A MULTIVARIATE GAUSSIAN PROCESS FACTOR MODEL FOR HAND SHAPE DURING REACH-TO-GRASP MOVEMENTS

Lucia Castellanos[1], Vincent Q. Vu[2], Sagi Perel[1], Andrew B. Schwartz[3], Robert E. Kass[1]

[1] *Carnegie Mellon University,* [2] *Ohio State University,* [3] *University of Pittsburgh*

Supplementary Material

Figure 1 illustrates the experimental setting, some grasped objects and the glove with reflective markers used to record finger position while reach-to-grasp replications.

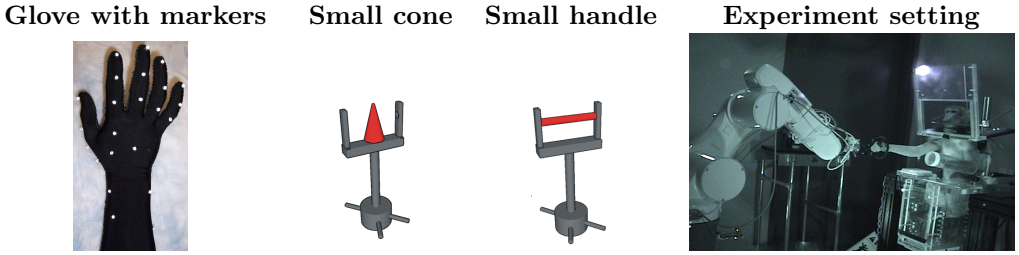

Figure 1: (*Left*) Custom made glove with 23 reflective markers, whose positions are tracked by an optical marker tracking system while the monkey performs the task. (*Center*) Two of the objects the monkey grasped, whose data we analyzed. (*Right*) Experiment setting showing the monkey sitting on the immobilizing chair, with the robot presenting the object to grasp.

S1 MGPFM: Symbol summary table, parametric assumptions and alogorithmic approach overview

Algorithm 1 summarizes our modelling and data analysis approach; Table 1 contains a description of the symbols used throughout the paper; and Table 2 lists the parametric assumptions.

input : Observed variables $\mathbb{Y}_{train}$
Low dimension d [Can be selected with BIC as in Section 4]
Parametric assumptions $\mathcal{P}$ for $\vec{\boldsymbol{\mu}}$, Σ and Ψ (Table 2)
Learning options: Initialization criteria ($\mathcal{I}$ = MLE based or down-projection),
Stop criteria ($\mathcal{S}$ = num. iterations or convergence threshold)
[Optional: Observed test variables $\mathbb{Y}_{test}$]

output: Inferred latent variables: $\hat{\mathbb{X}}_{train}$
Estimated parameters: $\hat{\Theta} = \{\hat{\vec{\boldsymbol{\mu}}}, \hat{\mathbf{B}}, \hat{\Sigma}, \hat{\Psi}\}$
[Optional: Inferred latent variables $\hat{\mathbb{X}}_{test}$]

1. Preprocessing: align observed data

input: $\mathbb{Y}_{train}$, [Optional: $\mathbb{Y}_{test}$]; **output:** $\mathbb{Y}^{aligned}_{train}$, [Optional: $\mathbb{Y}^{aligned}_{test}$]
method: Continuous registration
1) Summarize each trial with energy function (Equation 3)
2) Optimize `MINEIG` criterion (Section S4.1)

2. Estimation and inference:

2.1 Initialize parameter estimates (Section S2.4)

input: $\mathbb{Y}^{aligned}_{train}$, d, $\mathcal{I}$; **output:** $\hat{\Theta}_0$ (set $\hat{\Theta} = \hat{\Theta}_0$)
method:: Matrix normal MLE based estimation or down-projection strategy

2.2 Iterate until convergence criterion $\mathcal{C}$ is reached:

2.2.1 Hard E-step: **input:** $\mathbb{Y}^{aligned}_{train}$, $\hat{\Theta}$; **output:** $\hat{\mathbb{X}}_{train}$
$\mathbb{X}|\{\mathbb{Y}^{aligned}_{train}, \hat{\Theta}\} \sim \mathcal{N}(\eta, \Lambda)$; set $\hat{\mathbb{X}}_{train} = \eta$ as in Equation S2.12

2.2.2 M-step: **input:** $\mathbb{Y}^{aligned}_{train}$, $\hat{\mathbb{X}}$, parametric assumptions $\mathcal{P}$; **output:** $\hat{\Theta}$
Maximization of the model loglikelihood (Equation S2.3)
$\vec{\boldsymbol{\mu}}$ unconstrained (Equations S2.8, S2.9); $\vec{\boldsymbol{\mu}}$ splines (Equations S2.10, S2.11)
$\hat{\Sigma}$ unconstrained (Equation S2.6); $\hat{\Sigma}$ parametrized (see end of Section S2.2)
$\hat{\Psi}$ (Equation S2.7), $\hat{\mathbf{B}}$ (Equations S2.8-S2.11)

3. Predict new data:

input: $\mathbb{Y}^{aligned}_{test}$, $\hat{\Theta}$; **output:** $\hat{\mathbb{X}}^{aligned}_{test}$
method: Apply Hard-E step (Step 2.2.1)

Algorithm 1: MGPFM approach. Refer to Table 1 for symbol definitions and Table 2 for parametric assumptions.

Model: $\mathbf{Y}^r(t) = \boldsymbol{\mu}(t) + \mathbf{B}\mathbf{X}^r(t) + \boldsymbol{\epsilon}^r(t)$	
Symbol	*Description*
p	Number of kinematic variables representing a grasp.
d	Dimensionality of latent trajectory ($d < p$).
$t \in \{1, \dots, T\}$	Time.
$r \in \{1, \dots, R\}$	Replication or trial.
Observed variables: $\mathbb{Y} = \left\{ \{\mathbf{Y}^r(t)\}_{r=1}^R \right\}_{t=1}^T$	
$\mathbf{Y}^r(t) \in \mathbb{R}^{p \times 1}$	Kinematic p-dimensional grasping configuration at time t of replication r. $\mathbf{Y}^r(t) = \left[Y_1^r(t) \dots Y_p^r(t) \right]^\intercal$. The observed variables can be positional variables, but also velocity (denoted by a $\dot{Y}2_p(t)$) or other higher derivatives*.
Latent variables: $\mathbb{X} = \left\{ \{\mathbf{X}^r(t)\}_{r=1}^R \right\}_{t=1}^T$	
$\mathbf{X}^r(t) \in \mathbb{R}^{d \times 1}$	Latent factor d-dimensional trajectories $\mathbf{X}^r(t) = [X_1^r(t) \dots X_d^r(t)]^\intercal$ such that each X_j^r is drawn from an MGP with mean $\mathbf{0} \in \mathbb{R}^{d \times 1}$ and covariance function $\Sigma \in \mathbb{R}^{T \times T}$. These latent factors represent the low dimensional grasping structure that is specific to replication r.
Parameters being inferred: $\Theta = \{\vec{\boldsymbol{\mu}}, \boldsymbol{B}, \Sigma, \Psi\}$	
$\boldsymbol{\mu}(t) \in \mathbb{R}^{p \times 1}$	Deterministic mean p-dimensional function representing the kinematics that are common across replications, and that do not depend on a specific trial. Let $\boldsymbol{\mu}(t) = [\mu_1(t) \dots \mu_p(t)]^\intercal$ and $\vec{\boldsymbol{\mu}} = \{\boldsymbol{\mu}(t)\}_{t=1}^T$.
$\mathbf{B} \in \mathbb{R}^{p \times d}$	Deterministic factor loadings matrix whose columns correspond to the d latent factors and whose rows correspond to the p observed variables.
$\Sigma \in \mathbb{R}^{T \times T}$	Covariance matrix of the MGP; its form determines the temporal coherence properties of the low-dimensional representation.
$\Psi \in \mathbb{R}^{p \times p}$	Covariance of the normally distributed noise variables $\boldsymbol{\epsilon}^r(t) \in \mathbb{R}^{p \times 1}$, such that $\boldsymbol{\epsilon}^r(t) = \left[\epsilon_1^r(t) \dots \epsilon_p^r(t) \right]^\intercal$, where $\epsilon_j^r(t) \sim \mathcal{N}(\mathbf{0}, \Psi)$.

Table 1: Summary of symbols and variables. *We use $\dot{X}$ to denote the latent variables corresponding to fitting our model with the observed velocities $\dot{Y}$. The dimensions of the velocity variables correspond to the associated positional variables, that is, the dimensions of Y and X correspond to the dimensions of $\dot{Y}$ and $\dot{X}$ respectively.

Parametric assumptions: $\Theta = \{\vec{\boldsymbol{\mu}}, \mathbf{B}, \Sigma, \Psi\}$	
Parameter	*Assumptions*
$\vec{\boldsymbol{\mu}} = \{\boldsymbol{\mu}(t) \vert \boldsymbol{\mu}(t) \in \mathbb{R}^{p \times 1}; t = 1, \dots, T\}$	p-dimensional varying function in time. Can be learned: • Completely unrestricted / non-parametric approach, represented as a $p \times T$ matrix (with $p \times T$ parameters to learn). The completely unrestricted approach can lead to over fitting, specially with small datasets. • Represented with a $\mathcal{B}$-spline basis with c ($c < p$) basis functions ($c \times p$ parameters to learn). Imposes smoothing constraints in addition to reducing the number of parameters to learn.
$\mathbf{B} \in \mathbb{R}^{p \times d}$	Deterministic factor loadings matrix, where d is the number of latent factors and p the number of observed variables. To ensure identifiability of the model we assume that $\mathbf{B}^\intercal\mathbf{B}$ is diagonal.
$\Sigma \in \mathbb{R}^{T \times T}$	Covariance matrix of the MGP; its form determines the temporal coherence properties of the low-dimensional representation. Can be learned: • Completely unrestricted / non-parametric approach, represented as a $T \times T$ matrix. The completely unrestricted approach can lead to over fitting, specially when there is not much data available • Imposing structure on Σ by assuming a parametric form, for instance a stationary exponential covariance function: $\Sigma(i,j) = \exp\left(\frac{-(i-j)^2}{\theta_\Sigma}\right)$ where θ_Σ controls the width of the diagonal that decays exponentially.
$\Psi \in \mathbb{R}^{p \times p}$	Covariance of the normally distributed noise variables $\boldsymbol{\epsilon}^r(t) \in \mathbb{R}^{p \times 1}$. For simplicity we assume $\Psi = \rho \cdot I_{p \times p}$ with $\rho > 0$.

Table 2: Parametric assumptions.

S2 MGPFM: Estimation and inference derivation

Consider the $p-$dimensional observed dataset:

$$\{\ Y_i^r(t) \mid i = 1, \dots, p;\ t = 1, \dots, T;\ r = 1, \dots, R\ \}$$

$Y_i^r(t)$ is the i^{th} coordinate at time t of the p-dimensional trajectory that is the r^{th} replication of an event, R is the number of repeated trials, T the number of time slices and p the number of observed variables.

Let $\mathbf{Y}^r(t) = \left[Y_1^r(t) \dots Y_p^r(t)\right]^\intercal$, $\boldsymbol{\mu}(t) = [\mu_1(t) \dots \mu_p(t)]^\intercal$, $\mathbf{X}^r(t) = [X_1^r(t) \dots X_d^r(t)]^\intercal$ and $\boldsymbol{\epsilon}^r(t) = \left[\epsilon_1^r(t) \dots \epsilon_p^r(t)\right]^\intercal$. Then we can write the MGPFM as:

$$\mathbf{Y}^r(t) = \boldsymbol{\mu}(t) + \mathbf{B}\mathbf{X}^r(t) + \boldsymbol{\epsilon}^r(t). \tag{S2.1}$$

Alternatively, we model the mean trajectory $\boldsymbol{\mu}(t)$ with c ($<< T$) B-spline basis functions for each of the p variables. In this case $\boldsymbol{\mu}$ is described through B-splines:

$$\mathbf{Y}^r(t) = \boldsymbol{\mu}^{\mathtt{S}}(t) + \mathbf{B}\mathbf{X}^r(t) + \boldsymbol{\epsilon}^r(t), \qquad \boldsymbol{\mu}^{\mathtt{S}}(t) = \boldsymbol{\alpha} \cdot (\mathcal{S}(t))^\intercal \in \mathbb{R}^{p \times 1}, \tag{S2.2}$$

where $\mathcal{S} \in \mathbb{R}^{T \times c}$ is a matrix that holds the c spline basis functions, $\mathcal{S}(t)$ corresponds to one row of $\mathcal{S}$ and $\boldsymbol{\alpha} \in \mathbb{R}^{p \times c}$ contains the coefficients for each of the spline basis functions. This formulation drastically reduces the number of parameters to be estimated, while imposing a smoothing constraint on the learned functions.

S2.1 Loglikelihood

Denote $\vec{\boldsymbol{\mu}} = \{\boldsymbol{\mu}(t)\}_{t=1}^T$, $\mathbb{Y} = \left\{\{\mathbf{Y}^r(t)\}_{r=1}^R\right\}_{t=1}^T$ and $\mathbb{X} = \left\{\{\mathbf{X}^r(t)\}_{r=1}^R\right\}_{t=1}^T$. Then the joint distribution can be written as:

$$\mathbb{P}(\mathbb{Y}, \mathbb{X} | \vec{\boldsymbol{\mu}}, \mathbf{B}, \Psi, \Sigma) = \mathbb{P}(\mathbb{Y} | \mathbb{X}; \vec{\boldsymbol{\mu}}, \mathbf{B}, \Psi) \cdot \mathbb{P}(\mathbb{X} | \Sigma).$$

And the loglikelihood consists of two terms:

$$log\, \mathbb{P}(\mathbb{Y}, \mathbb{X} | \vec{\boldsymbol{\mu}}, \mathbf{B}, \Psi, \Sigma) = log\, \mathbb{P}(\mathbb{Y} | \mathbb{X}; \vec{\boldsymbol{\mu}}, \mathbf{B}, \Psi) \ + \ log\, \mathbb{P}(\mathbb{X} | \Sigma). \tag{S2.3}$$

If we simplify the model by defining $\Psi = \rho \cdot \mathbf{I}_{p \times p}$ with $\rho > 0$ then the first term corresponds to:

$$\begin{gathered} log\, \mathbb{P}(\mathbb{Y} | \mathbb{X}; \vec{\boldsymbol{\mu}}, \mathbf{B}, \rho) \\ = -\tfrac{1}{2} \textstyle\sum_{r=1}^R \sum_{k=1}^p \sum_{i=1}^T \frac{1}{\rho} \left(\mathbf{Y}_k^r(t_i) - \left[\mu_k(t_i) + \sum_{w=1}^d b_{k,w} X_w^r(t_i)\right]\right)^2 \\ -\tfrac{R \cdot p}{2} T \cdot log\, \rho \ - \tfrac{1}{2}\, p \cdot R \cdot T\, log\, 2\pi. \end{gathered} \tag{S2.4}$$

The second term of the loglikelihood corresponds to the distribution of the d iid MGPs indexed by s (denoted $\mathbf{X}_s \in \mathbb{R}^{T \times 1}$) given the covariance function $\Sigma(t_i, t_j)$:

$$log\, \mathbb{P}(\mathbb{X} | \Sigma) = -\frac{1}{2} \sum_{r=1}^R \sum_{s=1}^d X_s^{r\,\intercal} \Sigma^{-1} X_s^r - \frac{1}{2}\, d \cdot R\, log |\Sigma| - \frac{1}{2}\, d \cdot R \cdot T\, log\, 2\pi. \tag{S2.5}$$

In sum, if we consider the covariance simplifications then the loglikelihood of the model is given by the sum of expressions in Equation (S2.4) and (S2.5).

We can take Equation (S2.3) and use it as a loss function to learn the components of the model. Our approach is EM-based, in which we iterate between:

1. *Estimation problem*: Assuming that $\mathbb{X}$ are known, learn parameters $\vec{\boldsymbol{\mu}} \in \mathbb{R}^{p\times T}, \mathbf{B} \in \mathbb{R}^{p\times d}, \rho \in \mathbb{R}, \Sigma \in \mathbb{R}^{T\times T}$ which jointly constitute the parameter space.
2. *Inference problem*: Assuming that $\mathbb{Y}$ and all the parameters known, estimate the latent variables $\mathbb{X}$.

Iterations are stopped either by convergence of the loglikelihood or by number of iterations. In the experiments we noted that 50 iterations sufficed to reach convergence. Our approach can be thought of as *Hard EM* – in conventional EM, one computes a *soft* posterior distribution in the E-step; in hard EM, we simply maximize the posterior. For example, K-means can be seen as the Hard EM based algorithm for fitting Gaussian Mixture Models.

S2.2 Estimation problem

In the first problem we assume that the latent space trajectories $\mathbb{X}$ are known and we estimate the parameters $\vec{\boldsymbol{\mu}}, \mathbf{B}, \rho, \Sigma$. Here we will be maximizing the loglikelihood with respect to the parameters.

Note that to learn $\vec{\boldsymbol{\mu}} \in \mathbb{R}^{p\times T}, \mathbf{B} \in \mathbb{R}^{p\times d}$, and $\rho \in \mathbb{R}$ we only need the first term of Equation (S2.3), that is, Equation (S2.4). And to estimate Σ we only need the second term of Equation (S2.3), namely Equation (S2.5).

To estimate Σ we consider Equation (S2.5), let $\Omega = \Sigma^{-1}$ and perform standard optimization to learn the covariance function of the multivariate normal, obtaining:

$$\hat{\Sigma} = \frac{\sum_{r=1}^{R}\sum_{s=1}^{d} X_s^r \cdot X_s^{r\intercal}}{d\,R}. \tag{S2.6}$$

Estimating ρ from Equation (S2.4) is independent from estimating $\vec{\boldsymbol{\mu}}$ and $\mathbf{B}$. Differentiating Equation (S2.4) with respect to ρ and equating to zero we obtain:

$$\hat{\rho} = \frac{\sum_{r=1}^{R}\sum_{k=1}^{p}\sum_{i=1}^{T}\left(\mathbf{Y}_k^r(t_i) - \left[\hat{\mu}_k(t_i) + \sum_{w=1}^{d}\hat{b}_{k,w}X_w^r(t_i)\right]\right)^2}{RpT}. \tag{S2.7}$$

Maximizing Equation (S2.4) with respect to $\vec{\boldsymbol{\mu}}$ and $\mathbf{B}$ is equivalent to separately maximizing each term for a fixed $k = 1, \ldots, p$ and, in fact, corresponds to performing p multiple linear regressions.

Consider the data vector $\mathcal{Y}_k$, the design matrix $\mathcal{W}$, and the variables to learn β_k defined as follows: $\mathcal{Y}_k{}^{\intercal} = \left[\left[Y_k^1(t_1),\dots,Y_k^1(t_T)\right]^{\intercal},\dots,\left[Y_k^R(t_1),\dots,Y_k^R(t_T)\right]^{\intercal}\right] \in \mathbb{R}^{1\times(R\cdot T)}$,

$$\mathcal{W} = \begin{bmatrix} \begin{bmatrix} X_1^1(t_1) & \dots & X_d^1(t_1) \\ \vdots & \ddots & \vdots \\ X_1^1(t_T) & \dots & X_d^1(t_T) \end{bmatrix} & & I_{T\times T} \\ & \vdots & \vdots \\ \begin{bmatrix} X_1^R(t_1) & \dots & X_d^R(t_1) \\ \vdots & \ddots & \vdots \\ X_1^R(t_T) & \dots & X_d^R(t_T) \end{bmatrix} & & I_{T\times T} \end{bmatrix} \in \mathbb{R}^{R\cdot T\times(d+T)},$$

and

$$\beta_k{}^{\intercal} = [b_{k,1},\dots b_{k,d}\,,\ \mu_k(t_1),\dots,\mu_k(t_T)]^{\intercal} \in \mathbb{R}^{1\times(d+T)}. \tag{S2.8}$$

Then, we can consider the p independent linear regression models:

$$\mathcal{Y}_k = \mathcal{W}\cdot\beta_k, \quad k = 1,\dots,\ p. \tag{S2.9}$$

By solving these p linear regressions we can estimate the vectors β_k, from which we can read off the desired model parameters $\vec{\boldsymbol{\mu}} \in \mathbb{R}^{p\times T}$ and (after applying Gram Schmidt orthogonalization) $\mathbf{B} \in \mathbb{R}^{p\times d}$. In the case in which μ is assumed constant along time, the estimate corresponds to the mean across time of provided estimate.

Modelling $\boldsymbol{\mu}$ with splines

We have that the mean trajectory $\boldsymbol{\mu}$ can be written in a B-spline basis as follows:

$$\boldsymbol{\mu}^{\mathtt{S}}(t) = \boldsymbol{\alpha}\cdot(\mathcal{S}(t))^{\intercal} \in \mathbb{R}^{p\times 1}, \tag{S2.10}$$

where $\mathcal{S} \in \mathbb{R}^{T\times c}$ is a matrix that holds the c spline basis functions (one in each column). Vector $\mathcal{S}(t)$ corresponds to one row of $\mathcal{S}$. We are interested in learning $\boldsymbol{\alpha} \in \mathbb{R}^{p\times c}$ which contains the coefficients for each of the spline basis functions and describes the data.

Our goal is to formulate similar models to Equation S2.9 for each variable $k = 1,\dots,p$ but solving for the coefficients of the B-spline basis.

Instead of defining the auxiliary variable β_k as in Equation S2.8, we define: $\phi_k = [b_{k,1},\dots,b_{k,d},\alpha_k(1),\dots,\alpha_k(c)]^{\intercal} \in \mathbb{R}^{(d+c)\times 1}$, where k denotes the index of an observed variable $k = 1,\dots,p$ and $\alpha_k(i)$ denotes the coefficient of the i^{th} spline basis.

Equation S2.10 can be written as $[\mu_k(t_1),\dots,\mu_k(t_T)]^{\intercal} = [\alpha_k(1),\dots,\alpha_k(c)]^{\intercal}\cdot\mathcal{S}^{\intercal} \in \mathbb{R}^{1\times T}$ and we want to determine the values for $\{\alpha_k(i)\}_{i=1}^{c}$ for each $k \in \{1,\dots,p\}$. Note that $\beta_k = b_{k,1:d} \oplus \mu_k$ and $\phi_k = b_{k,1:d} \oplus \alpha_k$ where $\oplus$ denotes the stacking operation for vectors. Also $b_{k,1:d} = [b_{k,1},\dots,b_{k,d}]^{\intercal} \in \mathbb{R}^{d\times 1}$ corresponds to the k^{th} transposed row of the loading matrix $\boldsymbol{B} \in \mathbb{R}^{p\times d}$. Then: $\beta_k = (I_{d\times d} * b_{k,1:d}) \oplus (\mathcal{S}\cdot\alpha_k) = (I_{d\times d}\oplus\mathcal{S})\cdot(b_{k,1:d}\oplus\alpha_k) =$

$(I_{d\times d} \oplus \mathcal{S}) \cdot \phi_k$, and: $\mathcal{Y}_k = \mathcal{W} \cdot \beta_k = \mathcal{W} \cdot (I_{d\times d} \oplus \mathcal{S}) \cdot \phi_k$. Consequently, we have written an analogous problem as in Equations S2.9 to solve for the coefficients of the B-spline basis, namely:

$$\mathcal{Y}_k = \mathcal{W}^{\mathtt{S}} \cdot \phi_k, \qquad \mathcal{W}^{\mathtt{S}} = \mathcal{W} \cdot (I_{d\times d} \oplus \mathcal{S}) \tag{S2.11}$$

for $k = 1, \ldots, p$. And the problem reduces to solve for ϕ_k in a similar way as before.

Constraining Σ

In the estimation procedure we can either learn Σ *free* as in Equation S2.6, or learn θ_Σ, the univariate parameter that determines the covariance function of the MGP: $\Sigma(i,j) = \exp\left(\frac{-(i-j)^2}{\theta_\Sigma}\right)$. The latter can be done by gradient descent (numerically maximizing the loglikelihood) or through a one dimensional search over a space of reasonable values for θ_Σ. In our implementation we follow the last strategy.

S2.3 Inference problem

For the second problem (inference) we assume that the parameters $\mathbf{\Theta} = \{\vec{\boldsymbol{\mu}}, \mathbf{B}, \Psi, \Sigma\}$ are now known and we learn the hidden variables $\mathbb{X}$ by maximizing the posterior probability of $\mathbb{X}$ given $\mathbb{Y}$ and $\mathbf{\Theta}$. We observe that the vectorized elements of the latent factors for replication r (denoted by $\mathtt{vecX}^r \in \mathbb{R}^{(d\cdot T)\times 1}$) are distributed as $\mathcal{N}(\mathbf{0}, \Sigma \otimes I_{d\times d})$ where $\otimes$ denotes the Kronecker product of two matrices. Also, the vectorized difference of observed trajectory Y^r and mean μ given the latent factors of that replication are normally distributed:

$$(\mathtt{vecY}^r - \mathtt{vec}\mu) | \mathtt{vecX}^r \sim \mathcal{N}\left((I_{T\times T} \otimes \mathbf{B}) \cdot \mathtt{vecX}^r, \Psi \otimes I_{p\times p}\right).$$

Using standard properties of normal distributions we conclude that the posterior distribution of the latent factors given $\mathbb{Y}$ and $\mathbf{\Theta}$ is

$$\mathtt{vecX}^r | (\mathtt{vecY}^r - \mathtt{vec}\mu) \sim \mathcal{N}(\eta, \Lambda)$$

with:

$$\eta = \mathbf{0} + [(I_{T\times T} \otimes \mathbf{B}) \cdot (\Sigma \otimes I_{d\times d})]^{\mathtt{T}} \cdot \tag{S2.12}$$
$$\left[(I_{T\times T} \otimes \mathbf{B}) \cdot (\Sigma \otimes I_{d\times d}) \cdot (I_{T\times T} \otimes \mathbf{B})^{\mathtt{T}} + (\Psi \otimes I_{p\times p})\right]^{-1} \cdot$$
$$(\mathtt{vecY}^r - \mathtt{vec}\mu - \mathbf{0}),$$

and

$$\Lambda = [\Sigma \otimes I_{d\times d}] - \tag{S2.13}$$
$$[(I_{T\times T} \otimes \mathbf{B}) \cdot (\Sigma \otimes I_{d\times d})] \cdot \left[(I_{T\times T} \otimes \mathbf{B}) \cdot (\Sigma \otimes I_{d\times d}) \cdot (I_{T\times T} \otimes \mathbf{B})^{\mathtt{T}} + (\Psi \otimes I_{p\times p})\right]^{-1}$$
$$\cdot [(I_{T\times T} \otimes \mathbf{B}) \cdot (\Sigma \otimes I_{d\times d})]^{\mathtt{T}}$$

And the mean of a normal distribution maximizes the loglikelihood, therefore we set: $\hat{X} = \eta$. Note that the matrix we need to invert in this step is sparse and contains a lot of

structure that we can exploit to make computation efficiently. In particular, in Equation (S2.12), η is the product of two big matrices U and V and a vector w. Both U and V are sparse and we do not need to fully invert V, we only need to compute $V^{-1} \cdot w$. Hence, we can use sparse matrices to represent U and V, and we can efficiently calculate $V^{-1} \cdot w$ without explicitly inverting the matrix. In addition, matrices U and V are Kronecker products with the identity matrix, which is itself sparse, and thus we can represent it efficiently computationally.

S2.4 Initialization regimes

We considered two initialization regimes to avoid local optima of the loglikelihood: the matrix normal MLE and a down-projection strategy.

Matrix normal MLE based initialization.

A matrix $U \in \mathbb{R}^{p \times T}$ is said to be sampled from a matrix normal distribution $\mathcal{N}_{p,T}(M, F, G)$ with mean $M \in \mathbb{R}^{p \times T}$, among-row covariance matrix $F \in \mathbb{R}^{p \times p}$ and among-column covariance matrix $G \in \mathbb{R}^{T \times T}$ if its vectorized form $\texttt{vecU}$ is distributed as the multivariate normal: $\mathcal{N}_{p \cdot T}(\texttt{vecM}, F \otimes G)$. Conceivably the observed data generated with the MGPFM can be close to a matrix normal distribution or, at least, we can use this distribution for initialization purposes (for more details of the matrix normal distribution see (Dawid, 1981)). There are no analytical solutions for the MLE for the among-row and among-column covariance matrices of the matrix normal distribution. However, Dutilleul (1999) presents an iterative algorithm (also called *flip-flop* algorithm) to obtain the MLE of its three parameters (M, F, G). We propose to initialize the parameters of the MGPFM as follows: $\mu_0 = \tilde{M}$ and $\Sigma_0 = \tilde{G}$, where $\tilde{}$ denotes the MLE. To initialize $\mathbf{B}_0$ and ρ_0 we obtain the spectral decomposition of $\tilde{F}$. Intuitively, $\mathbf{B}_0$ contains the first d normalized eigenvectors of $\tilde{F}$ and ρ is the residual obtained by subtracting $\mathbf{B}_0 \cdot \mathbf{B}_0^{\mathsf{T}}$ from $\tilde{F}$. Let $D \in \mathbb{R}^{p \times p}$ be the diagonal matrix containing the decreasing eigenvalues of $\tilde{F}$ and let $E \in \mathbb{R}^{p \times p}$ contain in its columns the corresponding eigenvectors. We set $\mathbf{B}_0 = E_{:,1:d} \cdot \sqrt{D_{1:d,1:d}}$, and $\rho_0 = \sqrt{\frac{\sum_i \sum_j \check{e}_{i,j}^2}{p}}$ where $\check{e}_{i,j}$ is the (i,j) element of the matrix $\check{E}$ defined as $E_{:,d+1:p} \cdot \sqrt{D_{d+1:p,d+1:p}}$.

Down-projection based initialization.

In this second initialization approach we want to learn the model parameters when the latent dimension is d_{goal}, but begin with a higher dimensional problem.

1. We first run the MGPFM learning algorithm (as before) for a latent dimension d_{high} higher than desired i.e. $d_{goal} < d_{high}$ and obtain as an output the estimates: $\hat{\mu}_h \in \mathbb{R}^{p \times T}, \hat{\rho}_h \in \mathbb{R}, \hat{\Sigma}_h \in \mathbb{R}^{T \times T}$, and $\hat{\mathbf{B}}_h \in \mathbb{R}^{p \times d_{high}}$.

2. Project the estimated parameters to the target latent dimension d_{goal}. We note that only $\hat{\mathbf{B}}_h$ needs to be projected. We use the SVD decomposition of the matrix: $\hat{\mathbf{B}}_h = U \cdot S \cdot V^{\mathtt{T}}$ and define $\mathbf{B}_{proj} = \hat{\mathbf{B}}_h \cdot V_{:,1:d_{goal}}$.
3. Use the projected estimates as initial values for a second run of the MGPFM learning algorithm, that is, set: $\mu_0 = \hat{\mu}_h$, $\rho_0 = \hat{\rho}_h$, $\Sigma_0 = \hat{\Sigma}_h$ and $\mathbf{B}_0 = \mathbf{B}_{proj}$.

We find in practice that this second initialization method is often effective but also significantly more computationally expensive since it requires optimization in a higher dimension first.

S3 Supplement for simulation studies

We show results of two exemplary simluations. We generated $R = 80$ samples for training and 500 samples for testing. We used the following learning settings: we initialized the parameters with the MLE of the matrix-normal distribution, we assumed $\boldsymbol{\mu}$ to be modelled with B-splines and stopped the algorithm after 50 iterations. We learned three models: the first one modelling the observed data only with the mean (as a baseline), the last two corresponding to the MGPFM assuming Σ free and Σ constrained.

In Figure 2 (top panel) we show the true latent process X together with two dimensions of the observed Y and estimated $\hat{Y}$ for the two MGPFM models. Note that we obtain smoother estimates when constraining Σ. In the middle and bottom panels of Figure 2 we show error profiles for the three models. The baseline model (that disregards the MGP term) results in significantly worse estimates as compared to either setting for the MGPFM. In addition, by constraining Σ we are able to remove all unaccounted structure left in the residuals when modelling Σ free.

S4 Supplement for reach-to-grasp data analysis

S4.1 Alignment of kinematic curves

The MINEIG criterion for estimating warping functions is explained in (Ramsay and Silverman, 2005), and it is written as:

$$\texttt{MINEIG}(h) = \gamma_2 \cdot \texttt{det} \begin{pmatrix} \int E_0(t)^2 dt & \int E_0(t) E^r(h(t)) dt \\ \int E_0(t) E^r(h(t)) dt & \int E^r(h(t))^2 dt \end{pmatrix}, \tag{S4.1}$$

where $E_0(t)$ is the target curve and γ_2 is the size of the second smallest eigenvalue of the enclosed matrix. The basic idea is that the matrix is like the covariance matrix of the curves: $(\{E_0(t) : t\}, \{E^r(h(t)) : t\})$. If one of the curves is exactly proportional to the other then the matrix is singular and so $\texttt{MINEIG}(h) = 0$. The advantage of using this criterion is that there are well developed R and Matlab packages (`fda`; Ramsay et al. 2009) for minimizing the roughness penalized criterion:

$$\texttt{MINEIG}(h) + \lambda \int \{W^{(m)}(t)\}^2 dt, \tag{S4.2}$$

when h is of the form:

$$h(t) = C_0 + C_1 \int_0^t \texttt{exp}\, W(u) du \tag{S4.3}$$

with W expanded into a $\mathcal{B}$-spline basis and $C_0, C_1 \in \mathbb{R}$. The basic strategy for alignment then follows the iterative procedure known as *Procrustes method*:

1. Initialize the target $E_0(t) \leftarrow E^r(t)$ to some $E^r(t)$.
2. Repeat until convergence:
   (a) For each trial $r = 1, ..., R$ fit a time warping function $h^r(t)$ using criterion in Equation S4.2
   (b) Update the target $E_0(t) \leftarrow \frac{1}{R} \sum_r E^r(h^r(t))$

S4.2 Results of estimating the MGPFM in the grasping dataset

In Figure 3 we show the observed velocity profiles, the MGPFM estimates and the residuals decomposed per marker and finger for a specific replication for the small cone presented at 45° abduction. These plots, which are representative of the grasping behavior in the dataset, show that the thumb's amount of movement is very small as compared to the amount of movement by all the other fingers. The MGPFM captures most of the variation leaving residuals close to zero.

Figure 4 shows more visualizations of the columns of $\mathbf{B}$ under different experimental conditions. Unlike PCA, in which one obtains *canonical* directions of movement, these visualizations only exemplify the space of possible configurations of change of movement in the dataset. The loadings are orthogonal, but presented sorted based on their norm.

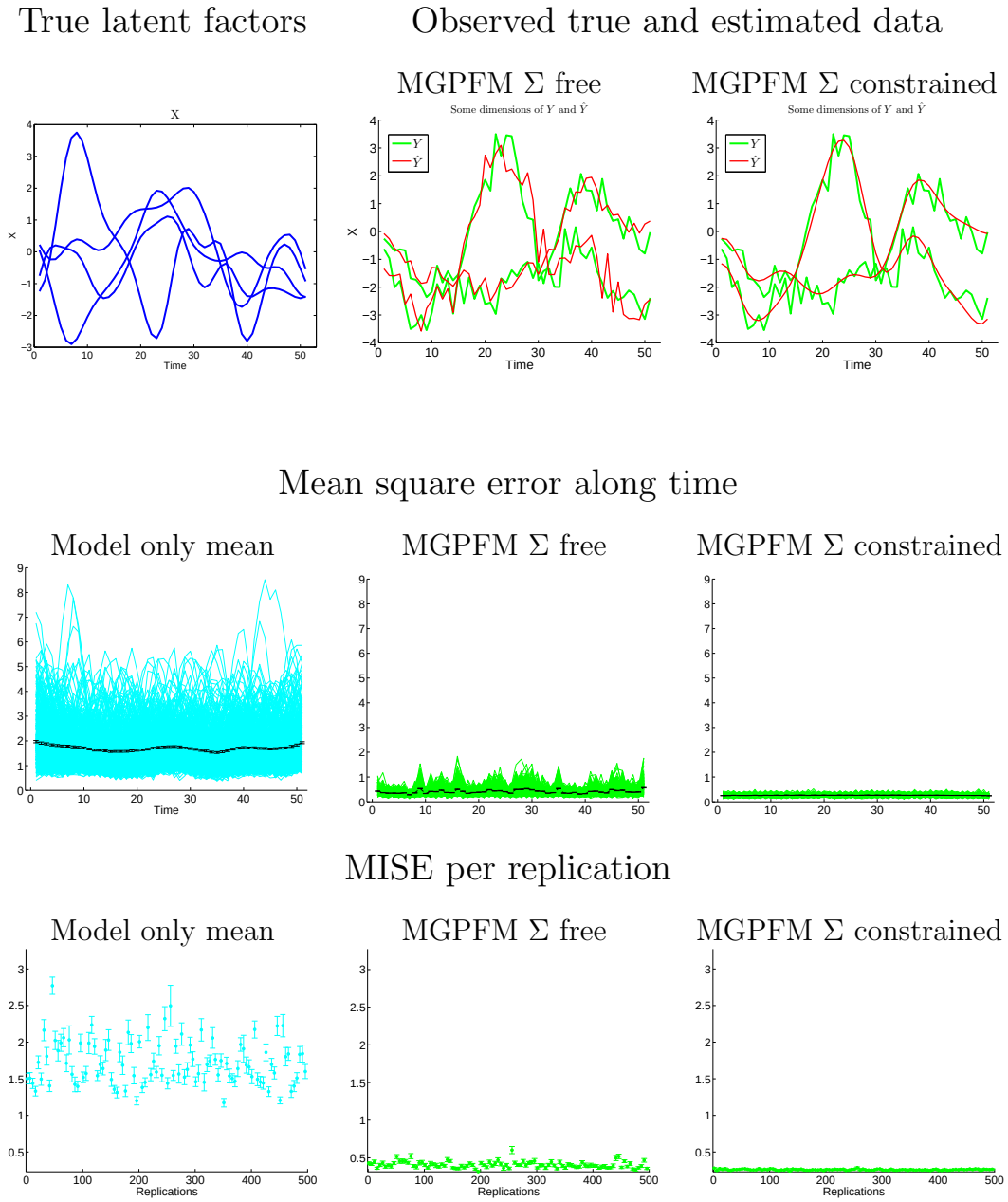

Figure 2: Results of two exemplary simulations. (Top panel) True latent process X, and two dimensions of the 50-dimensional Y and $\hat{Y}$. Estimates of $\hat{Y}$ are smoother when Σ is constrained. (Middle and bottom panel) Error profiles for three models: baseline when modelling only the mean (left), MGPFM with Σ free (middle) and MGPFM with Σ constrained (left). Best results are achieved with the MGPFM when Σ is constrained.

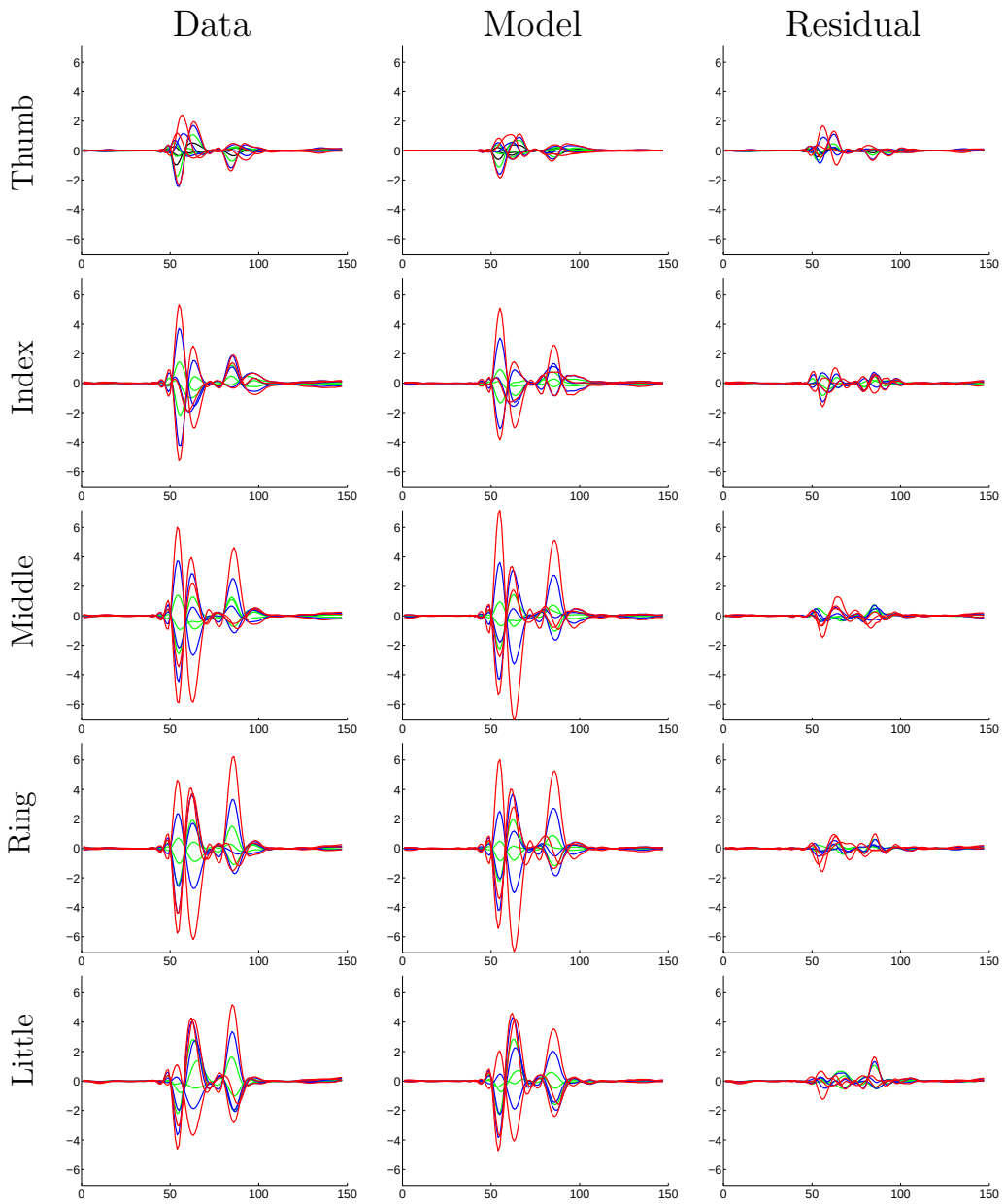

Figure 3: Observed velocity profiles, MGPFM fit and residuals for one replication of the subject grasping the small cone at 45° of abduction. Each row represents a finger and the trajectories corresponding to a specific marker are plotted in the same color: red for the most distal marker, green for the most proximal marker and blue for the maker in the middle. The thumb has a fourth marker plotted in black. The magnitude of motion in the thumb markers is very small as compared to the other fingers. The MGPFM estimates are very close to the observed data yielding residuals close to zero.

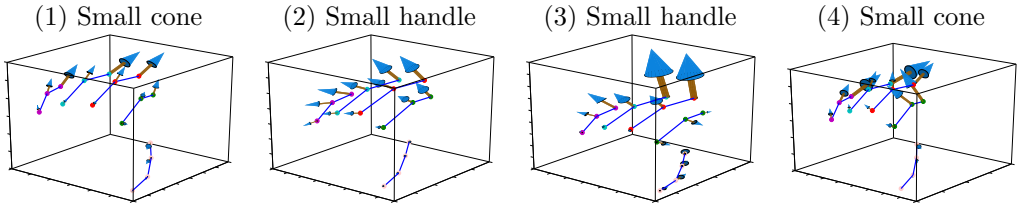

Figure 4: Visualization of the columns of the factor loading matrix $\hat{\mathbf{B}}$ in selected conditions. These visualizations exemplify some of the ways that a replication can differentiate itself from others in the same data set. (1) and (2) two types of grasp opening, the former through extension of fingers corresponding to interphalangeal joint angle extension and the latter through metacarpophalangeal joint angle extension; (3) Markers of two fingers move significantly faster than the others in a non-synchronized grasping movement; (4) Complex movement corresponding to curling fingers around a cone.