically analyze data in a different setup without the need of tuning a lot of parameters. We analyze the structure in the space of the latent control variables to show the difference between experts and novices.

Finally, we demonstrate the beneficial properties of the mixture model in representing the bi-modal distribution required to successfully execute the KUKA reaching task. We use the learned mixture model to generate trajectories to new target locations, not encountered during training, and execute them on the real robot. We demonstrate that our proposed approach successfully avoids the obstacle, while the standard ProMPs average over the two modes and the generalization fails.

In both experiments we used linear regression to compute the feature weights $\boldsymbol{w}$ and we subsequently applied a principal component analysis. We initialized our model with the first ten principal components.

6 Outlook

In this section we first summarize how we introduced control variables into Probabilistic Movement Primitives and present possible future work afterwards.

6.1 Conclusion

In motor control approaches having a low number of control parameters is a desired approach. These control parameters can be used to generalize from learned movements to new or changing situations. Predefining such parameters is a naive approach that can not adapt to the complexity of the task and is lacking flexibility. To face this problem we proposed a probabilistic movement primitive representation with a hierarchical prior. The control parameters are encoded in the prior distribution and are learned from demonstration using variational inference. As in the original formulation of probabilistic movement primitives our model is able to predict unseen tasks by conditioning on the control parameters. The advantage of our model is that the control parameters are lower dimensional than the weight vectors, which is less prone to overfitting.

The model naturally extends to mixture models that can be used to represent multiple movement types. With our proposed model probabilistic movement primitives got more powerful as they now additionally implement control parameters. We demonstrated on synthetic and kinesthetic teaching datasets that these control variables can be used to generate new trajectories or to analyze the data. Additionally, we showed that our model can easily be used in different setups without tuning a lot of parameters. and that our proposed method learns more accurate models which helps analyzing the data.

6.2 Future work

In future work one could think of extending the model in multiple ways. For instance a Dirchlet prior for the mixing coefficients can be used as it is often done in the variational inference for Gaussian mixture models. Also one could use different or additional hyperpriors. Using a Laplace prior on the latent control variable will obtain sparser models, which might be helpful in analyzing the data. One can start with a high number of latent control variables which is automatically reduced to the effective number of hidden states contained in the data. Since this is closely related to non-parametric methods another enhancement might be using established non-parametric methods as Dirchlet process to obtain the number of mixture components for multiple movement types. One also can think of using more complex methods such as Gaussian Processes to capture nonlinearities. One either could use Gaussian Process latent variables models or adapting the method of cost-regularized kernel regression [14]. Finally, one should evaluate the proposed method on more challenging real-robot tasks which contain a larger number of modes to inspect how the proposed method scales in more complex scenarios.

Bibliography

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. In *Machine Learning*. press, 2007.

[2] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[3] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, USA, 2006.

[4] R. Caruana. Multitask learning, 1997.

[5] H. Daume. Bayesian multitask learning with latent hierarchies. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, 2009.

[6] Andrea d'Avella, Philippe Saltiel, and Emilio Bizzi. Combinations of Muscle Synergies in the Construction of a Natural Motor Behavior. *Nature*, 6(3):300–308, March 2003.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[8] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[9] A. Ijspeert and S. Schaal. Learning Attractor Landscapes for Learning Motor Primitives. In *Advances in Neural Information Processing Systems 15*, (NIPS). MIT Press, Cambridge, MA, 2003.

[10] C. Kemp, A. Perfors, and J. Tenenbaum. Learning overhypotheses with hierarchical bayesian models.

[11] M. Khansari-Zadeh and A. Billard. Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models. *IEEE Transaction on Robotics*, 2011.

[12] J. Kober, E. Oztop, and J. Peters. Reinforcement Learning to adjust Robot Movements to New Situations. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*, 2010.

[13] J. Kober and J. Peters. Policy Search for Motor Primitives in Robotics. *Machine Learning*, pages 1–33, 2010.

[14] J. Kober, A. Wilhelm, E. Oztop, and J. Peters. Reinforcement learning to adjust parametrized motor primitives to new situations. *Autonomous Robots*, 33(4):361–379, 2012.

[15] A. Kumar and Daume H. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th international conference on Machine Learning*, 2012.

[16] A. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-Efficient Contextual Policy Search for Robot Movement Skills. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013.

[17] C.H. Lampert and J. Peters. Real-time detection of colored objects in multiple camera streams with off-the-shelf hardware components. *Journal of Real-Time Image Processing*, 2012.

[18] A. Lazaric and M. Ghavamzadeh. Bayesian multi-task reinforcement learning. In *ICML '10 Proceedings of the 27th international conference on Machine Learning*, 2010.

[19] K. Mülling, J. Kober, and J. Peters. A Biomimetic Approach to Robot Table Tennis. *Adaptive Behavior Journal*, (5), 2011.

[20] A. Paraschos, C. Daniel, J. Peters, and G Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems (NIPS), Cambridge, MA: MIT Press.*, 2013.

[21] A. Paraschos, G Neumann, and J. Peters. A probabilistic approach to robot trajectory generation. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, 2013.

[22] A. Passos, P. Rai, J. Wainer, and H. Daume. Flexible modeling of latent task structures in multitask learning. In *In Proceedings of International Conference on Machine Learning*, 2012.

[23] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and Generalization of Motor Skills by Learning from Demonstration. In *International Conference on Robotics and Automation (ICRA)*, 2009.

[24] P. Rai and H. Daume. Infinite predictor subspace models for multitask learning. In *Int. Conf. on Artificial Intelligence and Statistics*, 2010.

[25] B. Romera-Paredes, H. Aung, N. Bianchi-Berthouze, and M. Pontil. Multilinear multitask learning. In *Proc. Int. Conf. on Machine Learning*, 2013.

[26] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras. Learning Collaborative Impedance-Based Robot Behaviors. In *AAAI Conference on Artificial Intelligence*, 2013.

[27] P. Ruvolo and E. Eaton. Online multi-task learning via sparse dictionary optimization. In *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, July 2014.

[28] M. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey.

[29] A. Ude, A. Gams, T. Asfour, and J. Morimoto. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. *Trans. Rob.*, (5), October 2010.

[30] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:2007, 2007.

[31] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *ICML '05 Proceedings of the 22nd international conference on Machine learning*, pages 1012 – 1019, 2005.

A List of publications

[1] E. Rueckert, J. Mundo, A. Paraschos, J. Peters, G. Neumann. (2014). Extracting Low-Dimensional Control Variables for Movement Primitives. In Review process for *IEEE International Conference on Robotics and Automation (ICRA 2015)*, Washington, USA, 2015.

A.1 Comments and Contributions to Publications

The paper *Extracting Low-Dimensional Control Variables for Movement Primitives* was written by myself (JM), Elmar Rueckert (ER), Jan Peters (JP) and Gerhard Neumann (GN). While GN and ER developed the basic ideas of this paper and created a simpler hierarchical Bayesian model, I extended their method. In particular, an additional prior distribution was used to improve numerical stability. The implementation of the model was done by JM while ER implemented most of the experiments. Section 3 and Section 5 build partially on results presented in the paper.

B Update equations LMProMPs

We derive the update equations for the Latent Manifold Probabilistic Movement Primitives (LMProMPS). We start with the proposed model for single movement types and conclude with the model of multiple movement types.

B.1 Single movement type

First we define 4 helper functions, which are used heavily throughout this derivation:

$$1. \sum_{v=1}^{V} \boldsymbol{m}^{[v]} \boldsymbol{h}^{[v,i]} = \boldsymbol{M} \boldsymbol{h}^{[i]},$$

$$\begin{aligned} 2. < \boldsymbol{h}^{[v,i]^T} \boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} \boldsymbol{h}^{[v,i]} >_{\boldsymbol{h}^{[v,i]}} &= \boldsymbol{h}^{[v,i]} \boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} \boldsymbol{h}^{[v,i]} + tr[\boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} \sigma_{h^{[v,i]}}] \\ &= \boldsymbol{m}^{[v]^T} (\boldsymbol{h}^{[v,i]})^2 \boldsymbol{m}^{[v]^T} + \boldsymbol{m}^{[v]^T} \sigma_{h^{[v,i]}} \boldsymbol{m}^{[v]} \\ &= \boldsymbol{m}^{[v]^T} (\boldsymbol{h}^{[v,i]^T} \boldsymbol{h}^{[v,i]} + \sigma_{h^{[v,i]}}) \boldsymbol{m}^{[v]^T}. \end{aligned}$$

This is because $\boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} \sigma_{h^{[v,i]}} \in \mathbb{R}^{1x1}$.

$$3. < \boldsymbol{M}^T \boldsymbol{M} >_{\boldsymbol{M}} = \bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \begin{bmatrix} \sigma_{\boldsymbol{m}^{[1]}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\boldsymbol{m}^{[V]}} \end{bmatrix} = \bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}},$$

$$\begin{aligned} 4. < \boldsymbol{h}^{[i]^T} \boldsymbol{M}^T \boldsymbol{M} \boldsymbol{h}^{[i]} >_{\boldsymbol{h}^{[i]}, \boldsymbol{M}} &= < \boldsymbol{h}^{[i]^T} < \boldsymbol{M}^T \boldsymbol{M} >_{\boldsymbol{M}} \boldsymbol{h}^{[i]} >_{\boldsymbol{h}^{[i]}} \\ &= < \boldsymbol{h}^{[i]^T} (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) \boldsymbol{h}^{[i]} >_{\boldsymbol{h}^{[i]}} \\ &= \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[(\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}]. \end{aligned}$$

This is valid because we do not have a correlation between $\boldsymbol{h}^{[i]}$ and $\boldsymbol{M}$ since we assume a complete factorization of the q distributions. Note that also the expectation over M factorizes $< \cdot >_{\boldsymbol{M}} = < \cdot >_{\boldsymbol{m}^{[1]}, \ldots, \boldsymbol{m}^{[V]}}$ since all the columns of $\boldsymbol{M}$ are considered as independent latent variables and are only combined into a matrix for easier computation.

The update equations for the approximate variational posterior of the parameter vector $\boldsymbol{w}^{[i]}$ reads as follows:

$$\begin{aligned} \log q^*(\boldsymbol{w}^{[i]}) &= < \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}, \beta) p(\boldsymbol{w}^{[i]} | \boldsymbol{b}, \boldsymbol{M}, \boldsymbol{h}^{[i]}, \alpha) >_{\boldsymbol{h}^{[i]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}} \\ &= < \log \mathcal{N}(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]}, \beta^{-1} \boldsymbol{I}_S) \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b} + \boldsymbol{M} \boldsymbol{h}^{[i]}, (\alpha)^{-1} \boldsymbol{I}_d) >_{\boldsymbol{h}^{[i]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}} \end{aligned}$$

$$
\begin{aligned}
=&\frac{S}{2}\log 2\pi\beta - \frac{\beta}{2}(\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]})^T(\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]}) + \\
&< \frac{d}{2}\log 2\pi\alpha - \frac{\alpha}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]}) >_{\boldsymbol{h}^{[i]},\boldsymbol{b},\boldsymbol{M},\alpha,\lambda^{[1:V]}} \\
=&-\frac{\beta}{2}(-2\boldsymbol{w}^{[i]^T}\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} + \boldsymbol{w}^{[i]^T}\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]}) \\
&-\frac{\bar{\alpha}}{2}(\boldsymbol{w}^{[i]^T}\boldsymbol{w}^{[i]} - 2\boldsymbol{w}^{[i]^T}\boldsymbol{\mu}_{\boldsymbol{b}} - 2\boldsymbol{w}^{[i]^T}\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) + const \\
=&-\frac{1}{2}\boldsymbol{w}^{[i]^T}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]} + \bar{\alpha}\boldsymbol{I}_d)\boldsymbol{w}^{[i]} + \boldsymbol{w}^{[i]^T}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} + \bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})) + const
\end{aligned}
$$

This is a cannonical Gaussian distribution. We get the optimal solution

$$
\begin{aligned}
q^*(\boldsymbol{w}^{[i]}) =& \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}), \text{ with} \\
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} =& \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} + \bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})), \\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} =& (\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]} + \bar{\alpha}\boldsymbol{I}_d)^{-1}.
\end{aligned}
$$

The offset vector $\boldsymbol{b}$ updates are

$$
\begin{aligned}
\log q^*(\boldsymbol{b}) =& < \log\prod_{i=1}^{L} p(\boldsymbol{w}^{[i]}|\boldsymbol{b},\boldsymbol{M},\boldsymbol{h}^{[i]},\alpha)p(\boldsymbol{b}|\lambda^{[0]}) >_{\boldsymbol{w}^{[1:L]},\boldsymbol{h}^{[1:L]},\boldsymbol{M},\alpha,\lambda^{[1:V]}} \\
=& < \log\prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{b}+\boldsymbol{M}\boldsymbol{h}^{[i]},(\alpha)^{-1}\boldsymbol{I}_d)\mathcal{N}(\boldsymbol{b}|\boldsymbol{0},(\lambda^{[0]})^{-1}\boldsymbol{I}_d) >_{\boldsymbol{w}^{[1:L]},\boldsymbol{h}^{[1:L]},\boldsymbol{M},\alpha,\lambda^{[1:V]}} \\
=& < \sum_{i=1}^{L}\frac{d}{2}\log 2\pi\alpha - \frac{\alpha}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]}) >_{\boldsymbol{w}^{[1:L]},\boldsymbol{h}^{[1:L]},\boldsymbol{M},\alpha,\lambda^{[1:V]}} \\
&+\frac{d}{2}\log 2\pi\lambda^{[0]} - \frac{\lambda^{[0]}}{2}\boldsymbol{b}^T\boldsymbol{b} \\
=& \sum_{i=1}^{L}\{-\frac{\bar{\alpha}}{2}(\boldsymbol{b}^T\boldsymbol{b} - 2\boldsymbol{b}^T\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} + 2\boldsymbol{b}^T\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})\} - \frac{\lambda^{[0]}}{2}\boldsymbol{b}^T\boldsymbol{b} + const \\
=& -\frac{1}{2}\boldsymbol{b}^T(\sum_{i=1}^{L}\{\bar{\alpha}\} + \lambda^{[0]})\boldsymbol{b} + \boldsymbol{b}^T\sum_{i=1}^{L}\{\bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})\} + const
\end{aligned}
$$

This is a cannonical Gaussian distribution where

$$
\begin{aligned}
q^*(\boldsymbol{b}) =& \mathcal{N}(\boldsymbol{b}|\boldsymbol{\mu}_{\boldsymbol{b}}, \sigma_{\boldsymbol{b}}\boldsymbol{I}_d), \text{ with} \\
\boldsymbol{\mu}_{\boldsymbol{b}} =& \sigma_{\boldsymbol{b}}(\sum_{i=1}^{L}\{\bar{\alpha}(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})\}), \\
\sigma_{\boldsymbol{b}} =& \sum_{i=1}^{L}\{\bar{\alpha}\} + \lambda^{[0]} = L\bar{\alpha} + \lambda^{[0]}.
\end{aligned}
$$

The projection vector $\boldsymbol{m}^{[v]}$ updates are

$$
\log q^*(\boldsymbol{m}^{[v]}) = < \log\prod_{i=1}^{L} p(\boldsymbol{w}^{[i]}|\boldsymbol{b},\boldsymbol{M},\boldsymbol{h}^{[i]},\alpha)p(\boldsymbol{m}^{[v]}|\lambda^{[v]}) >_{\boldsymbol{w}^{[1:L]},\boldsymbol{h}^{[1:L]},\boldsymbol{b},\alpha,\lambda^{[v]}}
$$

$$
\begin{aligned}
&= < \log \prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, (\alpha)^{-1}\boldsymbol{I}_d) \mathcal{N}(\boldsymbol{m}^{[v]} | \boldsymbol{0}, (\lambda^{[v]})^{-1}\boldsymbol{I}_d) >_{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \alpha, \lambda^{[v]}} \\
&= < \sum_{i=1}^{L} \frac{d}{2} \log 2\pi\alpha - \frac{\alpha}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \sum_{v=1}^{V} \boldsymbol{m}^{[v]} \boldsymbol{h}^{[v,i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \sum_{v=1}^{V} \boldsymbol{m}^{[v]} \boldsymbol{h}^{[v,i]}) \\
&\quad + \frac{d}{2} \log 2\pi\lambda^{[v]} - \frac{\lambda^{[v]}}{2} \boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} >_{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \alpha, \lambda^{[v]}} \\
&= \sum_{i=1}^{L} \{ -\frac{\bar{\alpha}}{2} (2\boldsymbol{m}^{[v]^T} \mu_{h^{[v,i]}}{}^T \boldsymbol{\mu}_{\boldsymbol{b}} - 2\boldsymbol{m}^{[v]^T} \mu_{h^{[v,i]}}{}^T \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} \\
&\quad + \boldsymbol{m}^{[v]^T} \mu_{h^{[v,i]}}{}^T \mu_{h^{[v,i]}} \boldsymbol{m}^{[v]} + \boldsymbol{m}^{[v]^T} \sigma_{h^{[v,i]}} \boldsymbol{m}^{[v]}) \} - \frac{\bar{\lambda}^{[v]}}{2} \boldsymbol{m}^{[v]^T} \boldsymbol{m}^{[v]} + const \\
&= -\frac{1}{2} \boldsymbol{m}^{[v]^T} (\sum_{i=1}^{L} \{ \bar{\alpha} (\mu_{h^{[v,i]}}{}^T \mu_{h^{[v,i]}} + \sigma_{h^{[v,i]}}) \} + \bar{\lambda}^{[v]}) \boldsymbol{m}^{[v]} \\
&\quad + \boldsymbol{m}^{[v]^T} (\sum_{i=1}^{L} \{ \bar{\alpha} (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}}) \mu_{h^{[v,i]}}{}^T \}) + const
\end{aligned}
$$

This is a cannonical Gaussian distribution where

$$
\begin{aligned}
q^*(\boldsymbol{m}^{[v]}) &= \mathcal{N}(\boldsymbol{m}^{[v]} | \boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}, \sigma_{\boldsymbol{m}^{[v]}} \boldsymbol{I}_d), \text{ with} \\
\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} &= \sigma_{\boldsymbol{m}^{[v]}} (\sum_{i=1}^{L} \{ \bar{\alpha} \mu_{h^{[v,i]}} (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}}) \}), \\
\sigma_{\boldsymbol{m}^{[v]}} &= (\sum_{i=1}^{L} \{ \bar{\alpha} ((\mu_{h^{[v,i]}})^2 + \sigma_{h^{[v,i]}}) \} + \bar{\lambda}^{[v]})^{-1} = (\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}(v,v))^{-1},
\end{aligned}
$$

where $\sigma_{h^{[v,i]}}$ is the $v-th$ row and $v-th$ column of $\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}$ The vectors $\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}$ are combined into the matrix $\bar{\boldsymbol{M}} = [\boldsymbol{\mu}_{\boldsymbol{m}^{[1]}}, \ldots, \boldsymbol{\mu}_{\boldsymbol{m}^{[V]}}]$. The precision parameters $\sigma_{\boldsymbol{m}^{[v]}}$ are combined into the matrix

$$
\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}} = \begin{bmatrix} \sigma_{\boldsymbol{m}^{[1]}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\boldsymbol{m}^{[V]}} \end{bmatrix}.
$$

The updates of the control variable $\boldsymbol{h}^{[i]}$ read

$$
\begin{aligned}
\log q^*(\boldsymbol{h}^{[i]}) &= < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}, \boldsymbol{M}, \boldsymbol{h}^{[i]}, \alpha) p(\boldsymbol{h}^{[i]} | \gamma) >_{\boldsymbol{w}^{[i]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}} \\
&= < \log \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, (\alpha)^{-1}\boldsymbol{I}_d) \mathcal{N}(\boldsymbol{h}^{[i]} | \boldsymbol{0}, \gamma^{-1}\boldsymbol{I}_V) >_{\boldsymbol{w}^{[i]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}} \\
&= < \frac{d}{2} \log 2\pi\alpha - \frac{\alpha}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]}) >_{\boldsymbol{w}^{[i]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}} \\
&\quad + \frac{V}{2} \log 2\pi\gamma - \frac{\gamma}{2} \boldsymbol{h}^{[i]^T} \boldsymbol{I}_V \boldsymbol{h}^{[i]} \\
&= -\frac{\bar{\alpha}}{2} (2\boldsymbol{h}^{[i]^T} \bar{\boldsymbol{M}}^T \boldsymbol{\mu}_{\boldsymbol{b}} - 2\boldsymbol{h}^{[i]^T} \bar{\boldsymbol{M}}^T \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} + \boldsymbol{h}^{[i]^T} (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) \boldsymbol{h}^{[i]} \\
&\quad - \frac{\gamma}{2} \boldsymbol{h}^{[i]^T} \boldsymbol{I}_V \boldsymbol{h}^{[i]} + const \\
&= -\frac{1}{2} \boldsymbol{h}^{[i]^T} (\bar{\alpha} \boldsymbol{I}_V (\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) + \gamma \boldsymbol{I}_V)) \boldsymbol{h}^{[i]} + \boldsymbol{h}^{[i]^T} (\bar{\alpha} \boldsymbol{I}_V \bar{\boldsymbol{M}}^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}})) + const
\end{aligned}
$$

This is a cannonical Gaussian distribution where

$$\begin{aligned} q^*(\boldsymbol{h}^{[i]}) =& \mathcal{N}(\boldsymbol{h}^{[i]}|\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}), \text{ with} \\ \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} =& \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}(\bar{\alpha}\boldsymbol{I}_V \bar{\boldsymbol{M}}^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}})), \\ \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}} =& (\bar{\alpha}\boldsymbol{I}_V(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}} + d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}) + \gamma \boldsymbol{I}_V)^{-1}. \end{aligned}$$

In difference to the updates for the projection vectors $\boldsymbol{m}^{[v]}$ we learn a covariance between the different dimensions of $\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}$.

The updates for the precision parameter $\lambda^{[v]}$ of the projection vector read

$$\begin{aligned} \log q^*(\lambda^{[v]}) =& < \log p(\boldsymbol{m}^{[v]}|\lambda^{[v]}) p(\lambda^{[v]}|c^{[0]}, d^{[0]}) >_{\boldsymbol{m}^{[v]}} \\ =& < \log \mathcal{N}(\boldsymbol{m}^{[v]}|\boldsymbol{0}, (\lambda^{[v]})^{-1}\boldsymbol{I}_d)\Gamma(\lambda^{[v]}|c^{[0]}, d^{[0]}) >_{\boldsymbol{m}^{[v]}} \\ =& \frac{d}{2}\log 2\pi + \frac{d}{2}\log\lambda^{[v]} - < \frac{\lambda^{[v]}}{2} w_v^T w_v >_{\boldsymbol{m}^{[v]}} \\ & - \Gamma(c^{[0]}) + c^{[0]}\log d^{[0]} + (c^{[0]} - 1)\log\lambda^{[v]} - d^{[0]}\lambda^{[v]} \\ =& (c^{[0]} + \frac{d}{2} - 1)\log\lambda^{[v]} - (d^{[0]} + \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}{}^T\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} + d\sigma_{\boldsymbol{m}^{[v]}})\lambda^{[v]} + const \end{aligned}$$

This is a Gamma distribution, where we get the optimal solution

$$\begin{aligned} q^*(\lambda^{[v]}) =& \Gamma(\lambda^{[v]}|\bar{c}, \bar{d}), \text{ with} \\ \bar{c} =& c^{[0]} + \frac{d}{2}, \\ \bar{d} =& d^{[0]} + \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}{}^T\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} + d\sigma_{\boldsymbol{m}^{[v]}}). \end{aligned}$$

The expectation of the precision parameter $\bar{\lambda}^{[v]}$ is given by

$$\bar{\lambda}^{[v]} = < \lambda^{[v]} >_{\lambda^{[v]}} = \frac{\bar{c}}{\bar{d}}.$$

The updates for precision parameter of the weight vector α read

$$\begin{aligned} \log q^*(\alpha) =& < \log \prod_{i=1}^{L} p(\boldsymbol{w}^{[i]}|\boldsymbol{b}, \boldsymbol{M}, \boldsymbol{h}^{[i]}, \alpha) p(\alpha|a^{[0]}, b^{[0]}) >_{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \lambda^{[1:V]}} \\ =& < \log \prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, \alpha\boldsymbol{I}_d)\Gamma(\alpha|a^{[0]}, b^{[0]}) >_{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \lambda^{[1:V]}} \\ =& < \sum_{i=1}^{L} \frac{d}{2}\log 2\pi\alpha - \frac{\alpha}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]}) >_{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \lambda^{[1:V]}} \\ & - \Gamma(a^{[0]}) + a^{[0]}\log b^{[0]} + (a^{[0]} - 1)\log\alpha - b^{[0]}\alpha \\ =& \sum_{i=1}^{L} \frac{d}{2}\log\alpha - \frac{\alpha}{2}[(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) \\ & + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}} + d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}]] \end{aligned}$$

$$
\begin{aligned}
&+(a^{[0]}-1)\log\alpha - b^{[0]}\alpha + const \\
=&((a^{[0]}+\frac{dL}{2}-1)\log\alpha - (b^{[0]}+\frac{1}{2}\sum_{i=1}^{L}\{(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}-\boldsymbol{\mu}_{\boldsymbol{b}}-\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T \\
&(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}-\boldsymbol{\mu}_{\boldsymbol{b}}-\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} \\
&+tr[(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}}+d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}])\alpha + const
\end{aligned}
$$

This is a Gamma distribution. We get the optimal solution

$$
\begin{aligned}
q^*(\alpha) =& \Gamma(\alpha|\bar{a},\bar{b}), \text{ with} \\
\bar{a} =& a^{[0]}+\frac{dL}{2}, \\
\bar{b} =& b^{[0]}+\frac{1}{2}\sum_{i=1}^{L}\{(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}-\boldsymbol{\mu}_{\boldsymbol{b}}-\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}-\boldsymbol{\mu}_{\boldsymbol{b}}-\bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) \\
&+tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + d\sigma_{\boldsymbol{b}} + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}}+d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}].
\end{aligned}
$$

The expectation of the precision parameter $\bar{\alpha}$ is given by

$$
\bar{\alpha} = <\alpha>_{\alpha} = \frac{\bar{a}}{\bar{b}}.
$$

We use the helper functions from Section B.1. The update equations for the approximate variational posterior of the parameter vector $\boldsymbol{w}^{[i]}$ reads

$$\begin{aligned}
\log q^*(\boldsymbol{w}^{[i]}) &= < \log p(\boldsymbol{y}^{[i]}|\boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}, \beta) \prod_{k=1}^{K} p(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) >_{\boldsymbol{Z}\setminus\boldsymbol{w}^{[1:L]}} \\
&= < \log \mathcal{N}(\boldsymbol{y}^{[i]}|\boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]}, \beta^{-1}\boldsymbol{I}_S) \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k + \boldsymbol{M}_k\boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1}\boldsymbol{I}_d)^{z_k^{[i]}} >_{\boldsymbol{Z}\setminus\boldsymbol{w}^{[1:L]}} \\
&= \frac{S}{2}\log 2\pi\beta - \frac{\beta}{2}(\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]})^T(\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]}) + \sum_{k=1}^{K} < z_k^{[i]} [\frac{d}{2}\log 2\pi\alpha_k \\
&\quad - \frac{\alpha_k}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})] >_{\boldsymbol{Z}\setminus\boldsymbol{w}^{[1:L]}} \\
&= -\frac{\beta}{2}(-2\boldsymbol{w}^{[i]^T}\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} + \boldsymbol{w}^{[i]^T}\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]}\boldsymbol{w}^{[i]}) \\
&\quad - \sum_{k=1}^{K} \frac{\mu_{z_k^{[i]}}\bar{\alpha}_k}{2}(\boldsymbol{w}^{[i]^T}\boldsymbol{w}^{[i]} - 2\boldsymbol{w}^{[i]^T}\boldsymbol{\mu}_{\boldsymbol{b}_k} - 2\boldsymbol{w}^{[i]^T}\bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) + const \\
&= -\frac{1}{2}\boldsymbol{w}^{[i]^T}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]} + \sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k\boldsymbol{I}_d)\boldsymbol{w}^{[i]} + \boldsymbol{w}^{[i]^T}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} \\
&\quad + \sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k\boldsymbol{I}_d(\boldsymbol{\mu}_{\boldsymbol{b}_k} + \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})) + const.
\end{aligned}$$

This is a cannonical Gaussian distribution. We get the optimal solution

$$\begin{aligned}
q^*(\boldsymbol{w}^{[i]}) &= \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}) \text{ with} \\
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} &= \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}(\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{y}^{[i]} + \sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k(\boldsymbol{\mu}_{\boldsymbol{b}_k} + \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})), \\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} &= (\beta\boldsymbol{\Psi}^{[i]^T}\boldsymbol{\Psi}^{[i]} + \sum_{k=1}^{K}\mu_{z_k^{[i]}}\bar{\alpha}_k\boldsymbol{I}_d)^{-1}.
\end{aligned}$$

Here we used $< \cdot >_{\boldsymbol{Z}\setminus\boldsymbol{w}^{[1:L]}}$ to denote the expectations with respect to all variational distributions expect for $\boldsymbol{w}^{[1:L]}$. The updates of the offset vector $\boldsymbol{b}_k$ for each mixture component reads

$$\begin{aligned}
\log q^*(\boldsymbol{b}_k) &= < \log \prod_{i=1}^{L} p(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) p(\boldsymbol{b}_k|\lambda_k^{[0]}) >_{\boldsymbol{Z}\setminus\boldsymbol{b}_k} \\
&= < \log \prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k + \boldsymbol{M}_k\boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1}\boldsymbol{I}_d)^{z_k^{[i]}} \mathcal{N}(\boldsymbol{b}_k|\boldsymbol{0}, (\lambda_k^{[0]})^{-1}\boldsymbol{I}_d) >_{\boldsymbol{Z}\setminus\boldsymbol{b}_k} \\
&= < \sum_{i=1}^{L} z_k^{[i]} [\frac{d}{2}\log 2\pi\alpha_k - \frac{\alpha_k}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})] >_{\boldsymbol{Z}\setminus\boldsymbol{b}_k} \\
&\quad + \frac{d}{2}\log 2\pi\lambda_k^{[0]} - \frac{\lambda_k^{[0]}}{2}\boldsymbol{b}_k{}^T\boldsymbol{b}_k
\end{aligned}$$

$$= \sum_{i=1}^{L} \{ -\frac{\mu_{z_k^{[i]}} \bar{\alpha}_k}{2} (\boldsymbol{b}_k{}^T \boldsymbol{b}_k - 2\boldsymbol{b}_k{}^T \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} + 2\boldsymbol{b}_k{}^T \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) \} - \frac{\lambda_k^{[0]}}{2} \boldsymbol{b}_k{}^T \boldsymbol{b}_k + const$$

$$= -\frac{1}{2} \boldsymbol{b}_k{}^T (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k \} + \lambda_k^{[0]}) \boldsymbol{b}_k + \boldsymbol{b}_k{}^T \sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) \} + const$$

This is a cannonical Gaussian distribution

$$q^*(\boldsymbol{b}_k) = \mathcal{N}(\boldsymbol{b}_k | \boldsymbol{\mu}_{\boldsymbol{b}_k}, \sigma_{\boldsymbol{b}_k} \boldsymbol{I}_d), \text{ with}$$

$$\boldsymbol{\mu}_{\boldsymbol{b}_k} = \sigma_{\boldsymbol{b}_k} (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) \}),$$

$$\sigma_{\boldsymbol{b}_k} = (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k \} + \lambda_k^{[0]})^{-1}.$$

The projection vectors $\boldsymbol{m}_k^{[v]}$ are updated as

$$\log q^*(\boldsymbol{m}_k^{[v]}) = < \log \prod_{i=1}^{L} p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) p(\boldsymbol{m}_k^{[v]} | \lambda_k^{[v]}) >_{\boldsymbol{Z} \setminus \boldsymbol{m}_k^{[v]}}$$

$$= < \log \prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1} \boldsymbol{I}_d)^{z_k^{[i]}} \mathcal{N}(\boldsymbol{m}_k^{[v]} | \boldsymbol{0}, (\lambda_k^{[v]})^{-1} \boldsymbol{I}_d) >_{\boldsymbol{Z} \setminus \boldsymbol{m}_k^{[v]}}$$

$$= < \sum_{i=1}^{L} z_k^{[i]} [\frac{d}{2} \log 2\pi\alpha_k - \frac{\alpha_k}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \sum_{v=1}^{V} \boldsymbol{m}_k^{[v]} \boldsymbol{h}_k^{[v,i]})^T$$

$$(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \sum_{v=1}^{V} \boldsymbol{m}_k^{[v]} \boldsymbol{h}_k^{[v,i]})] + \frac{d}{2} \log 2\pi\lambda_k^{[v]} - \frac{\lambda_k^{[v]}}{2} \boldsymbol{m}_k^{[v]T} \boldsymbol{m}_k^{[v]} >_{\boldsymbol{Z} \setminus \boldsymbol{m}_k^{[v]}}$$

$$= \sum_{i=1}^{L} \{ -\frac{\mu_{z_k^{[i]}} \bar{\alpha}_k}{2} (2\boldsymbol{m}_k^{[v]T} \mu_{\boldsymbol{h}_k^{[v,i]}}{}^T \boldsymbol{\mu}_{\boldsymbol{b}_k} - 2\boldsymbol{m}_k^{[v]T} \mu_{\boldsymbol{h}_k^{[v,i]}}{}^T \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}$$

$$+ \boldsymbol{m}_k^{[v]T} \mu_{\boldsymbol{h}_k^{[v,i]}}{}^T \mu_{\boldsymbol{h}_k^{[v,i]}} \boldsymbol{m}_k^{[v]} + \boldsymbol{m}_k^{[v]T} \sigma_{\boldsymbol{h}_k^{[v,i]}} \boldsymbol{m}_k^{[v]}) \} - \frac{\bar{\lambda}_k^{[v]}}{2} \boldsymbol{m}_k^{[v]T} \boldsymbol{m}_k^{[v]} + const$$

$$= -\frac{1}{2} \boldsymbol{m}_k^{[v]T} (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k ((\mu_{\boldsymbol{h}_k^{[v,i]}})^2 + \sigma_{\boldsymbol{h}_k^{[v,i]}}) \} + \bar{\lambda}_k^{[v]}) \boldsymbol{m}_k^{[v]}$$

$$+ \boldsymbol{m}_k^{[v]T} (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k}) \mu_{\boldsymbol{h}_k^{[v,i]}} \}) + const$$

This is a cannonical Gaussian distribution

$$q^*(\boldsymbol{m}_k^{[v]}) = \mathcal{N}(\boldsymbol{m}_k^{[v]} | \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}, \sigma_{\boldsymbol{m}_k^{[v]}} \boldsymbol{I}_d), \text{ with}$$

$$\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} = \sigma_{\boldsymbol{m}_k^{[v]}} (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k}) \mu_{\boldsymbol{h}_k^{[v,i]}} \})$$

$$\sigma_{\boldsymbol{m}_k^{[v]}} = (\sum_{i=1}^{L} \{ \mu_{z_k^{[i]}} \bar{\alpha}_k ((\mu_{\boldsymbol{h}_k^{[v,i]}})^2 + \sigma_{\boldsymbol{h}_k^{[v,i]}}) \} + \bar{\lambda}_k^{[v]})^{-1} = (\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}(v,v))^{-1}$$

where $\sigma_{h_k^{[\nu,i]}}$ is the $\nu - th$ row and $\nu - th$ column of $\mathbf{\Sigma}_{\boldsymbol{h}_k^{[i]}}$ The vectors $\boldsymbol{\mu}_{\boldsymbol{m}_k^{[\nu]}}$ are combined into the matrix $\bar{\boldsymbol{M}}_k = [\boldsymbol{\mu}_{\boldsymbol{m}_k^{[1]}}, \ldots, \boldsymbol{\mu}_{\boldsymbol{m}_k^{[V]}}]$. The precision parameters $\lambda_k^{[\nu]}$ are combined into the matrix

$$\mathbf{\Sigma}_{\boldsymbol{M}_k^{[V]}} = \begin{bmatrix} \sigma_{\boldsymbol{m}_k^{[1]}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\boldsymbol{m}_k^{[V]}} \end{bmatrix}$$

for each component k.

The updates for the control variables $\boldsymbol{h}_k^{[i]}$ are

$$\begin{aligned}
\log q^*(\boldsymbol{h}_k^{[i]}) &= < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) p(\boldsymbol{h}_k^{[i]} | \gamma_k) >_{\boldsymbol{Z} \setminus \boldsymbol{h}_k^{[i]}} \\
&= < \log \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1} \boldsymbol{I}_d)^{z_k^{[i]}} \mathcal{N}(\boldsymbol{h}_k^{[i]} | \boldsymbol{0}, (\gamma_k)^{-1} \boldsymbol{I}_V) >_{\boldsymbol{Z} \setminus \boldsymbol{h}_k^{[i]}} \\
&= < \sum_{k=1}^{K} z_k^{[i]} [\frac{d}{2} \log 2\pi\alpha_k - \frac{\alpha_k}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]})] >_{\boldsymbol{Z} \setminus \boldsymbol{h}_k^{[i]}} \\
&\quad + \frac{V}{2} \log 2\pi\gamma_k - \frac{\gamma_k}{2} \boldsymbol{h}_k^{[i]T} \boldsymbol{I}_V \boldsymbol{h}_k^{[i]} \\
&= \sum_{k=1}^{K} - \frac{\mu_{z_k^{[i]}} \bar{\alpha}_k}{2} (2 \boldsymbol{h}_k^{[i]T} \bar{\boldsymbol{M}}_k^T \boldsymbol{\mu}_{\boldsymbol{b}_k} - 2 \boldsymbol{h}_k^{[i]T} \bar{\boldsymbol{M}}_k^T \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} + \boldsymbol{h}_k^{[i]T} (\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d \mathbf{\Sigma}_{\boldsymbol{M}_k^{[V]}}) \boldsymbol{h}_k^{[i]} \\
&\quad - \frac{\gamma_k}{2} \boldsymbol{h}_k^{[i]T} \boldsymbol{I}_V \boldsymbol{h}_k^{[i]} + const \\
&= - \frac{1}{2} \boldsymbol{h}_k^{[i]T} (\sum_{k=1}^{K} \mu_{z_k^{[i]}} \bar{\alpha}_k (\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d \mathbf{\Sigma}_{\boldsymbol{M}_k^{[V]}}) + \gamma_k \boldsymbol{I}_V)) \boldsymbol{h}_k^{[i]} + \\
&\quad \boldsymbol{h}_k^{[i]T} (\sum_{k=1}^{K} \mu_{z_k^{[i]}} \bar{\alpha}_k \bar{\boldsymbol{M}}_k^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k})) + const
\end{aligned}$$

This is a cannonical Gaussian distribution

$$\begin{aligned}
q^*(\boldsymbol{h}_k^{[i]}) &= \mathcal{N}(\boldsymbol{h}_k^{[i]} | \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}, \mathbf{\Sigma}_{\boldsymbol{h}_k^{[i]}}), \text{ with} \\
\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} &= \mathbf{\Sigma}_{\boldsymbol{h}_k^{[i]}} (\sum_{k=1}^{K} \mu_{z_k^{[i]}} \bar{\alpha}_k \bar{\boldsymbol{M}}_k^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k})) \\
\mathbf{\Sigma}_{\boldsymbol{h}_k^{[i]}} &= (\sum_{k=1}^{K} \mu_{z_k^{[i]}} \bar{\alpha}_k (\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d \mathbf{\Sigma}_{\boldsymbol{M}_k^{[V]}}) + \gamma_k \boldsymbol{I}_V)^{-1}
\end{aligned}$$

In difference to the updates for the projection vectors $\boldsymbol{m}_k^{[\nu]}$ we learn a covariance between the different dimensions of $\boldsymbol{h}_k^{[i]}$.

The updates for the precision parameters $\lambda_k^{[\nu]}$ of the projection vector read

$$\begin{aligned}
\log q^*(\lambda_k^{[\nu]}) &= < \log p(\boldsymbol{m}_k^{[\nu]} | \lambda_k^{[\nu]}) p(\lambda_k^{[\nu]} | c_k^{[0]}, d_k^{[0]}) >_{\boldsymbol{m}_k^{[\nu]}} \\
&= < \log \mathcal{N}(\boldsymbol{m}_k^{[\nu]} | \boldsymbol{0}, (\lambda_k^{[\nu]})^{-1} \boldsymbol{I}_V) \Gamma(\lambda_k^{[\nu]} | c_k^{[0]}, d_k^{[0]}) >_{\boldsymbol{m}_k^{[\nu]}}
\end{aligned}$$

$$
\begin{aligned}
=&\frac{d}{2}\log 2\pi + \frac{d}{2}\log \lambda_k^{[v]} - < \frac{\lambda_k^{[v]}}{2} \boldsymbol{m}_k^{[v]^T} \boldsymbol{m}_k^{[v]} >_{\boldsymbol{m}_k^{[v]}} \\
&- \Gamma(c_k^{[0]}) + c_k^{[0]} \log d_k^{[0]} + (c_k^{[0]} - 1)\log \lambda_k^{[v]} - d_k^{[0]} \lambda_k^{[v]} \\
=&(c_k^{[0]} + \frac{d}{2} - 1)\log \lambda_k^{[v]} - (d_k^{[0]} + \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}{}^T \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + d\sigma_{\boldsymbol{m}_k^{[v]}})\lambda_k^{[v]} + const
\end{aligned}
$$

This is a Gamma distribution. We get the optimal solution

$$
\begin{aligned}
q^*(\lambda_k^{[v]}) =& \Gamma(\lambda_k^{[v]} | \bar{c}_k, \bar{d}_k), \text{ with} \\
\bar{c}_k =& c_k^{[0]} + \frac{d}{2}, \\
\bar{d}_k =& d_k^{[0]} + \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}{}^T \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + d\sigma_{\boldsymbol{m}_k^{[v]}}).
\end{aligned}
$$

The expectation of the precision parameters $\bar{\lambda}_k^{[v]}$ is given by

$$
\bar{\lambda}_k^{[v]} = < \lambda_k^{[v]} >_{\lambda_k^{[v]}} = \frac{\bar{c}_k}{\bar{d}_k}.
$$

The updates for precision parameter of the weight vector α_k read

$$
\begin{aligned}
\log q^*(\alpha_k) =& < \log \prod_{i=1}^{L} p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) p(\alpha_k | a_k^{[0]}, b_k^{[0]}) >_{\boldsymbol{Z} \setminus \alpha_k} \\
=& < \log \prod_{i=1}^{L} \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1} \boldsymbol{I}_d)^{z_k^{[i]}} \Gamma(\alpha_k | a_k^{[0]}, b_k^{[0]}) >_{\boldsymbol{Z} \setminus \alpha_k} \\
=& \sum_{i=1}^{L} < z_k^{[i]} [\frac{d}{2}\log 2\pi\alpha_k - \frac{\alpha_k}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]})] >_{\boldsymbol{Z} \setminus \alpha_k} \\
&- \Gamma(a_k^{[0]}) + a_k^{[0]} \log b_k^{[0]} + (a_k^{[0]} - 1)\log \alpha_k - b_k^{[0]} \alpha_k \\
=& \sum_{i=1}^{L} \mu_{z_k^{[i]}} [\frac{d}{2}\log \alpha_k - \frac{\alpha_k}{2} [(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) \\
&+ tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{b_k} \boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} \\
&+ tr[\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}]] + (a_k^{[0]} - 1)\log \alpha_k - b_k^{[0]} \alpha_k + const \\
=& (a_k^{[0]} + \frac{d}{2}\sum_{i=1}^{L} \{\mu_{z_k^{[i]}}\} - 1)\log \alpha_k - (b_k^{[0]} + \frac{1}{2}\sum_{i=1}^{L} \{(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T \\
&(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{b_k} \boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} \\
&+ tr[(\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}])\alpha_k + const
\end{aligned}
$$

This is a Gamma distribution

$$
\begin{aligned}
q^*(\alpha_k) =& \Gamma(\alpha_k | \bar{a}_k, \bar{b}_k), \text{ with} \\
\bar{a}_k =& a_k^{[0]} + \frac{d}{2}\sum_{i=1}^{L} \{\mu_{z_k^{[i]}}\},
\end{aligned}
$$

$$\bar{b}_k = b_k^{[0]} + \frac{1}{2}\sum_{i=1}^{L}\{(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[(\bar{\boldsymbol{M}}_k^T\bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}].$$

The expectation of the precision parameters $\bar{\alpha}_k$ is given by

$$\bar{\alpha}_k = <\alpha_k>_{\alpha_k} = \frac{\bar{a}_k}{\bar{b}_k}.$$

The updates for the mixture indices $\boldsymbol{z}$ read

$$\begin{aligned}\log q^*(z) &= <\log\prod_{i=1}^{L}\prod_{k=1}^{K} p(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]})p(z_k^{[i]}|\pi_k)>_{\boldsymbol{Z}\setminus\boldsymbol{z}} \\ &= <\log\prod_{i=1}^{L}\prod_{k=1}^{K}\mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{b}_k + \boldsymbol{M}_k\boldsymbol{h}_k^{[i]}, (\alpha_k)^{-1}\boldsymbol{I}_d)^{z_k^{[i]}}(\pi_k)^{z_k^{[i]}}>_{\boldsymbol{Z}\setminus\boldsymbol{z}} \\ &= \sum_{i=1}^{L}\sum_{k=1}^{K} z_k^{[i]}\log\rho_k^{[i]},\end{aligned}$$

where we define

$$\begin{aligned}\log\rho_k^{[i]} &= \log\pi_k + <\frac{d}{2}\log\alpha_k - \frac{\alpha_k}{2}(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})^T(\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k\boldsymbol{h}_k^{[i]})>_{\boldsymbol{Z}\setminus\boldsymbol{z}} \\ &= \log\pi_k + \frac{d}{2}(\Psi(\bar{a}_k) - \log\bar{b}_k) - \frac{\bar{\alpha}_k}{2}[(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) \\ &\quad + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[(\bar{\boldsymbol{M}}_k^T\bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}]].\end{aligned}$$

With taking the exponential on both sides above, we get

$$q^*(z) \propto \prod_{i=1}^{L}\prod_{k=1}^{K}(\rho_k^{[i]})^{z_k^{[i]}}$$

We normalize for each i the quantities since they must sum to one because it is a probability distribution. Summarizing the variational factor for the mixture indices z is a multinomial and reads

$$\begin{aligned}q^*(z) &= \prod_{i=1}^{L}\prod_{k=1}^{K}(\mu_{z_k^{[i]}})^{z_k^{[i]}}, \text{ with} \\ \mu_{z_k^{[i]}} &= \frac{\rho_k^{[i]}}{\sum_{j=1}^{K}\rho_j^{[i]}}, \\ \rho_k^{[i]} &= \exp(\log\pi_k + \frac{d}{2}(\Psi(\bar{a}_k) - \log\bar{b}_k) - \frac{\bar{\alpha}_k}{2}[(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\mu_{h_k^{[v,i]}})^T \\ &\quad (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k\mu_{h_k^{[v,i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} \\ &\quad + tr[(\bar{\boldsymbol{M}}_k^T\bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}]].\end{aligned}$$

Finally the mixture coefficients π_k are optimized. Since we want to keep the model simple we did not place a prior on the mixture coefficients they are optimized as

$$\pi_k = \frac{1}{L} \sum_{i=1}^{L} z_k^{[i]}.$$

In future work one could consider a Dirichlet prior on the mixture coefficients and evaluate how this additional prior increases the performance of the proposed model.

C Lower bound - LMProMPs

In this section we derive the lower bound $\mathscr{L}$ for the Latent Manifold Probabilistic Movement Primitives (LMProMPs). We start with the proposed model for single movement types and conclude with the model of multiple movement types.

C.1 Single movement type

For a reminder the latent variables of the LMProMP for single movement types reads

$$\xi = \{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \alpha, \lambda^{[1:V]}\}.$$

The lower bound is computed as follows:

$$\begin{aligned}
\mathscr{L}(q) &= < \log p(\boldsymbol{y}^{[1:L]}, \xi) > - < \log q(\xi) > \\
&= \sum_{i=1}^{L} \{< \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}) > + < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}, \boldsymbol{M}, \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}, \alpha) > + < \log p(\boldsymbol{h}^{[i]}) >\} \\
&\quad + < \log p(\boldsymbol{b}) > + < \log p(\boldsymbol{M}) > + < \log p(\alpha) > + < \log p(\lambda^{[1:V]}) > \\
&\quad + \sum_{i=1}^{L} \{- < \log q(\boldsymbol{w}^{[i]}) > - < \log q(\boldsymbol{h}^{[i]}) >\} - < \log q(\boldsymbol{b}) > - < \log q(\boldsymbol{M}) > \\
&\quad - < \log q(\alpha) > - < \log q(\lambda^{[1:V]}) > .
\end{aligned}$$

We left out the parameters $\theta = \{\beta, \lambda^{[0]}, \gamma, a^{[0]}, b^{[0]}, c^{[0]}, d^{[0]}\}$ to keep the notation uncluttered, as well as the subscripts of the expectation, because each expectation is taken w.r.t all the variational posterior distributions of the corresponding latent variables.
In the following we derive all of the summands separately. All terms which does not change during the updates are denoted as constant by the term $const$ and are not considered in the lower bound.

$$\begin{aligned}
< \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}) > &= < \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}, \beta) > = < \log \mathcal{N}(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]}, \beta^{-1} \boldsymbol{I}_S) > \\
&= \frac{S}{2} \log 2\pi\beta - < \frac{\beta}{2} (\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]})^T (\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]}) > \\
&= -\frac{\beta}{2} (-2 \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}{}^T \boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{y}^{[i]} + \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}{}^T \boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{\Psi}^{[i]} \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} \\
&\quad + tr[\boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{\Psi}^{[i]} \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}]) + const
\end{aligned}$$

$$\begin{aligned}
< \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}, \boldsymbol{M}, \boldsymbol{h}^{[i]}, \alpha) > &= < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, \alpha \boldsymbol{I}_d) > \\
&= < \log \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, \alpha \boldsymbol{I}_d) > \\
&= \frac{d}{2} \log 2\pi\alpha - < \frac{\alpha}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b} - \boldsymbol{M}\boldsymbol{h}^{[i]}) >
\end{aligned}$$

$$
\begin{aligned}
&= \frac{d}{2} < \log \alpha >_{\alpha} - \frac{\alpha}{2}[(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) \\
&\quad + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}}\boldsymbol{I}_d] \\
&\quad + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}} + \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}] + const \\
&= \frac{d}{2}(\mathscr{K}(\bar{a}) - \log \bar{b}) - \frac{\bar{\alpha}}{2}[(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) \\
&\quad + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}}\boldsymbol{I}_d] + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} \\
&\quad + tr[(\bar{\boldsymbol{M}}^T\bar{\boldsymbol{M}} + \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}})\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}]] + const
\end{aligned}
$$

For more details see derivation of the update equation of $q^*(\alpha)$. Again $\mathscr{K}$ denotes the Digamma function.

$$
\begin{aligned}
< \log p(\boldsymbol{h}^{[i]}) > &= < \log p(\boldsymbol{h}^{[i]}|\gamma) > = < \log \mathscr{N}(\boldsymbol{h}^{[i]}|\boldsymbol{0}, \gamma^{-1}\boldsymbol{I}_V) > \\
&= \frac{V}{2}\log 2\pi\gamma - < \frac{\gamma}{2}\boldsymbol{h}^{[i]T}\boldsymbol{h}^{[i]} > \\
&= -\frac{\gamma}{2}(\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}{}^T\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}]) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\boldsymbol{b}) > &= < \log p(\boldsymbol{b}|\lambda^{[0]}) > = < \log \mathscr{N}(\boldsymbol{b}|\boldsymbol{0}, \lambda^{[0]^{-1}}\boldsymbol{I}_d) > \\
&= \frac{d}{2}\log 2\pi\lambda^{[0]} - < \frac{\lambda^{[0]}}{2}\boldsymbol{b}^T\boldsymbol{b} > \\
&= -\frac{\lambda^{[0]}}{2}(\boldsymbol{\mu}_{\boldsymbol{b}}{}^T\boldsymbol{\mu}_{\boldsymbol{b}} + tr[\sigma_{\boldsymbol{b}}\boldsymbol{I}_d]) + const \\
&= -\frac{\lambda^{[0]}}{2}(\boldsymbol{\mu}_{\boldsymbol{b}}{}^T\boldsymbol{\mu}_{\boldsymbol{b}} + d\sigma_{\boldsymbol{b}}) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\boldsymbol{M}) >_{\boldsymbol{M}} &= < \log \prod_{v=1}^{V} p(\boldsymbol{m}^{[v]}|\lambda^{[v]}) > = < \sum_{v=1}^{V} \log \mathscr{N}(\boldsymbol{m}^{[v]}|\boldsymbol{0}, \lambda^{[v]^{-1}}\boldsymbol{I}_d) > \\
&= \sum_{v=1}^{V} \frac{d}{2}\log 2\pi\lambda^{[v]} - < \frac{\lambda^{[v]}}{2}\boldsymbol{m}^{[v]T}\boldsymbol{m}^{[v]} > \\
&= \sum_{v=1}^{V} -\frac{\lambda^{[v]}}{2}(\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}{}^T\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} + tr[\sigma_{\boldsymbol{m}^{[v]}}\boldsymbol{I}_d]) + const \\
&= \sum_{v=1}^{V} -\frac{\lambda^{[v]}}{2}(\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}{}^T\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} + d\sigma_{\boldsymbol{m}^{[v]}}) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\alpha) > &= < \log p(\alpha|a^{[0]}, b^{[0]}) > = < \log \Gamma(\alpha|a^{[0]}, b^{[0]}) > \\
&= < -\Gamma(a^{[0]}) + a^{[0]}\log b^{[0]} + (a^{[0]} - 1)\log\alpha - b^{[0]}\alpha > \\
&= < (a^{[0]} - 1)\log\alpha - b^{[0]}\alpha > + const \\
&= (a^{[0]} - 1)(\mathscr{K}(\bar{a}) - \log\bar{b}) - b^{[0]}\frac{\bar{a}}{\bar{b}} + const
\end{aligned}
$$

$$
< \log p(\lambda^{[v]}) > = < \log p(\lambda^{[v]}|c^{[0]}, d^{[0]}) > = < \log \Gamma(\lambda^{[v]}|c^{[0]}, d^{[0]}) >
$$

$$
\begin{aligned}
&= < -\Gamma(c^{[0]}) + c^{[0]} \log d^{[0]} + (c^{[0]} - 1) \log \lambda^{[\nu]} - d^{[0]} \lambda^{[\nu]} > \\
&= < (c^{[0]} - 1) \log \lambda^{[\nu]} - d^{[0]} \lambda^{[\nu]} > + const \\
&= (c^{[0]} - 1)(\mathcal{K}(\bar{c}) - \log \bar{d}) - d^{[0]} \frac{\bar{c}}{\bar{d}} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{w}^{[i]}) > &= < \log \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} < (\boldsymbol{w}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} (\boldsymbol{w}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} [(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} \\
&\quad (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}) + tr[(\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} tr[\boldsymbol{I}_d] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| + const
\end{aligned}
$$

Here we need to compute the log of the determinante which might result in numerical issues. We therefore rewrite the term by

$$
\log |X| = \log \prod_j \lambda_j = \sum_j \log \lambda_j,
$$

where λ_i is the i-th eigenvalue of X. We obtain

$$
< \log q(\boldsymbol{w}^{[i]}) > = -\frac{1}{2} \sum_j^d \log |2\pi \lambda_j| + const
$$

$$
\begin{aligned}
< \log q(\boldsymbol{h}^{[i]}) > &= < \log \mathcal{N}(\boldsymbol{h}^{[i]} | \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}| - \frac{1}{2} < (\boldsymbol{h}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}})^{-1} (\boldsymbol{h}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}| - \frac{1}{2} tr[\boldsymbol{I}_V] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}| + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{b}) > &= < \log \mathcal{N}(\boldsymbol{b} | \boldsymbol{\mu}_{\boldsymbol{b}}, \sigma_{\boldsymbol{b}} \boldsymbol{I}_d) > \\
&= \frac{d}{2} \log 2\pi\sigma_{\boldsymbol{b}} - \frac{1}{2} < (\boldsymbol{b} - \boldsymbol{\mu}_{\boldsymbol{b}})^T (\sigma_{\boldsymbol{b}})^{-1} \boldsymbol{I}_d (\boldsymbol{b} - \boldsymbol{\mu}_{\boldsymbol{b}}) > \\
&= \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{b}}} - \frac{1}{2} tr[\boldsymbol{I}_d] + const \\
&= \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{b}}} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{M}) >_M &= < \log \prod_{v=1}^{V} \mathcal{N}(\boldsymbol{m}^{[v]} | \boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}, \sigma_{\boldsymbol{m}^{[v]}} \boldsymbol{I}_d) > \\
&= \sum_{v=1}^{V} \frac{d}{2} \log 2\pi\sigma_{\boldsymbol{m}^{[v]}} - \frac{1}{2} < (\boldsymbol{m}^{[v]} - \boldsymbol{\mu}_{\boldsymbol{m}^{[v]}})^T (\sigma_{\boldsymbol{m}^{[v]}})^{-1} \boldsymbol{I}_d (\boldsymbol{m}^{[v]} - \boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}) > \\
&= \sum_{v=1}^{V} \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{m}^{[v]}}} + \frac{1}{2} tr[\boldsymbol{I}_d] + const \\
&= \sum_{v=1}^{V} \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{m}^{[v]}}} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\alpha) > &= < \log \Gamma(\alpha | \bar{a}, \bar{b}) > \\
&= < -\log \Gamma(\bar{a}) + \bar{a} \log \bar{b} + (\bar{a} - 1) \log \alpha - \bar{b}\alpha > \\
&= -\log \Gamma(\bar{a}) + \bar{a} \log \bar{b} + (\bar{a} - 1)(\mathcal{K}(\bar{a}) - \log \bar{b}) - \bar{b} \frac{\bar{a}}{\bar{b}} \\
&= -\log \Gamma(\bar{a}) + \log \bar{b} + (\bar{a} - 1)\mathcal{K}(\bar{a}) - \bar{a}
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\lambda^{[v]}) > &= < \log \Gamma(\lambda^{[v]} | \bar{c}, \bar{d}) > \\
&= < -\log \Gamma(\bar{c}) + \bar{c} \log \bar{d} + (\bar{c} - 1) \log \lambda^{[v]} - \bar{d}\lambda^{[v]} > \\
&= -\log \Gamma(\bar{c}) + \bar{c} \log \bar{d} + (\bar{c} - 1)(\mathcal{K}(\bar{c}) - \log \bar{d}) - \bar{d} \frac{\bar{c}}{\bar{d}} \\
&= -\log \Gamma(\bar{c}) + \log \bar{d} + (\bar{c} - 1)\mathcal{K}(\bar{c}) - \bar{c}
\end{aligned}
$$

For multiple movements we obtain an extended version of the lower bound, since the joint distribution contains multiple mixture components.

$$
\begin{aligned}
\mathcal{L}(q) &= < \log p(\boldsymbol{y}^{[1:L]}, \xi) > - < \log q(\xi) > \\
&= \sum_{i=1}^{L} \{ < \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}) > + \sum_{k=1}^{K} \{ < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k, z_k^{[i]}) > \\
&\quad + < \log p(\boldsymbol{h}_k^{[i]} | \gamma) > + < \log p(z_k^{[i]}) > \} \} \\
&\quad + \sum_{k=1}^{K} \{ < \log p(\boldsymbol{b}_k) > + < \log p(\boldsymbol{M}_k | \lambda_k^{[1:V]}) > + < \log pk(\lambda_k^{[1:V]}) > + < \log p(\alpha_k) > \} \\
&\quad + \sum_{i=1}^{L} \{ - < \log q(\boldsymbol{w}^{[i]}) > - \sum_{k=1}^{K} \{ < \log q(\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) > - < \log q(z_k^{[i]}) > \} \} \\
&\quad - \sum_{k=1}^{K} \{ < \log q(\boldsymbol{b}_k) > - < \log q(\boldsymbol{M}_k) > - < \log q(\alpha_k) > - < \log q(\lambda_k^{[1:V]}) > \}
\end{aligned}
$$

The individual terms read

$$
\begin{aligned}
< \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}) > &= < \log p(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]}, \boldsymbol{w}^{[i]}, \beta) > = < \log \mathcal{N}(\boldsymbol{y}^{[i]} | \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]}, \beta^{-1} \boldsymbol{I}_S) > \\
&= \frac{S}{2} \log 2\pi\beta - < \frac{\beta}{2} (\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]})^T (\boldsymbol{y}^{[i]} - \boldsymbol{\Psi}^{[i]} \boldsymbol{w}^{[i]}) > \\
&= - \frac{\beta}{2} (-2 \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}{}^T \boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{y}^{[i]} + \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}{}^T \boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{\Psi}^{[i]} \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} \\
&\quad + tr[\boldsymbol{\Psi}^{[i]}{}^T \boldsymbol{\Psi}^{[i]} \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}]) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k, \boldsymbol{M}_k, \boldsymbol{h}_k^{[i]}, \alpha_k) > &= < \log p(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, \alpha_k \boldsymbol{I}_d) > \\
&= < \log \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{b}_k + \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}, \alpha_k \boldsymbol{I}_d) > \\
&= \frac{d}{2} \log 2\pi\alpha_k - < \frac{\alpha_k}{2} (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]})^T (\boldsymbol{w}^{[i]} - \boldsymbol{b}_k - \boldsymbol{M}_k \boldsymbol{h}_k^{[i]}) > \\
&= \frac{d}{2} < \log \alpha_k >_{\alpha_k} - \frac{\alpha_k}{2} [(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T \\
&\quad (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k} \boldsymbol{I}_d] \\
&\quad + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T \boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[(\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + \boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}] + const \\
&= \frac{d}{2} (\mathcal{K}(\bar{a}_k) - \log \bar{b}_k) - \frac{\bar{\alpha}_k}{2} [(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T \\
&\quad (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + tr[\sigma_{\boldsymbol{b}_k} \boldsymbol{I}_d] \\
&\quad + \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T \boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[(\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + \boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}) \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}]] + const
\end{aligned}
$$

For more details see derivation of the update equation of $q^*(\alpha_k)$. Again $\mathcal{K}$ denotes the Digamma function.

$$
\begin{aligned}
< \log p(\boldsymbol{h}_k^{[i]}) > &= < \log p(\boldsymbol{h}_k^{[i]} | \gamma_k) > = < \log \mathcal{N}(\boldsymbol{h}_k^{[i]} | \boldsymbol{0}, \gamma_k^{-1} \boldsymbol{I}_V) > \\
&= \frac{V}{2} \log 2\pi\gamma_k - < \frac{\gamma_k}{2} \boldsymbol{h}_k^{[i]^T} \boldsymbol{h}_k^{[i]} > \\
&= - \frac{\gamma_k}{2} (\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}{}^T \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} + tr[\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}]) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\boldsymbol{b}_k) > &= < \log p(\boldsymbol{b}_k | \lambda_k^{[0]}) > = < \log \mathcal{N}(\boldsymbol{b}_k | \boldsymbol{0}, \lambda_k^{[0]^{-1}} \boldsymbol{I}_d) > \\
&= \frac{d}{2} \log 2\pi\lambda_k^{[0]} - < \frac{\lambda_k^{[0]}}{2} \boldsymbol{b}_k{}^T \boldsymbol{b}_k > \\
&= - \frac{\lambda_k^{[0]}}{2} (\boldsymbol{\mu}_{\boldsymbol{b}_k}{}^T \boldsymbol{\mu}_{\boldsymbol{b}_k} + tr[(\sigma_{\boldsymbol{b}_k})^{-1} \boldsymbol{I}_d]) + const \\
&= - \frac{\lambda_k^{[0]}}{2} (\boldsymbol{\mu}_{\boldsymbol{b}_k}{}^T \boldsymbol{\mu}_{\boldsymbol{b}_k} + d(\sigma_{\boldsymbol{b}_k})^{-1}) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\boldsymbol{M}_k) >_{\boldsymbol{M}_k} &= < \log \prod_{v=1}^{V} p(\boldsymbol{m}_k^{[v]} | \lambda_k^{[v]}) > = < \sum_{v=1}^{V} \log \mathcal{N}(\boldsymbol{m}_k^{[v]} | \boldsymbol{0}, \lambda_k^{[v]^{-1}} \boldsymbol{I}_d) > \\
&= \sum_{v=1}^{V} \frac{d}{2} \log 2\pi\lambda_k^{[v]} - < \frac{\lambda_k^{[v]}}{2} \boldsymbol{m}_k^{[v]^T} \boldsymbol{m}_k^{[v]} > \\
&= \sum_{v=1}^{V} - \frac{\lambda_k^{[v]}}{2} (\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}{}^T \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + tr[\sigma_{\boldsymbol{m}_k^{[v]}} \boldsymbol{I}_d]) + const \\
&= \sum_{v=1}^{V} - \frac{\lambda_k^{[v]}}{2} (\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}{}^T \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + d\sigma_{\boldsymbol{m}_k^{[v]}}) + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\alpha_k) > &= < \log p(\alpha_k | a_k^{[0]}, b_k^{[0]}) > = < \log \Gamma(\alpha_k | a_k^{[0]}, b_k^{[0]}) > \\
&= < -\Gamma(a_k^{[0]}) + a_k^{[0]} \log b_k^{[0]} + (a_k^{[0]} - 1) \log \alpha_k - b_k^{[0]} \alpha_k > \\
&= < (a_k^{[0]} - 1) \log \alpha_k - b_k^{[0]} \alpha_k > + const \\
&= (a_k^{[0]} - 1)(\mathcal{K}(\bar{a}_k) - \log \bar{b}_k) - b_k^{[0]} \frac{\bar{a}_k}{\bar{b}_k} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(\lambda_k^{[v]}) > &= < \log p(\lambda_k^{[v]} | c_k^{[0]}, d_k^{[0]}) > = < \log \Gamma(\lambda_k^{[v]} | c_k^{[0]}, d_k^{[0]}) > \\
&= < -\Gamma(c_k^{[0]}) + c_k^{[0]} \log d_k^{[0]} + (c_k^{[0]} - 1) \log \lambda_k^{[v]} - d_k^{[0]} \lambda_k^{[v]} > \\
&= < (c_k^{[0]} - 1) \log \lambda_k^{[v]} - d_k^{[0]} \lambda_k^{[v]} > + const \\
&= (c_k^{[0]} - 1)(\mathcal{K}(\bar{c}_k) - \log \bar{d}_k) - d_k^{[0]} \frac{\bar{c}_k}{\bar{d}_k} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log p(z_k^{[i]}) >_{z_k^{[i]}} &= < \log \pi_k^{z_k^{[i]}} > \\
&= < z_k^{[i]} \log \pi_k > \\
&= \mu_{z_k^{[i]}} \log \pi_k
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{w}^{[i]}) > &= < \log \mathcal{N}(\boldsymbol{w}^{[i]} | \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} < (\boldsymbol{w}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} (\boldsymbol{w}^{[i]} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} [(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} (\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}) \\
&\quad + tr[(\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})^{-1} \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| - \frac{1}{2} tr[\boldsymbol{I}_d] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}| + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{h}_k^{[i]}) > &= < \log \mathcal{N}(\boldsymbol{h}_k^{[i]} | \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}| - \frac{1}{2} < (\boldsymbol{h}_k^{[i]} - \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}})^T (\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}})^{-1} (\boldsymbol{h}_k^{[i]} - \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}) > \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}| - \frac{1}{2} tr[\boldsymbol{I}_V] \\
&= -\frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}}| + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{b}_k) > &= < \log \mathcal{N}(\boldsymbol{b}_k | \boldsymbol{\mu}_{\boldsymbol{b}_k}, \sigma_{\boldsymbol{b}_k} \boldsymbol{I}_d) > \\
&= \frac{d}{2} \log 2\pi \sigma_{\boldsymbol{b}_k} - \frac{1}{2} < (\boldsymbol{b}_k - \boldsymbol{\mu}_{\boldsymbol{b}_k})^T (\sigma_{\boldsymbol{b}_k})^{-1} \boldsymbol{I}_d (\boldsymbol{b}_k - \boldsymbol{\mu}_{\boldsymbol{b}_k}) > \\
&= \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{b}_k}} - \frac{1}{2} tr[\boldsymbol{I}_d] + const \\
&= \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{b}_k}} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\boldsymbol{M}_k) >_{\boldsymbol{M}_k} &= < \log \prod_{v=1}^{V} \mathcal{N}(\boldsymbol{m}_k^{[v]} | \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}, \sigma_{\boldsymbol{m}_k^{[v]}} \boldsymbol{I}_d) > \\
&= \sum_{v=1}^{V} \frac{d}{2} \log 2\pi \sigma_{\boldsymbol{m}_k^{[v]}} - \frac{1}{2} < (\boldsymbol{m}_k^{[v]} - \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}})^T (\sigma_{\boldsymbol{m}_k^{[v]}})^{-1} \boldsymbol{I}_d (\boldsymbol{m}_k^{[v]} - \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}) > \\
&= \sum_{v=1}^{V} \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{m}_k^{[v]}}} + \frac{1}{2} tr[\boldsymbol{I}_d] + const \\
&= \sum_{v=1}^{V} \frac{d}{2} \log \frac{1}{\sigma_{\boldsymbol{m}_k^{[v]}}} + const
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\alpha_k) > &= < \log \Gamma(\alpha_k | \bar{a}_k, \bar{b}_k) > \\
&= < -\log \Gamma(\bar{a}_k) + \bar{a}_k \log \bar{b}_k + (\bar{a}_k - 1) \log \alpha_k - \bar{b}_k \alpha_k > \\
&= -\log \Gamma(\bar{a}_k) + \bar{a}_k \log \bar{b}_k + (\bar{a}_k - 1)(\mathcal{K}(\bar{a}_k) - \log \bar{b}_k) - \bar{b}_k \frac{\bar{a}_k}{\bar{b}_k} \\
&= -\log \Gamma(\bar{a}_k) + \log \bar{b}_k + (\bar{a}_k - 1) \mathcal{K}(\bar{a}_k) - \bar{a}_k
\end{aligned}
$$

$$
\begin{aligned}
< \log q(\lambda_k^{[\nu]}) > &= < \log \Gamma(\lambda_k^{[\nu]} | \bar{c}_k, \bar{d}_k) > \\
&= < -\log \Gamma(\bar{c}_k) + \bar{c}_k \log \bar{d}_k + (\bar{c}_k - 1) \log \lambda_k^{[\nu]} - \bar{d}_k \lambda_k^{[\nu]} > \\
&= -\log \Gamma(\bar{c}_k) + \bar{c}_k \log \bar{d}_k + (\bar{c}_k - 1)(\mathscr{K}(\bar{c}_k) - \log \bar{d}_k) - \bar{d}_k \frac{\bar{c}_k}{\bar{d}_k} \\
&= -\log \Gamma(\bar{c}_k) + \log \bar{d}_k + (\bar{c}_k - 1)\mathscr{K}(\bar{c}_k) - \bar{c}_k
\end{aligned}
$$

$$
\begin{aligned}
< \log q(z_k^{[i]}) >_{z_k^{[i]}} &= < \log \mu_{z_k^{[i]}}^{z_k^{[i]}} > \\
&= < z_k^{[i]} \log \mu_{z_k^{[i]}} > \\
&= \mu_{z_k^{[i]}} \log \mu_{z_k^{[i]}}
\end{aligned}
$$