Keeping an Eye on Hidden Markov Models in Gaze Data Classification

Malte Lueken[1], Simon Kucharsky[1], & Ingmar Visser[1]

[1] Department of Psychology, University of Amsterdam. Amsterdam, The Netherlands

Author Note

This is an author note.

Correspondence concerning this article should be addressed to Malte Lueken, Postal address. E-mail: malte_lueken@arcor.de

Abstract

Eye-tracking allows researchers to infer cognitive processes from eye movements that are classified into distinct events. Parsing the events is typically done by algorithms that transform, filter, classify, and merge raw data samples. Previous algorithms have successfully used hidden Markov models (HMMs) for classification but still inhere weaknesses. Therefore, I developed gazeHMM, an HMM algorithm that has no critical parameters to be set by users, does not require human coded data as input, and classifies fixations, saccades, PSOs, and smooth pursuits. The development was guided by the question of whether HMMs are useful at describing eye movements and whether they improve the event classification. I evaluated gazeHMM in a simulation study and on benchmark data. The simulation study showed that gazeHMM successfully recovered HMM parameters and hidden state sequences. Critical exceptions for good recovery were adding a smooth pursuit like state to gazeHMM and noisy data. For benchmark data, model comparisons with gazeHMM yielded preferred models with more states than expected. I assessed the classification performance of gazeHMM compared to other algorithms by the agreement to human event coding. Here, gazeHMM improved the event classification but did not outperform all other algorithms in most cases. Both the simulation study and benchmark application showed poor classification performance for smooth pursuits. Thus, I advice to classify smooth pursuits only for exploration and recommend gazeHMM with fixations, saccades, and PSOs for application. Future HMM algorithms could use covariates to better capture eye movement processes and explicitly model event durations to improve the classification of smooth pursuits.

Keywords: eye movements; eye-tracking; parameter recovery; dependent mixture models

Word count: 12962

Keeping an Eye on Hidden Markov Models in Gaze Data Classification

Introduction

Eye-tracking is typically used to study cognitive processes involving attention and information search based on recorded gaze position (Schulte-Mecklenbeck et al., 2017). Before these processes can be studied, the raw gaze data is classified into events that are distinct in their physiological patterns (e.g., duration), underlying neurological mechanisms, or cognitive functions (Leigh & Zee, 2015). Commonly distinguished events are fixations, saccades, smooth pursuits, and post-saccadic oscillations (PSOs). Classifying eye-tracking data reduces their complexity and is the first step towards cognitive interpretation (Salvucci & Goldberg, 2000). The classification is normally done by algorithms, which is considered faster, more objective, and reproducible compared to human coding (Andersson, Larsson, Holmqvist, Stridh, & Nyström, 2017). Hein and Zangemeister (2017) give a comprehensive overview of different classification algorithms (for a structured review on classifying saccades, see also Stuart et al., 2019).

Probabilistic models are frequently implemented to classify events. Instead of discrete classification, these models assign probabilities for belonging to an event to each sample. Probabilistic models have the advantage that they learn parameters from the data, can adapt to the task- and individual-specific gaze signals, and can be easily applied online (Kasneci, Kasneci, Kübler, & Rosenstiel, 2014, p. 324).

One class of probabilistic models that are used in eye movement classification are hidden Markov models (HMMs). They estimate a sequence of hidden states (i.e., a discrete variable that cannot be directly observed) that evolves parallel to the gaze signal. Each gaze sample depends on its corresponding state. Each state depends on the previous but not on earlier states of the sequence (Zucchini, MacDonald, & Langrock, 2016).

In the context of eye movements, HMMs are suitable probabilistic models because the hidden states can be interpreted as eye movement events and gaze data are dependent time series (i.e., one gaze sample depends on the previous). On this basis, several classification algorithms using HMMs have been developed: One instance is described in Salvucci and Goldberg (2000) and combines the HMM with a threshold approach (named “Identification by HMM” [I-HMM]). Samples are first labeled as fixations or saccades, depending on whether their velocity exceeds a threshold, and then reclassified by the HMM. Pekkanen and Lappi (2017) developed an algorithm that filters the position of gaze samples through naive segmented linear regression (NSLR). The algorithm uses an HMM to parse the resulting segments into fixations, saccades, smooth pursuits, and PSOs based on their velocity and relative angle (named NSLR-HMM). Another version by Mihali, Opheusden, and Ma (2017) uses a Bayesian HMM to separate microsaccades (short saccades during fixations) from motor noise based on sample velocity (named “Bayesian Microsaccade Detection” [BMD]). Moreover, Houpt, Frame, and Blaha (2018) developed a hierarchical approach that describes sample velocity and acceleration through an autoregression (AR) model, computes the regression weights through an HMM, and estimates the number of events with a beta-process (BP) from the data (named BP-AR-HMM).

Several studies have tested the performance of HMM algorithms against other classification methods: I-HMM has been deemed as robust against noise, behaviorally accurate, and showing a high sample-to-sample agreement to human coders (Andersson et al., 2017; Komogortsev, Gobert, Jayarathna, Koh, & Gowda, 2010; Salvucci & Goldberg, 2000). However, the agreement was lower when compared to an algorithm using a Bayesian mixture model (Kasneci et al., 2014; Tafaj, Kasneci, Rosenstiel, & Bogdan, 2012). NSLR-HMM showed even higher agreement to human coding than I-HMM but was outperformed by a recently developed algorithm using convolutional neural networks (Bellet, Bellet, Nienborg, Hafed, & Berens, 2019; Pekkanen & Lappi, 2017). In sum, HMMs seem to be a promising method for classifying eye movements. Still, the existing HMM algorithms each have their weaknesses that could be improved upon (and that could explain their inferior performance in comparison to other methods).

First, I-HMM relies on setting an appropriate threshold, which can distort the results (Blignaut, 2009; Komogortsev et al., 2010; Shic, Scassellati, & Chawarska, 2008). Second, the current implementation of NSLR-HMM requires human-coded data, which limits its applicability. It also inherits fixed parameters that prevent the algorithm to adapt to the individual- or task-specific signals (contrary to the authors, we argue that this is a disadvantage). Third, BMD limits the classification to microsaccades which are irrelevant in many applications and sometimes even considered as noise (Duchowski, 2017). The opposite problem was observed for BP-AR-HMM: It tends to estimate an unreasonable number of events from the data of which many are considered as noise events. Therefore, the authors suggest using it as an exploratory tool followed by further event classification (Houpt et al., 2018).

HMMs have another property that has been rarely used in the context of eye movement classification: They can serve as a generative model for gaze data. Only Mihali et al. (2017) used this property by building an explicit generative model for drift and microsaccades.

For this article, we developed a novel algorithm for classifying gaze data named gazeHMM that relies on an HMM as a generative model and addresses the weaknesses of previous algorithms. The development was guided by four goals: (a) it should not require parameter settings through the user (e.g., thresholds) or (b) human-labeled data as input; (c) it should cover the most relevant eye movement events, namely fixations, saccades, smooth pursuit, and PSOs; (d) it should confirm rather than explore the presence of events in the data.

The Algorithm

Preprocessing

Algorithms require variables that describe gaze data (hereafter called *eye movement metrics*) to classify them into events. Many eye movement metrics have been proposed and used in previous algorithms (for examples, see Zemblys, Niehorster, Komogortsev, & Holmqvist, 2018, and @Andersson2017). However, many of them rely on thresholds or window ranges that have to be set by the user (e.g., the distance between the mean position in a 100 ms window before and after each sample, see Olsson, 2007). This can be problematic because such parameters are often set without theoretical justification and they differ substantially between metrics (Andersson et al., 2017). In gazeHMM, we used velocity, acceleration, and sample-to-sample angle (synonymous to relative or change in angle; Larsson, Nystrom, & Stridh, 2013) because they belong to the most basic metrics which do not require parameter settings.

Theoretically, these three metrics should separate eye movement events clearly. Fixations typically inherit samples with low velocity and acceleration (Larsson et al., 2013). Due to tremor, the angle between samples should not follow any direction but a random walk (Duchowski, 2017). In contrast, saccade samples usually have a high velocity and acceleration and follow the same direction. PSO samples tend to have moderate velocity and high acceleration since they occur between saccades and low velocity events (Larsson et al., 2013). They can be specifically distinguished by their change in direction clustered around 180 degrees (Pekkanen & Lappi, 2017). Importantly, the oscillations depend on the resolution of the gaze recording: Eye-trackers with higher sampling frequency yield more changes in direction and more samples in between those changes. Those samples in between typically follow the same direction. Thus, with high sampling frequencies, PSO samples might also cluster around a sample-to-sample angle of zero with outliers around 180 degrees. Lastly, smooth pursuit samples have a moderate velocity but low acceleration (due to the smoothness) and like saccades they follow the same direction (Larsson et al., 2013; Leigh & Zee, 2015). Other algorithms focus exclusively on classifying microsaccades (e.g., Mihali et al., 2017), but as stated earlier, they were not in the scope of gazeHMM.

The velocity and acceleration signals are computed from the raw gaze position by using a Savitzky-Golay filter (Savitzky & Golay, 1964; similar to Nyström & Holmqvist, 2010). In contrast, the absolute angle between two samples is calculated as the finite backward difference and the sample-to-sample angle as the finite forward difference.

Most of the missing data in eye movement classification are due to blinks. Therefore, the user can indicate in gazeHMM which samples should be labeled as blinks (other missing samples are treated as noise). Often, eye-trackers record a few samples with unreasonably high velocity and acceleration before losing the pupil signal when a blink occurs. Since these samples could distort the classification of saccades in the HMM, gazeHMM removes them heuristically. Before classifying the samples, it sets all samples within 50 ms before and after blink samples as missing. The window of 50 ms was rather motivated empirically than theoretically and can bet set to any value the user considers appropriate.

The Generative Model

We denote the three eye movement metrics by X, Y, and Z. Each metric was generated by a hidden state variable S. Given S, the HMM treats X, Y, and Z as conditionally independent. Conditional independence might not accurately resemble the relationship between velocity and acceleration (which are naturally correlated). This step was merely taken to keep the HMM simple and identifiable. In gazeHMM, S can take one of two, three, or four hidden states. The first state always represents fixations, the second saccades, the third PSOs, and the fourth smooth pursuits. Thus, users can choose whether they would like to classify only fixations and saccades, or additionally PSOs and/or smooth pursuits.

HMMs can be described by three submodels: An initial state model, a transition model, and a response model. The initial state model contains probabilites for the first state of the hidden sequence $\rho_i = P(S_1 = i)$, with i denoting the hidden states. In gazeHMM, the initial states are modeled by a multinomial distribution. The evolution of the sequence is in turn described by the trantision model, which comprises the probabilities for transitioning between different states in the HMM. Typically, probabilities to transition from state i to j $a_{ij} = P(S_{t+1} = j|S_t = i)$ are expressed in matrix form (Visser, 2011):

$$\mathbf{A} = \begin{bmatrix} a_{11} & ... & a_{1j} \\ \vdots & \ddots & \vdots \\ a_{i1} & ... & a_{ij} \end{bmatrix}.$$

Again, the transition probabilities for each state are modeled by multinomial distributions.

The response model encompasses distributions describing the response variables for every state in the model. Previous algorithms have used Gaussian distributions to describe velocity and acceleration signals (sometimes after log-transforming them). However, several reasons speak against choosing the Gaussian: First, both signals are usually positive (depending on the computation). Second, the distributions of both signals appear to be positively skewed conditionally on the states and third, to have variances increasing with their mean. Thus, instead of using the Gaussian, it could be more appropriate to describe velocity and acceleration with a distribution that follows these three properties. In gazeHMM, we use gamma distributions with a shape and scale parametrization for this purpose:

$$X \sim Gamma(\alpha, \beta),$$

$$Y \sim Gamma(\alpha, \beta).$$

It has to be noted that the gamma was chosen out of convenience and the best fitting distribution might be different between eye-trackers, subjects, and tasks.

To model the sample-to-sample angle, we pursued a novel approach in gazeHMM: A mixture of von Mises distributions (with a mean and concentration parameter) and a uniform distribution:

$$Z \sim vonMises(\mu, \kappa) + U(0, 2\pi).$$

Both the distributions and the metric operate on the full unit circle (i.e., between 0 and 2π), which should lead to rather symmetric distributions. Because the fixations change their direction similar to a random walk, their sample-to-sample angle can be modeled by a uniform distribution. Thus, the uniform distribution should distinguish fixations from the other events.

All three submodels taken together, the joint likelihood of the observed data and hidden states can be expressed as:

$$L(X, Y, Z, S|\lambda) = \rho_{S_1} f_{S_1}(X_1) f_{S_1}(Y_1) f_{S_1}(Z_1) \prod_{t=1}^{T-1} a_{S_t S_{t+1}} f_{S_{t+1}}(X_{t+1}) f_{S_{t+1}}(Y_{t+1}) f_{S_{t+1}}(Z_{t+1}),$$

with λ denoting the vector containing the initial state and transtition probabilities as well as the response parameters. By summing over all possible state sequences, the likelihood of the data given the HMM parameters becomes (Visser, 2011):

$$L(X, Y, Z|\lambda) = \sum_{\text{all S}} \rho_{S_1} f_{S_1}(X_1) f_{S_1}(Y_1) f_{S_1}(Z_1) \prod_{t=1}^{T-1} a_{S_t S_{t+1}} f_{S_{t+1}}(X_{t+1}) f_{S_{t+1}}(Y_{t+1}) f_{S_{t+1}}(Z_{t+1}).$$

The parameters of the HMM are estimated through maximum likelihood using an expectation-maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977; McLachlan & Krishnan, 1997). The EM algorithm is generally suitable to estimate likelihoods with missing variables. For HMMs, it imputes missing with expected values and iteratively maximizes the joint likelihood of parameters conditional on the observed data and the expected hidden states (eye movement events; Visser & Speekenbrink, 2019). When evaluating the likelihood for missing data, gazeHMM integrates over all possible values, which results in a probability density of one. The sequence of hidden states is estimated through the Viterbi algorithm (Forney Jr, 1973; Viterbi, 1967) by maximizing the posterior state probability. Parameters of the response distributions (except for the uniform distribution) are optimized on the log-scale (except for the mean parameter of the von

Mises distribution) using a spectral projected gradient method (Birgin, Martinez, & Raydan, 2000) and Barzilai-Borwein step lengths (Barzilai & Borwein, 1988).

Postprocessing

After classifying gaze samples into states, gazeHMM applies a postprocessing routine to the estimated state sequence. We implemented this routine because constraining the transition probabilities for PSOs to turn into non-saccade events to zero often caused PSOs not to appear in the state sequence at all. Moreover, gazeHMM does not explicitly control the duration of events in the HMM which occasionally led to unreasonably short events. Thus, the postprocessing routine heuristically compensates for such violations. This routine relabels one-sample fixations and smooth pursuits, saccades with a duration below a minimum threshold, and PSOs that follow non-saccade events. Samples are relabeled as the state of the previous event. Finally, samples initially indicated as missing are labeled as noise (including blinks) and event descriptives are computed (e.g., fixation duration).

The algorithm is implemented in R (version: 3.6.3; R Core Team, 2020) and uses the packages signal (Ligges, Short, & Kienzle, 2015) to compute velocity and acceleration signals, depmixS4 (Visser & Speekenbrink, 2010) for the HMM, and BB (Varadhan & Gilbert, 2009) for Barzilai-Borwein spectral projected gradient optimization. The algorithm is available on GitHub (www.github.com/maltelueken/gazeHMM).

Simulation Study

To assess how well the HMM recovers parameters and state sequences, we conducted a simulation study. The design and analysis of the study were preregistered on the Open Science Framework (https://doi.org/10.17605/OSF.IO/VDJGP). This section is majorly copied from the preregistration (with adapted tenses). The study was divided in four parts. Here, we only report the first two parts, the other two parts, which address starting values and missing data, can be found in the supplementary material (URL).

The HMM repeatedly generated data with a set of parameters (true parameter values). The same model was applied to estimate the parameters from the generated data (estimated parameter values). We compared the true with the estimated parameter values to assess whether a parameter was recovered by the model. Additionally, we contrasted the true states of the HMM with the estimated states to judge how accurately the model recovered the states that generated the data.

Starting Values

The HMM always started with a uniform distribution to estimate the initial state and state transition probabilities. Random starting values for the estimation of shape, scale, and concentration parameters were generated by gamma distributions with a shape parameter of $\alpha_{start} = 3$ and $\beta_{start} = \psi_{true}/2$, with ψ_{true} being the true value of the parameter to be estimated. This setup ensured that the starting values were positive, their distributions were moderately skewed, and the modes of their distributions equaled the true parameter values. Mean parameters of the von Mises distribution always started at their true values.

Design

In the first part, we varied the parameters of the HMM. For models with $k \in \{2, 3, 4\}$ states, $q \in \{10, 15, 20\}$ parameters were manipulated, respectively. For each parameter, the HMM generated 100 data sets with $N = 2500$ samples and the parameter varied in a specified interval in equidistant steps. This resulted in $100 \times (10 + 15 + 20) = 4500$ recoveries. Only one parameter alternated at once, the other parameters were set to their default values. All parameters of the HMM were estimated freely (i.e., there were no fixed parameters in the model). We did not manipulate the initial state probabilities because these are usually irrelevant in the context of eye movement classification. For the transition probabilities, we only simultaneously changed the probabilities for staying in the same state (diagonals of the transition matrix) to reduce the complexity of the simulation. The left over probability mass was split evenly between the probabilities for switching to a different state (per row of the transition matrix). Moreover, we did not modify the mean parameters of the von Mises distributions: As location parameters, they do not alter the shape of the distribution and they are necessary features for the HMM to distinguish between different states.

We defined approximate ranges for each response variable (see supplementary material) and chose true parameter intervals and default values so that they produced samples that roughly corresponded to these ranges. Tables 1 and 2 show the intervals and default values for each parameter in the simulation. Parameters were scaled down by factor 10 (compared to the reported ranges) to improve fitting of the gamma distributions. We set the intervals for shape parameters of the gamma distributions for all events to [1,5] to examine how skewness influenced the recovery (shape values above five approach a symmetric distribution). The scale parameters were set so that the respective distribution approximately matched the assumed ranges. Since the concentration parameters of the von Mises distribution are the inverse of standard deviations, they were varied on the inverse scale. An example of simulated data from the HMM with default parameters is visualized in Figure **??**.

In the second part, we manipulated the sample size of the generated data and the amount of noise added to it. The model parameters were set to their default values. For models with $k \in \{2, 3, 4\}$ states and sample sizes of $N \in \{500, 2500, 10000\}$, we generated 100 data sets ($100 \times 3 \times 3 = 900$ recoveries). These samples sizes roughly corresponded to small, medium, and large eye-tracking data sets for a single participant and trial. To simulate noise, we replaced velocity and acceleration values y with draws from a gamma distribution with $\alpha_{noise} = 3$ and $\beta_{noise} = (y/2)\tau_{noise}$ with $\tau_{noise} \in [1, 5]$ varying between data sets. This procedure ensured that velocity and acceleration values remained positive and were taken from moderately skewed distributions with modes equal to the original values.

To angle, we added white noise from a von Mises distribution with $\mu_{noise} = 0$ and $\kappa_{noise} \in 1/[0.1, 10]$ varying between data sets. τ_{noise} and κ_{noise} were increased simultaneously in equidistant steps in their intervals.

Data Analysis

For each parameter, we calculated the root median square proportion deviation (RMdSPD; analogous to root median square percentage errors, see Hyndman & Koehler, 2006) between the true and estimated parameter values:

$$\text{RMdSPD} = \sqrt{\text{Med}\left((\frac{\psi_{true} - \psi_{est}}{\psi_{true}})^2 \right)}.$$

Even though it was not explicitly mentioned in the preregistration, this measure is only appropriate when $\psi_{true} \neq 0$. This was not the case for some mean parameters of the von Mises distributions. In those cases, we used $\psi_{true} = 2\pi$ instead. We treated RMdSPD < 0.1 as good, $0.1 \leq$ RMdSPD < 0.5 as moderate, and RMdSPD ≥ 0.5 as bad recovery of a parameter. By taking the median, we reduced the influence of potential outliers in the estimation and using proportions enabled us to compare RMdSPD values across parameters and data sets.

Additionally, we applied a bivariate linear regression with the estimated parameter values as the dependent and the true parameter values as the independent variable to each parameter that has been varied on an interval in part one. Regression slopes closer to one indicated that the model better captured parameter change. Regression intercepts different from zero reflected a bias in parameter estimation.

To assess state recovery, we computed Cohen's kappa (for all events taken together, not for each event separately) as a measure of agreement between true and estimated states for each generated data set. Cohen's kappa estimates the agreement between two classifiers accounting for the agreement due to chance. Higher kappa values were interpreted as better model accuracy. We adopted the ranges proposed by Landis and Koch (1977) to

Table 1

Intervals and Default Parameter Values for the Transition Model in the Simulation

	Initial state prob.	Trans. prob. same state	Trans. prob. other state
Interval	-	[.01,.99]	1-a(i=j)/(k-1)
Default	1/k	0.9	0.1/(k-1)

Note. The transition probability for staying in the same state is denoted by $a_{i=j}$ and the probability for switching to a different state by $a_{i \neq j}$. The number of states in the model is denoted by k.

interpret kappa values. Models that could not be fitted were excluded from the recovery.

Results

Parameter Variation. In the first part of the simulation, I examined how varying the parameters in the HMM affects the deviation of estimated parameters and accuracy of estimated state sequences. Figure 1 displays the RMdSPD between true and estimated parameters depending on which parameter has been manipulated in the HMM. The RMdSPD was below 0.1 for all estimated and manipulated parameters, indicating good recovery.[1] The regressions between manipulated true and estimated parameters are shown in Figures 2 and 3. With one outlier at parameter $\alpha_{acc;1}$, the estimated parameters matched the true parameters very closely. Thus, parameter change was captured well and the estimation almost unbiased. Considering accuracy, Figure 4 displays Cohen's kappa between true and estimated hidden state sequences. With two exceptions, kappa values were almost one, suggesting nearly perfect agreement.

[1] Note that the initial state probability ρ_i has RMdSPD = 1. Since the HMM only simulated one state sequence, this parameter is always either zero or one (leading to RMdSPD = 1). Therefore, I decided not to include it in the analysis.

Table 2

Intervals and Default Parameter Values for the Response Model in the Simulation

	Velocity		Acceleration		Rel. angle	
	Shape	Scale	Shape	Scale	Mean	Concentration
State 1						
Interval	[1,5]	[0.1,0.6]	[1,5]	[0.05,0.25]	-	-
Default	3	0.35	3	0.25	-	-
State 2						
Interval	[1,5]	[5,15]	[1,5]	[1,5]	-	1/[0.1,10]
Default	3	10	3	3	0	1
State 3						
Interval	[1,5]	[0.5,1.5]	[1,5]	[1,5]	-	1/[0.1,10]
Default	3	1	3	3	pi	1
State 4						
Interval	[1,5]	[0.5,1.5]	[1,5]	[0.05,0.25]	-	1/[0.1,10]
Default	3	1	3	0.15	0	1

Note. Shape parameters are denoted by α, scale parameters by β, mean parameters by μ, and concentration parameters by κ. The default values for the uniform distribution in state one were min $= 0$ and max $= 2\pi$.

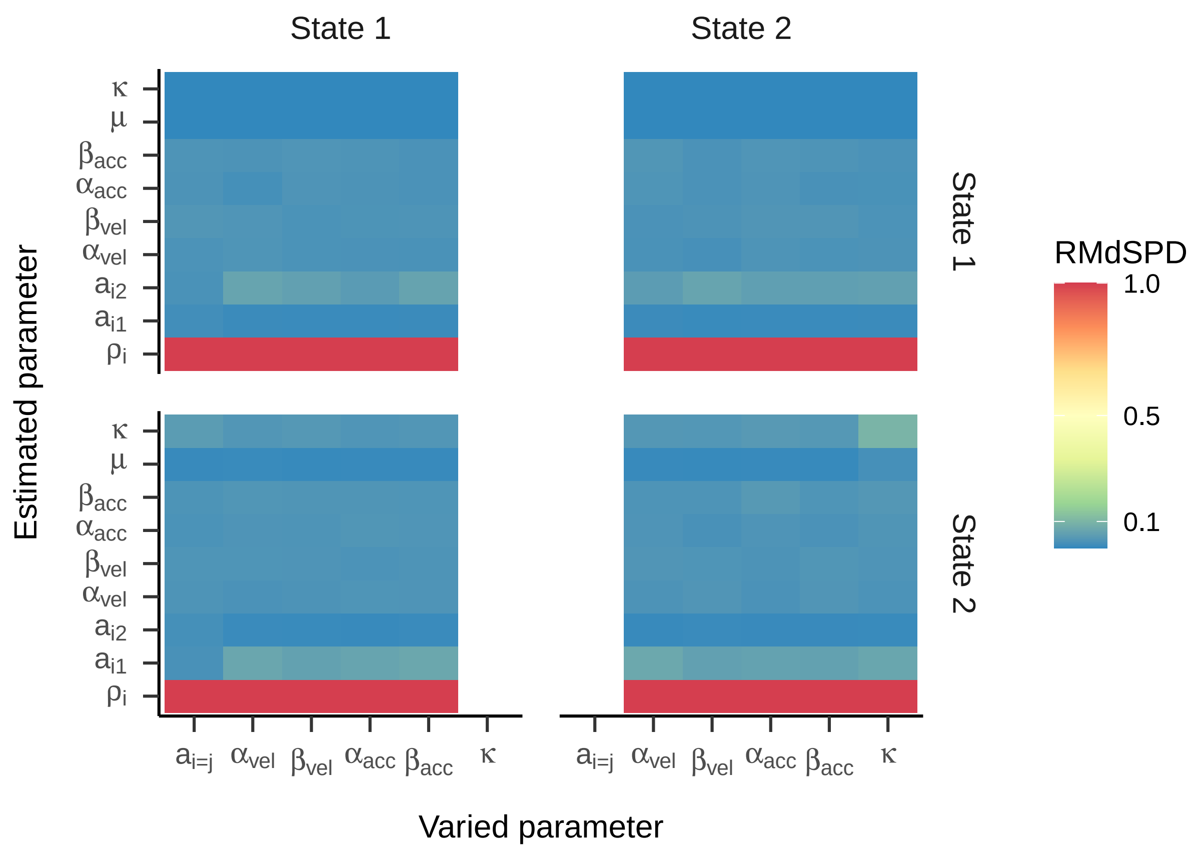

Figure 1. RMdSPD between true and estimated parameters of the two-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which state the parameters have been varied and right facet labels denote to which state estimated parameters belong.

For the HMM with three states, the RMdSPD is shown in Figure 5. When response parameters (other than $a_{i=j}$) were manipulated, the RMdSPDs for a_{12} and a_{31} were consistently between 0.1 and 0.5. Varying κ in states two and three led to RMdSPDs between 0.1 and 0.5 in the respective states, which we interpreted as moderate recovery. Otherwise, RMdSPDs were consistently lower than 0.1, indicating good recovery. Inspecting the regressions between manipulated true and estimated parameters (see Figures 6 and 7) revealed strong and unbiased linear relationships (intercepts close to zero

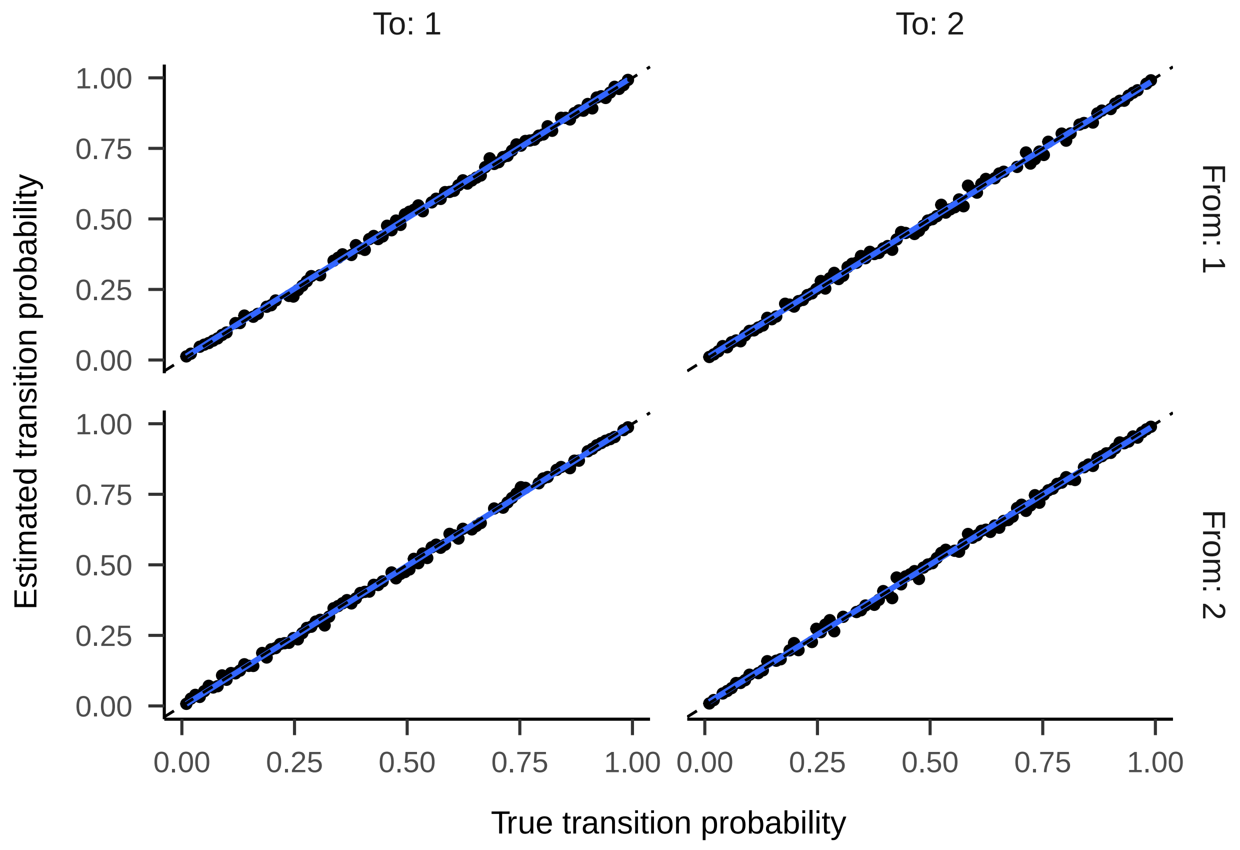

Figure 2. Regression lines between true and estimated transition probabilities for the two-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving. Dashed lines refer to perfect recovery.

and slopes close to one). In contrast to the two-state HMM, larger deviations and more outliers were observed. Cohen's kappa values are presented in Figure 8. For most estimated models, the kappa values between true and estimated state sequences were above 0.95, meaning almost perfect agreement. However, for some models, we observed kappas clustered around zero or -0.33, which suggests that state labels were switched.

The RMdSPDs for the four-state HMM is shown in Figure 9. For estimated transition probabilities and α_{vel} and β_{vel} parameters in states one and four, RMdSPDs were between 0.1 and 0.5, suggesting moderate recovery. Estimated kappa parameters in state four were also often moderately recovered when parameters in states two, three, and

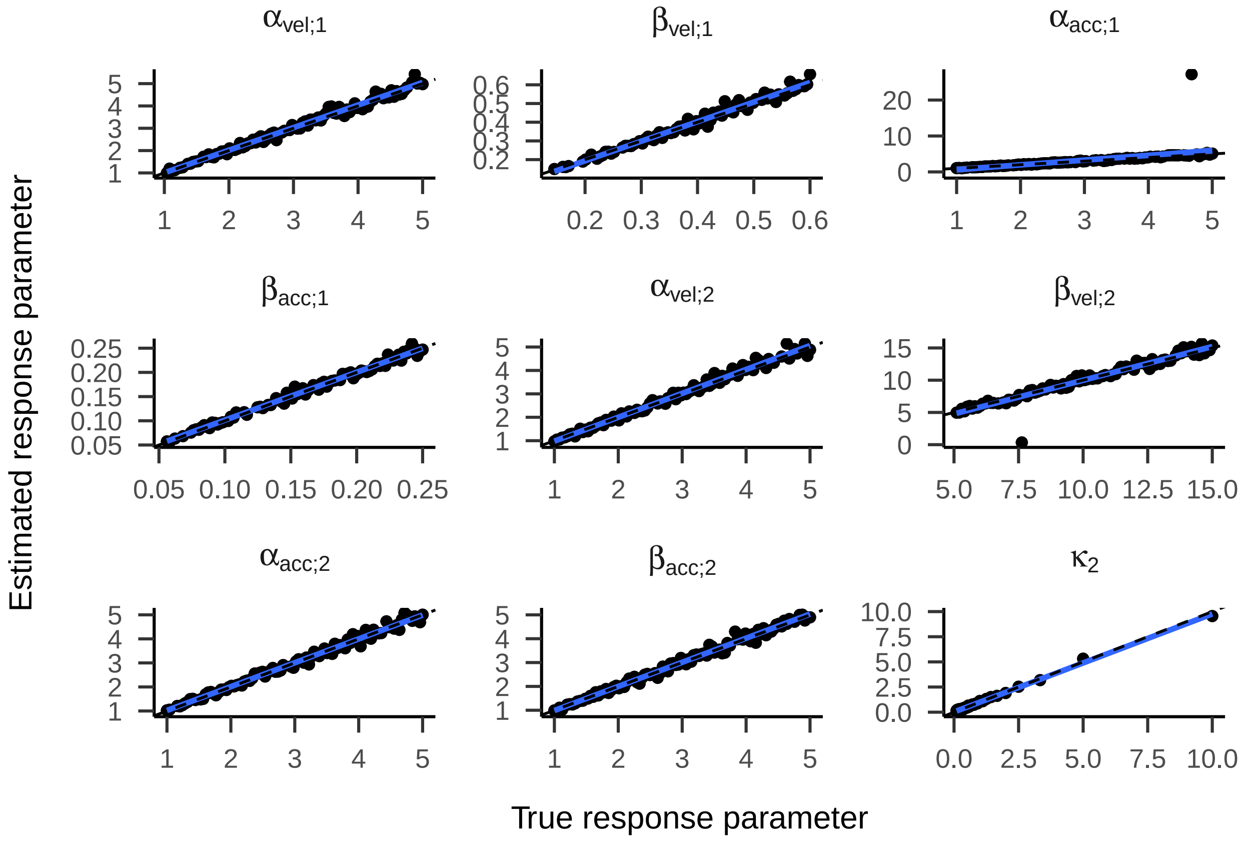

Figure 3. Regression lines between true and estimated response parameters of the two-state HMM in part one. Top facet labels indicate response parameters. Dashed lines refer to perfect recovery.

four were varied. Otherwise, RMdSPDs were below 0.1, indicating good recovery. Looking at the regressions between true and estimated parameters, Figures 10 and 11 illustrate strong and unbiased relationships. However, there were larger deviations and more outliers than in the previous models, especially for states one and four. Cohen's kappa ranged majorly between 0.6 and 0.9, meaning moderate to almost perfect agreement between true and estimated state sequences (see Figure 12). Here, some kappa values clustered around 0.25 and zero, which, again, can be interpreted as the result of label switching.

Overall, simulations in part one demonstrated that the HMM recovered parameters very well when they were manipulated. Deviations from true parameters were mostly

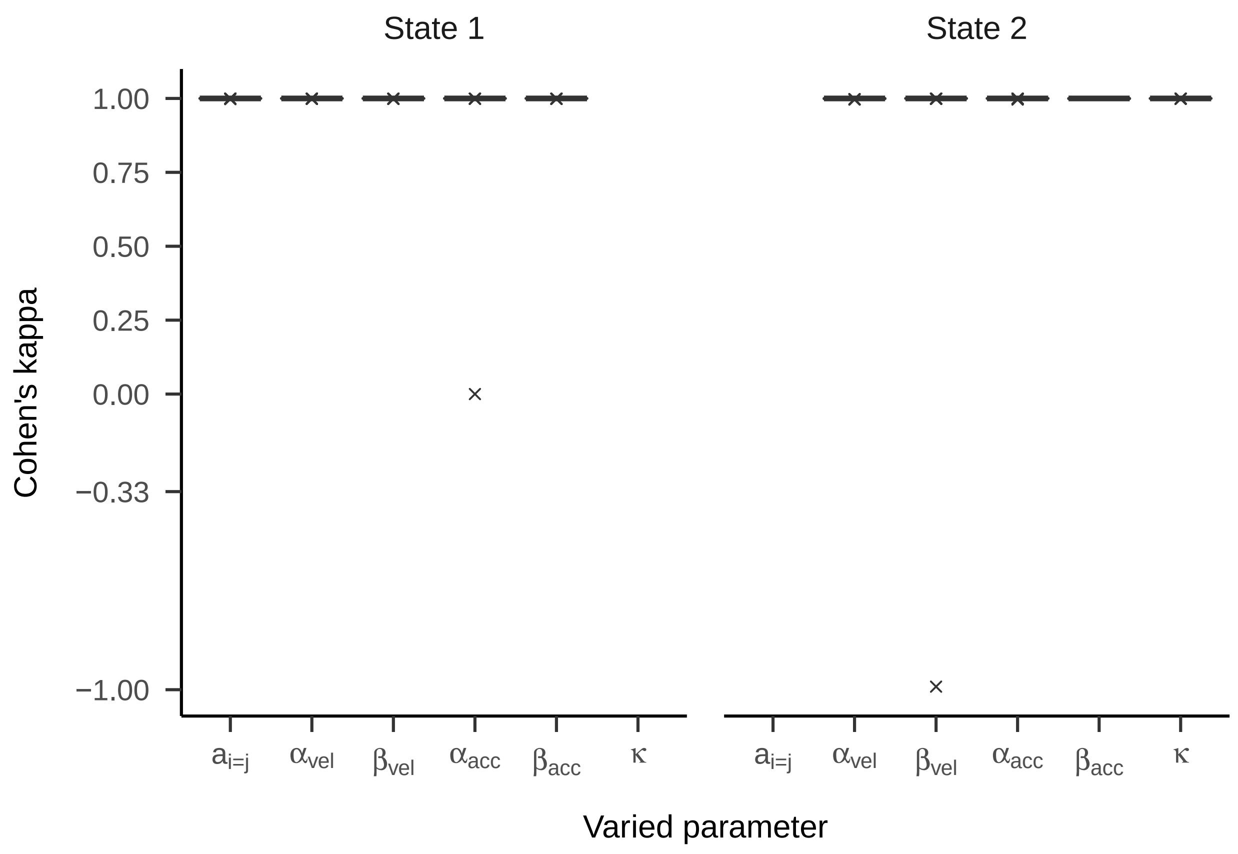

Figure 4. Boxplots displaying Cohen's kappa depending on which parameter of the two-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians. Crosses represent outliers (distance to first/third quartile higher than 1.5 times the inter-quartile range [IQR]).

small. In the four-state model, estimated transition probabilities for state one and four deviated moderately. Moreover, the HMM estimated state sequences very accurately with decreasing accuracy for the four-state model.

Sample Size and Noise Variation. In the second part, we varied the sample size of the HMM and added noise to the generated data. For the two-state HMM, the RMdSPDs were above 0.5 for β_{vel} and β_{acc} in both states (see Figure 13), suggesting bad recovery. The other estimated parameters showed RMdSPDs close to or below 0.1, which means they were recovered well. Increasing the sample size seemed to improve RMdSPDs

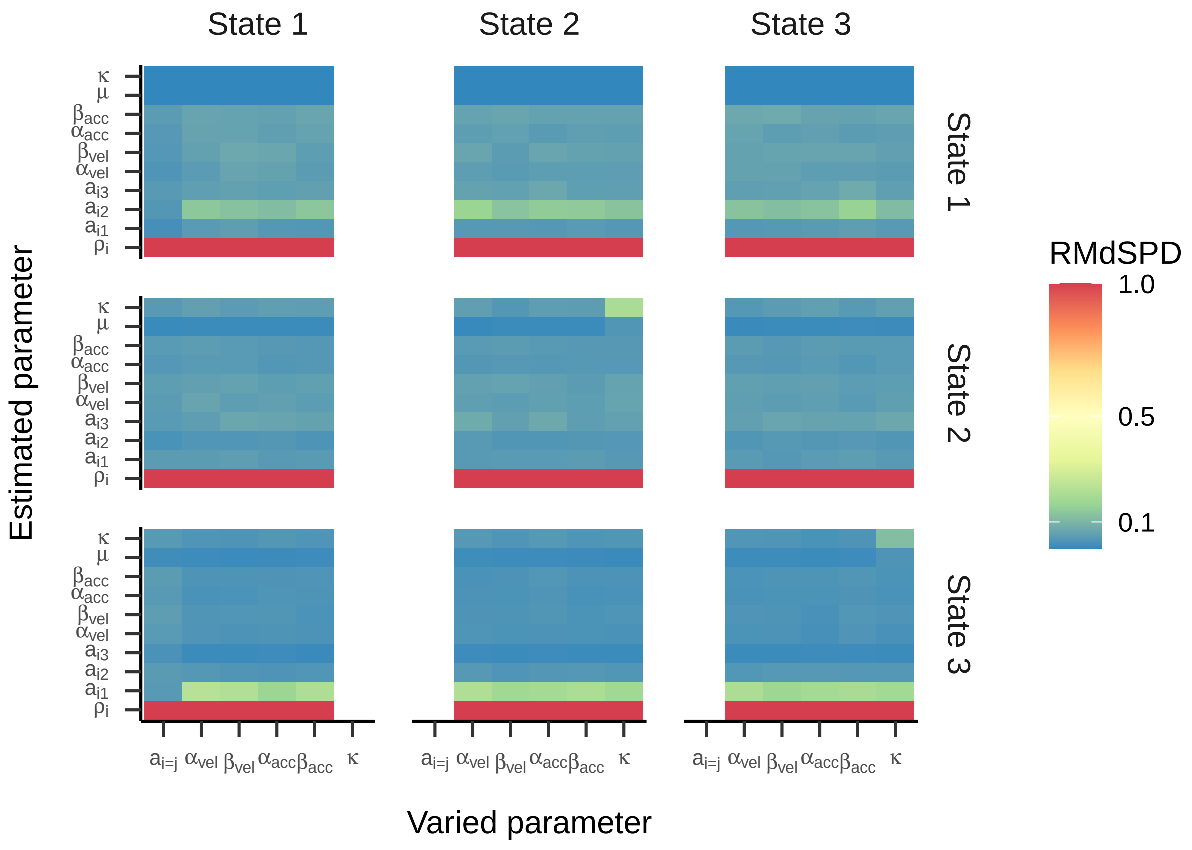

Figure 5. RMdSPD between true and estimated parameters of the three-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which state the parameters have been varied and right facet labels denote to which state estimated parameters belong.

for most parameters slightly. For β_{vel} and β_{acc} in both states, models with 2500 samples had the lowest RMdSPDs. Accuracy measured by Cohen's kappa was almost perfect with kappa values very close to one (see Figure 14, left plot).

Regarding three states, the RMdSPDs for the β_{vel} and β_{acc} were above 0.5 in all three states (see Figure 15), indicating bad recovery. Again, the other estimated parameters were below or close to 0.1, only a_{12} and a_{31} with 500 samples were closer to 0.5. For most parameters across all three states, models with higher sample sizes had lower RMdSPDs.

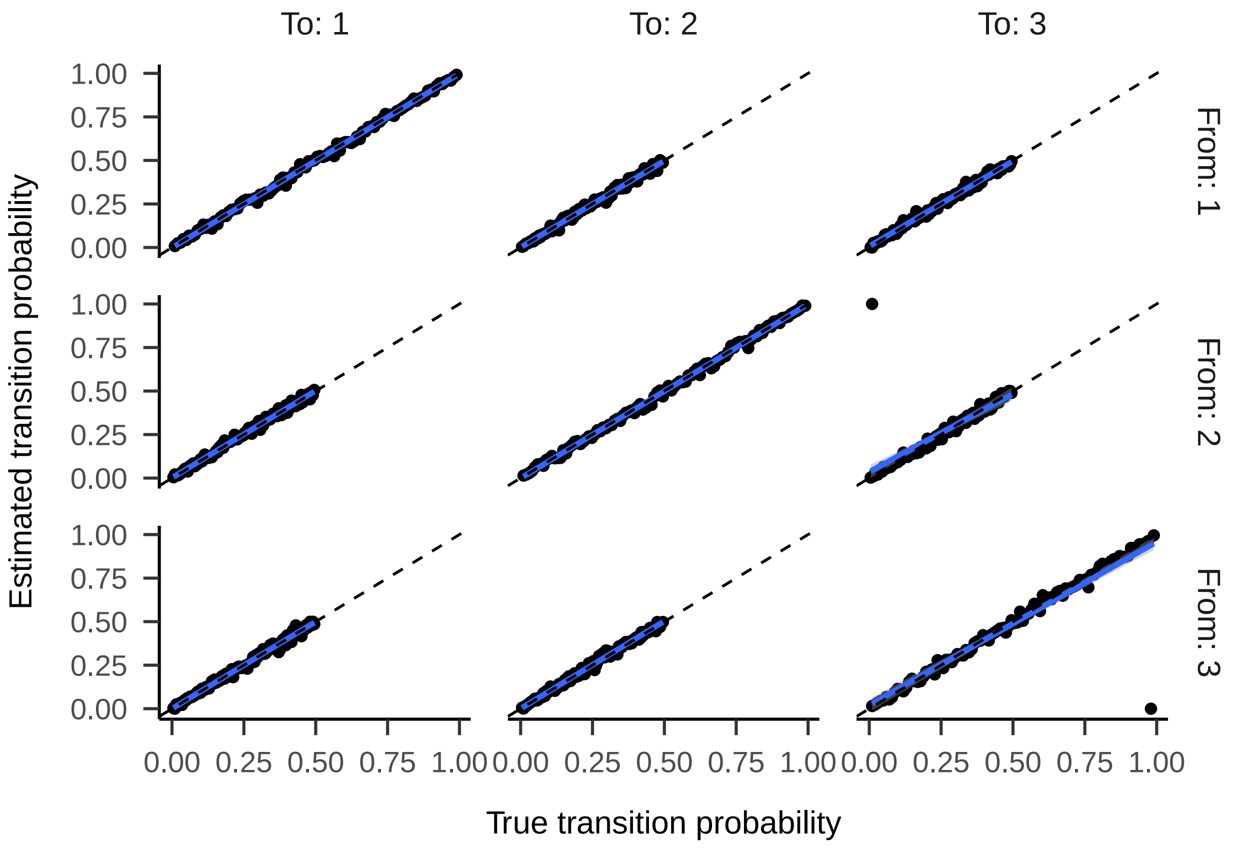

Figure 6. Regression lines between true and estimated transition probabilities for the three-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving. Dashed lines refer to perfect recovery.

The state recovery of the estimated models was almost perfect with most kappa values above 0.95 (see Figure 14, middle plot). Several outliers clustered around kappas of zero and -0.33, signaling label switching.

RMdSPDs regarding the four-state HMM are displayed in Figure 16. For states one and four, values for most parameters (including all transition probabilities) were above 0.5, suggesting bad recovery. Similarly, β_{vel} and β_{acc} in states two and three showed bad recovery. For states two and three, higher samples sizes showed slightly lower RMdSPDs. As in the previous part, most Cohen's kappa values ranged between 0.6 and 0.9, meaning substantial to almost perfect agreement between true and estimated states (Figure 14,

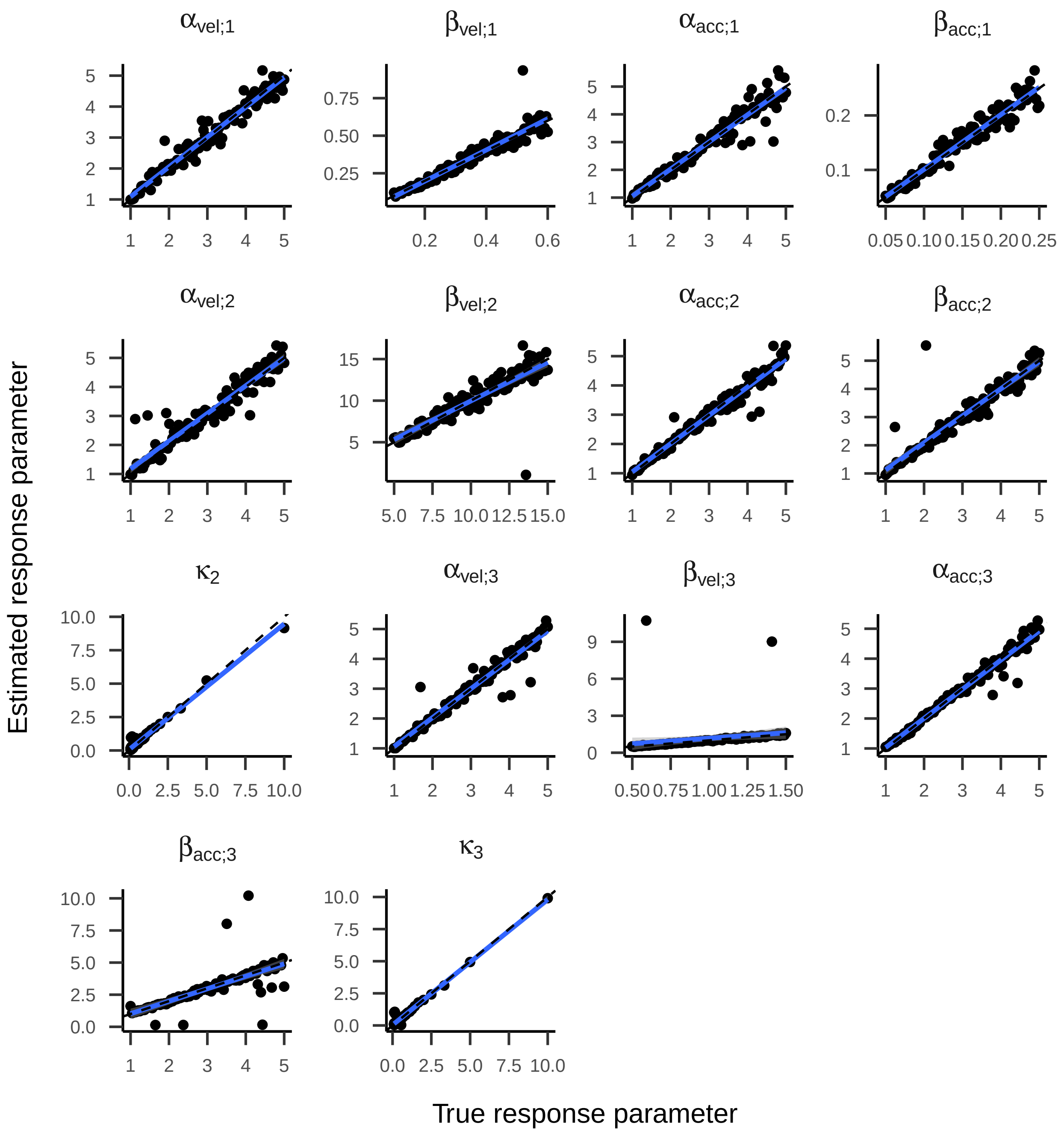

Figure 7. Regression lines between true and estimated response parameters of the three-state HMM in part one. Top facet labels indicate response parameters. Dashed lines refer to perfect recovery.

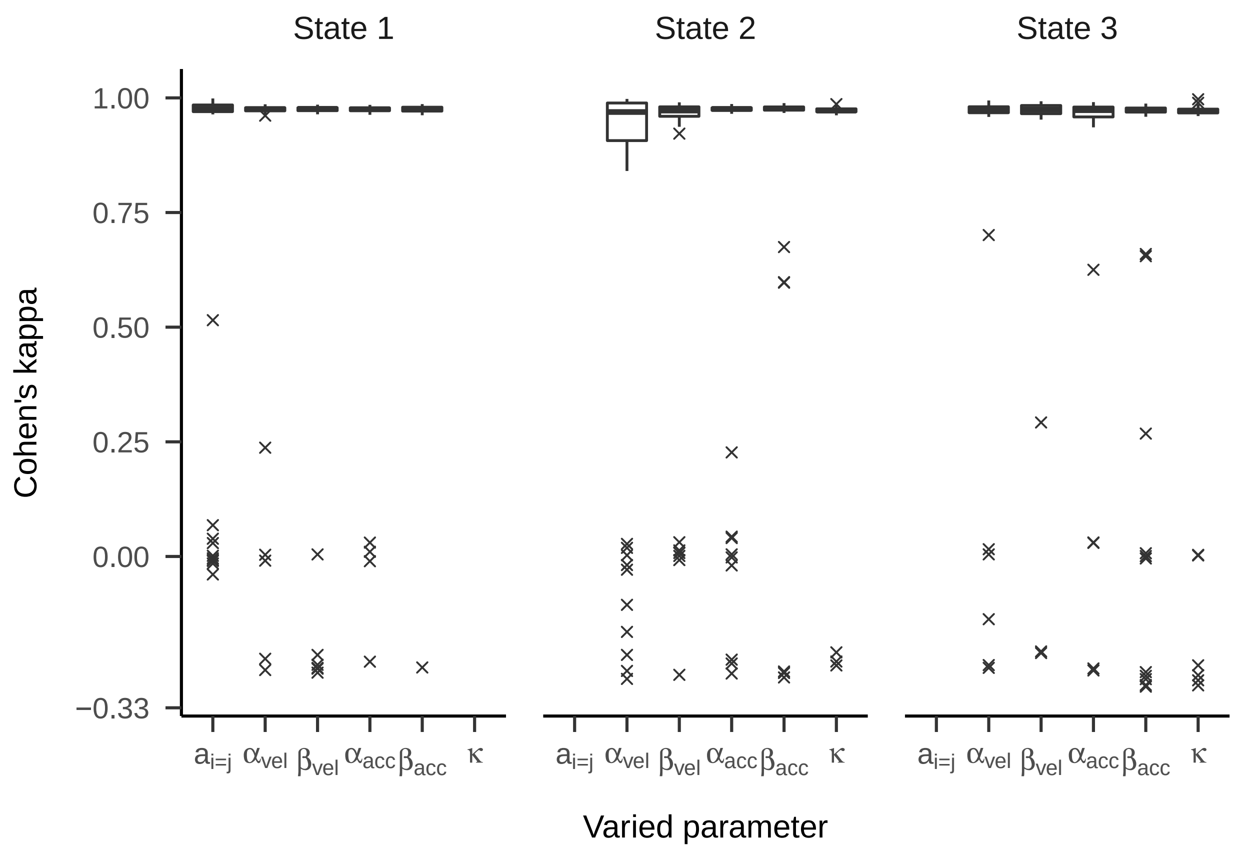

Figure 8. Boxplots displaying Cohen's kappa depending on which parameter of the three-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians and hinges the first and third quartile. Whiskers range from hinges to lowest/highest value within 1.5 times the IQR. Crosses represent outliers.

right plot). Multiple kappa values clustered around 0.25 or zero, which can be explained by label switching.

In general, the HMM recovered parameters well despite noise being added to the data. However, in the four-state model, the parameter recovery for states one and four substantially decreased. In the three- and four-state models, scale parameters of gamma distributions were badly recovered. Increasing the sample size in the HMM slightly improved the recovery of most parameters. The state recovery of the model was slightly

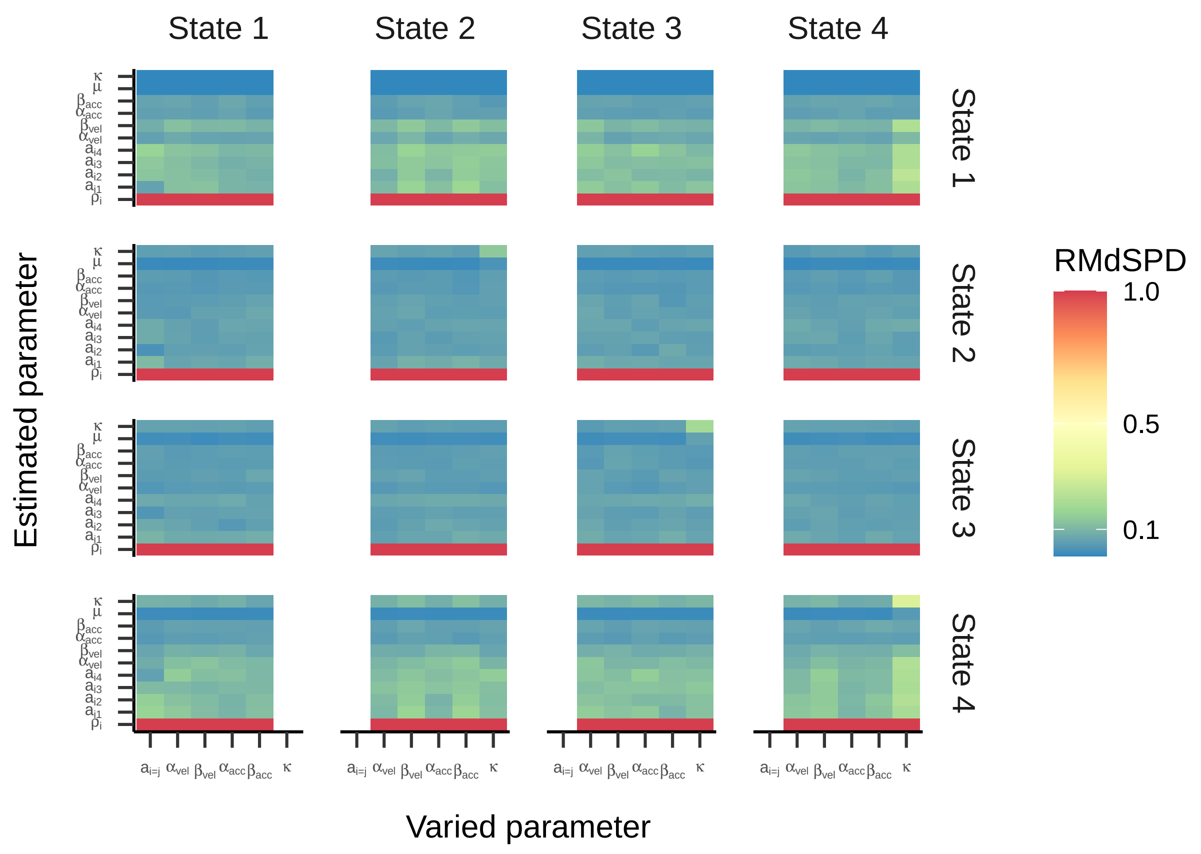

Figure 9. RMdSPD between true and estimated parameters of the four-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which state the parameters have been varied and right facet labels denote to which state estimated parameters belong.

lowered when more states were included, but it was neither affected by the noise variability τ_{noise} nor the sample size.

Validation Study

To validate gazeHMM, we applied the algortihm on two benchmark data sets. As starting values, we used $\rho = 1/k$ for the initial state model as well as $a_{i=j} = 0.9$ and $a_{i \neq j} = 0.1/k$ for the transition model. The values for the response model are displayed in

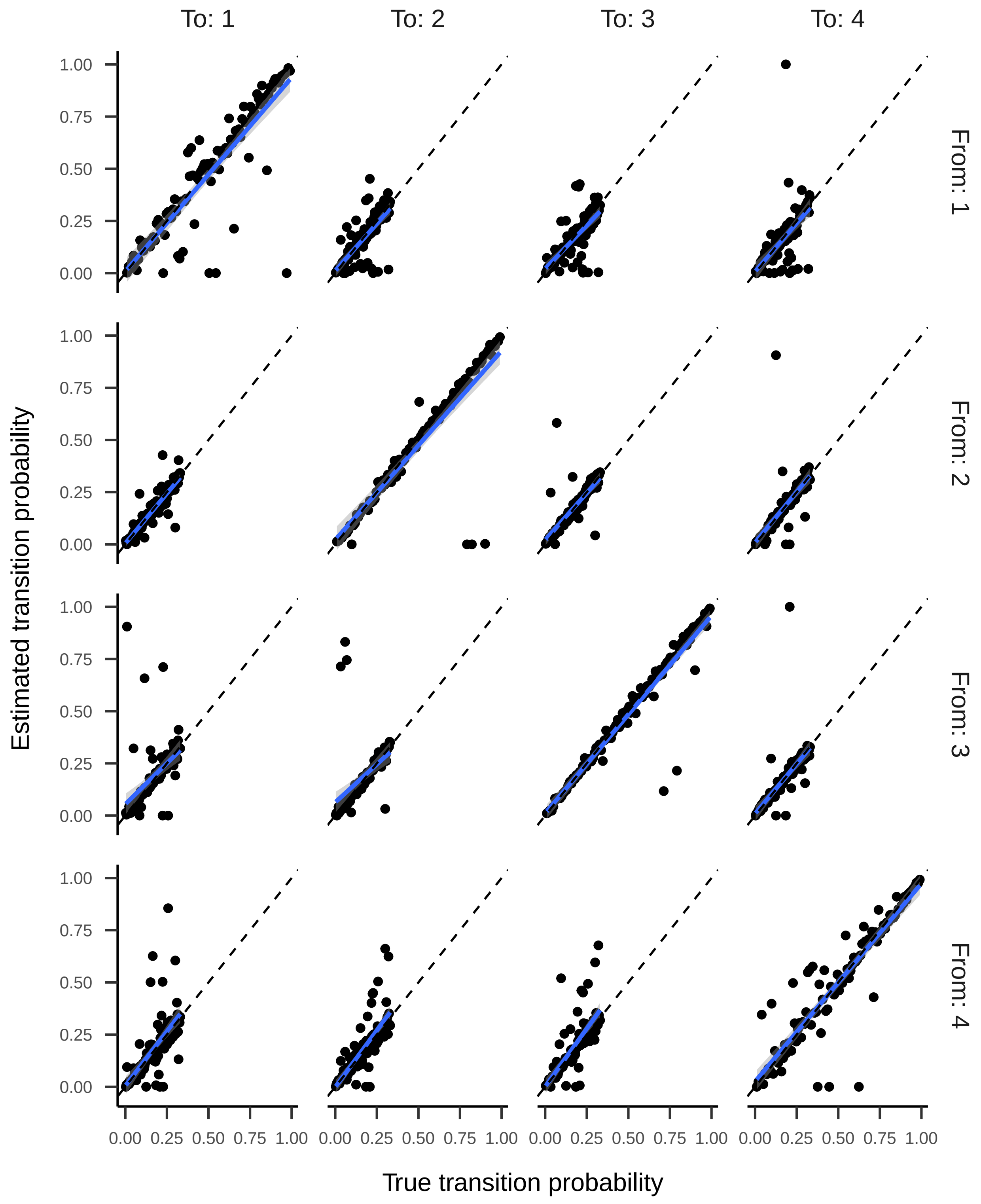

Figure 10. Regression lines between true and estimated transition probabilities for the four-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving. Dashed lines refer to perfect recovery.

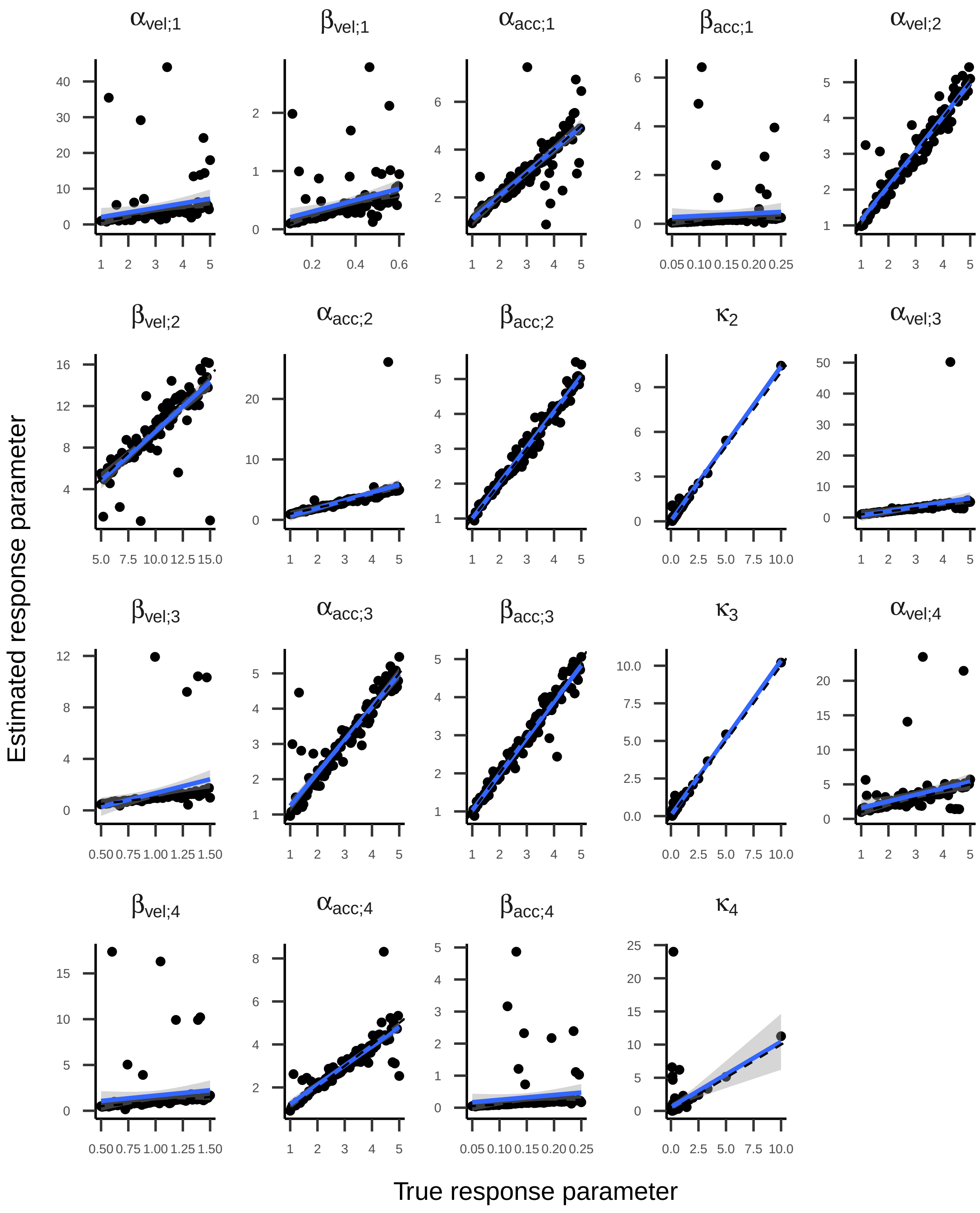

Figure 11. Regression lines between true and estimated response parameters of the four-state HMM in part one. Top facet labels indicate response parameters. Dashed lines refer to perfect recovery.

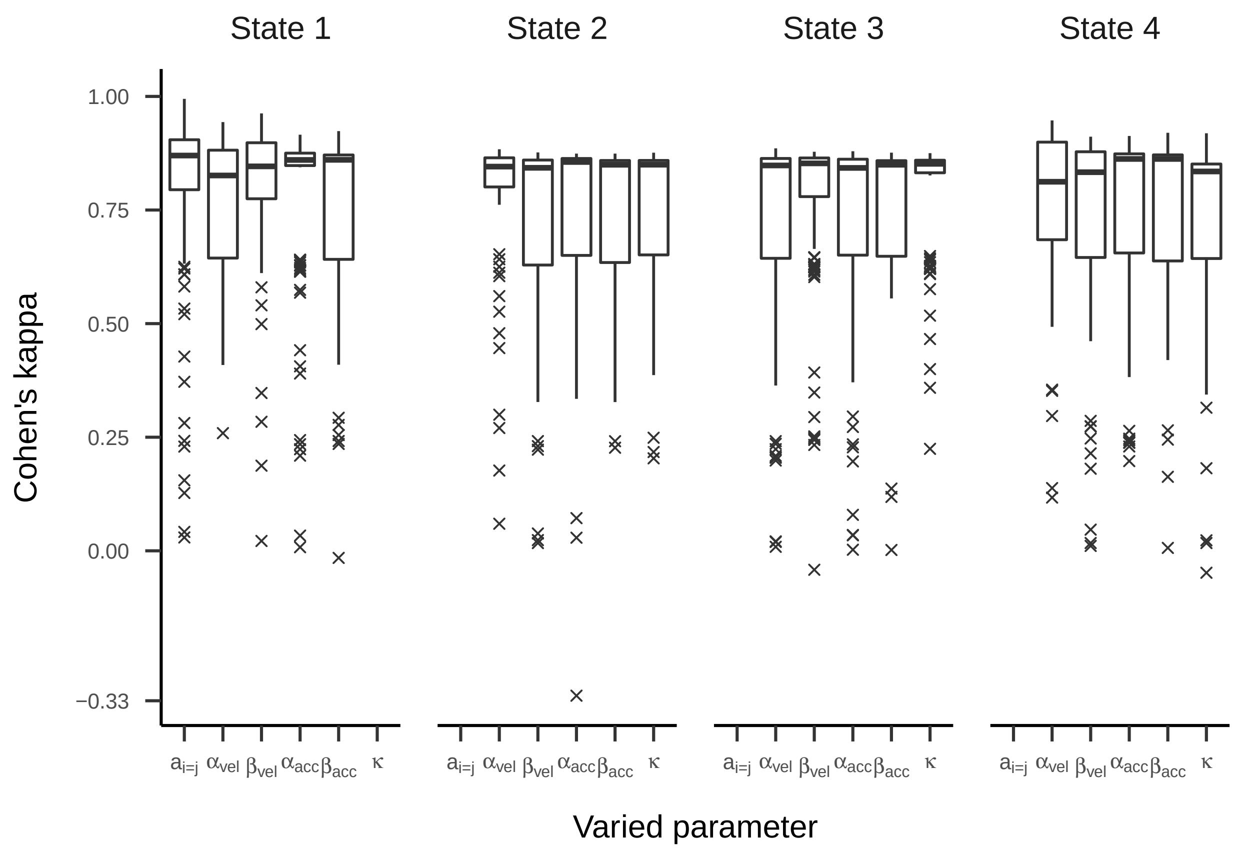

Figure 12. Boxplots displaying Cohen's kappa depending on which parameter of the four-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians and hinges the first and third quartile. Whiskers range from hinges to lowest/highest value within 1.5 times the IQR. Crosses represent outliers.

Table 3. In contrast to the simulation study, generating random starting values often led to bad model fits and label switching between states. To improve the fitting of the gamma distributions, velocity and acceleration signals were scaled down by factor 100[2] (so were the starting values for their gamma distributions).

[2] Scaling down by factor 100 differs from the simulation study (scaling down by 10). The algorithm allows the user to manually specify this factor, and in this case, factor 100 led to better model fits than factor 10.

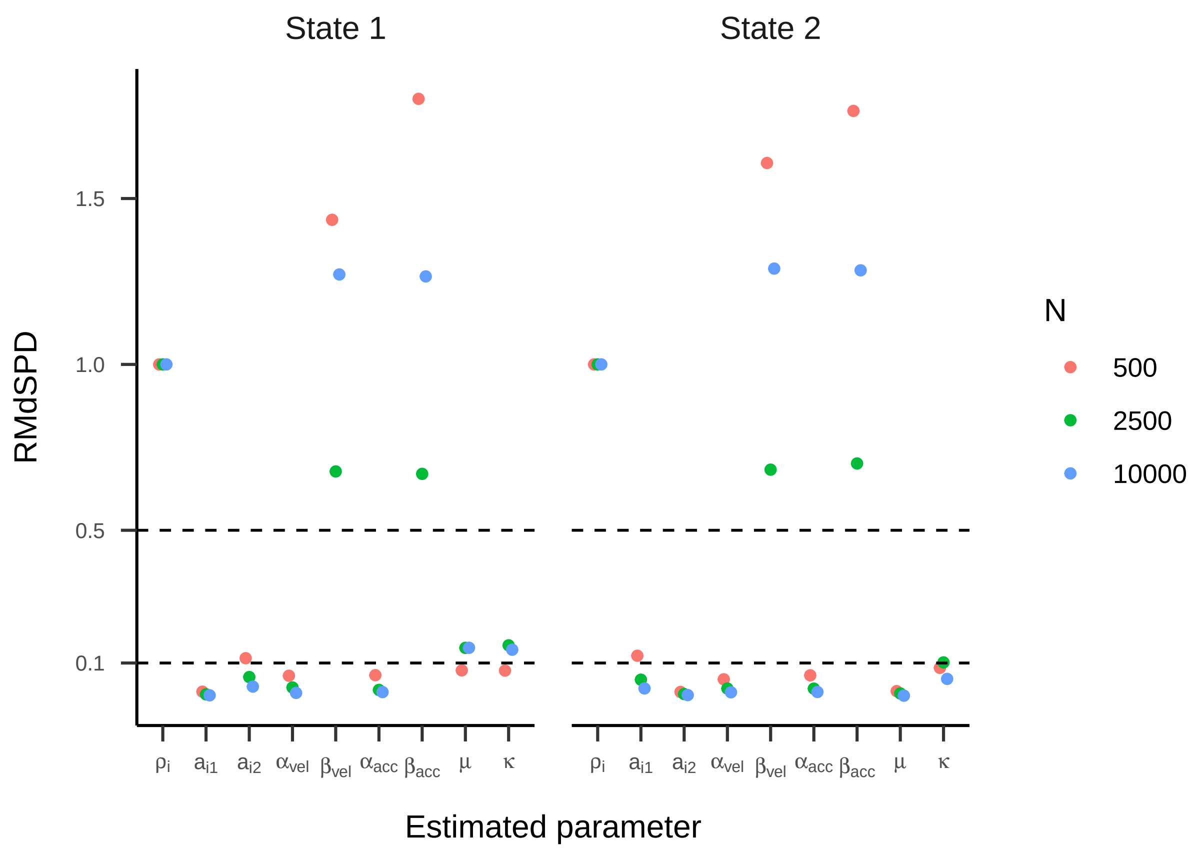

Figure 13. RMdSPD between true and estimated parameters of the two-state HMM in part two of the simulation. Colours indicate different sizes of generated data. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

Data Sets

We chose two data sets for validation: One was published in a study by Andersson et al. (2017) and has been widely used for validation purposes (e.g., Pekkanen & Lappi, 2017). It contains eye-tracking data from three conditions: A static condition, where subjects had to look freely at images, and two dynamic conditions, where they had to follow a constantly moving dot or objects in a video. The data was sampled with 500 Hz and two human coders (MN and RA) labeled them as belonging to six different eye movement events: Fixation, saccade, PSO, smooth pursuit, blink, or other. Andersson et

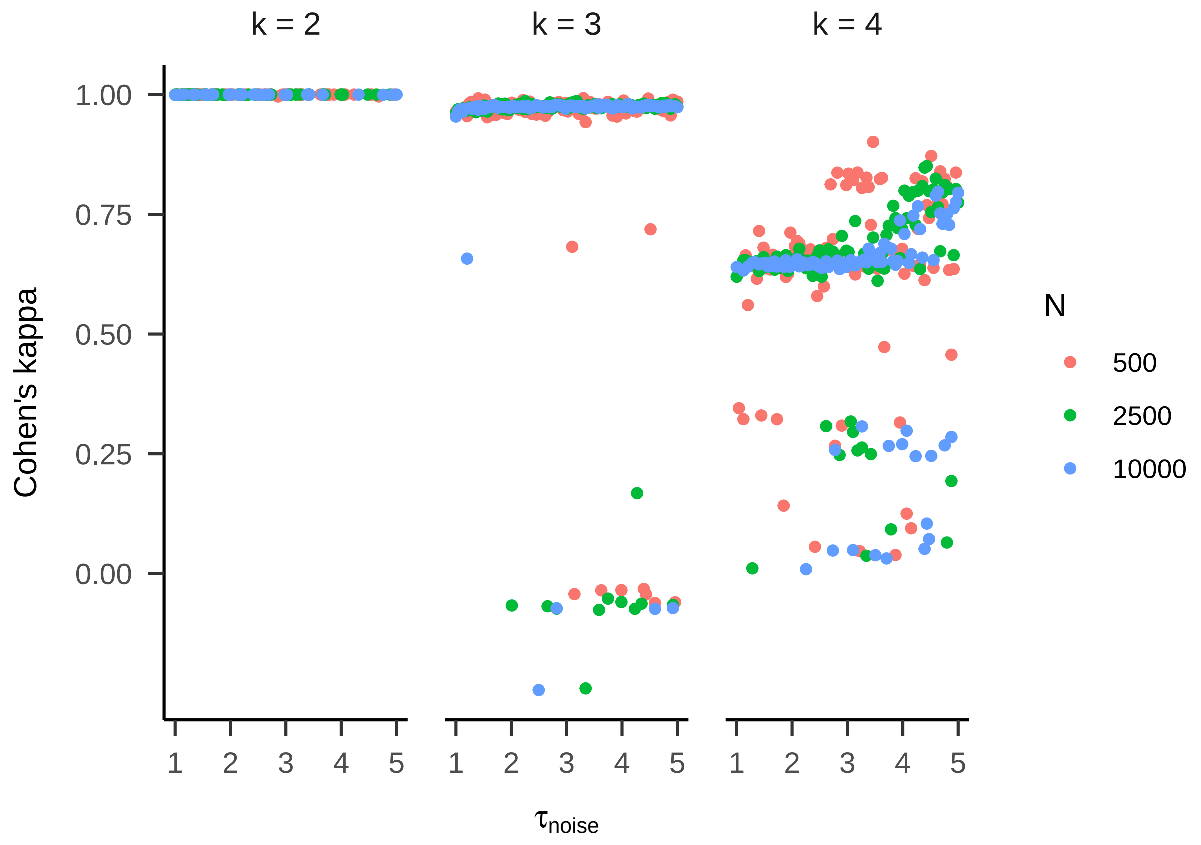

Figure 14. Cohen's kappa depending on the variation of noise added to the data generated by the HMM. Colours indicate different sizes of generated data. Top facet labels indicate the number of states in the HMM.

al. (2017) used the data to compare 10 different classification algorithms. We used their results to compare these 10 algorithms and the two human coders with gazeHMM.

The second data set was published in Ehinger, Groß, Ibs, and Peter (2019) and has to our knowledge not yet been used for validation. Here, we only use tasks four and five out of 10 tasks because these are qualitatively different to the first data set. In task four, subjects were instructed to fixate a central target for 20 s. Task 5 was set up similarly, but subjects had to blink when they heared one out of seven beeps (with a beep duration of 100 ms and 1.5 s intervals in between). Eye movements were recorded with 250-500 Hz. For simplicity, we only used data obtained by the EyeLink (SR Research Ltd., Ontario, Canada)

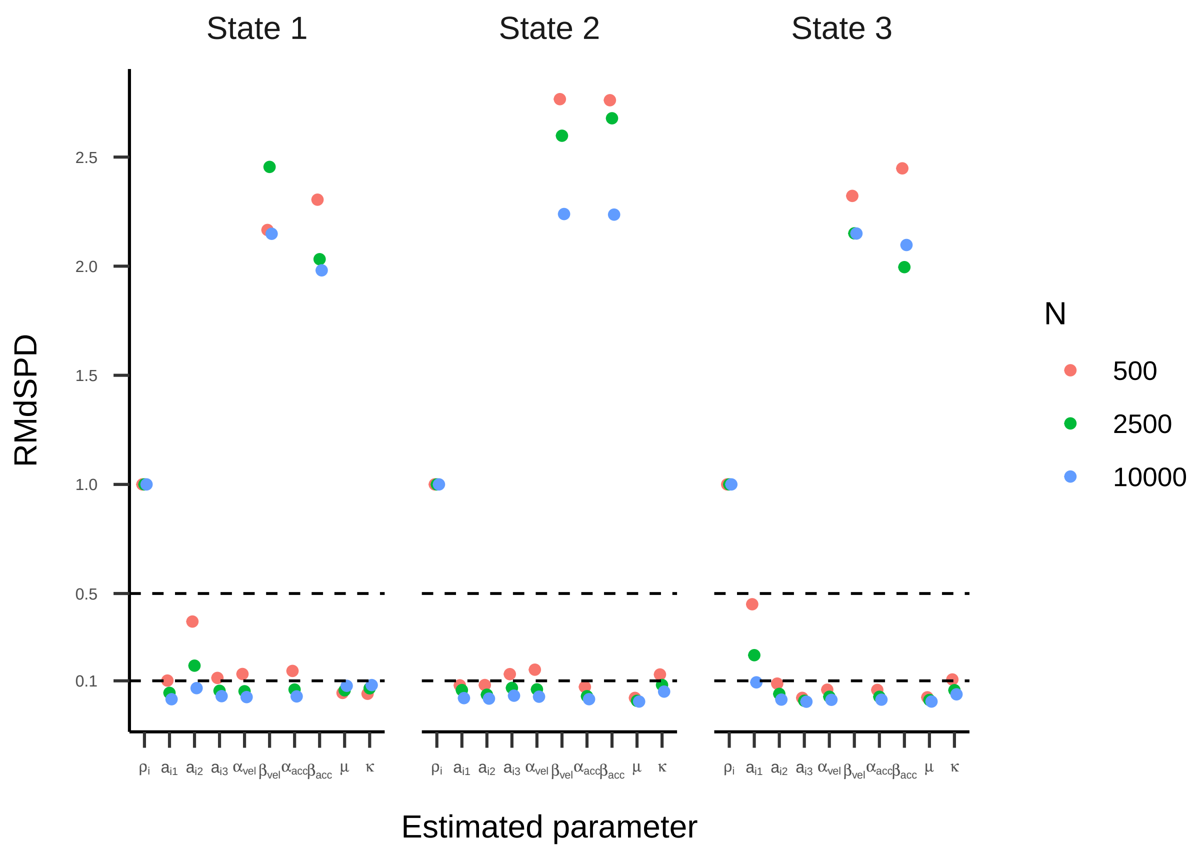

Figure 15. RMdSPD between true and estimated parameters of the three-state HMM in part two of the simulation. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

eye-tracker.

Data Analysis

A successful validation of gazeHMM was determined by using two approaches: First, we applied gazeHMM with generative models containing 1-5 states to both data sets. The fits of the generative models were compared using Schwarz weights (Wagenmakers & Farrell, 2004), a weighted version of the BIC. They can be interpreted as the probability of a model having generated the data compared to the competing models. For the static condition in the Andersson et al. (2017) data set, we expected the generative model with

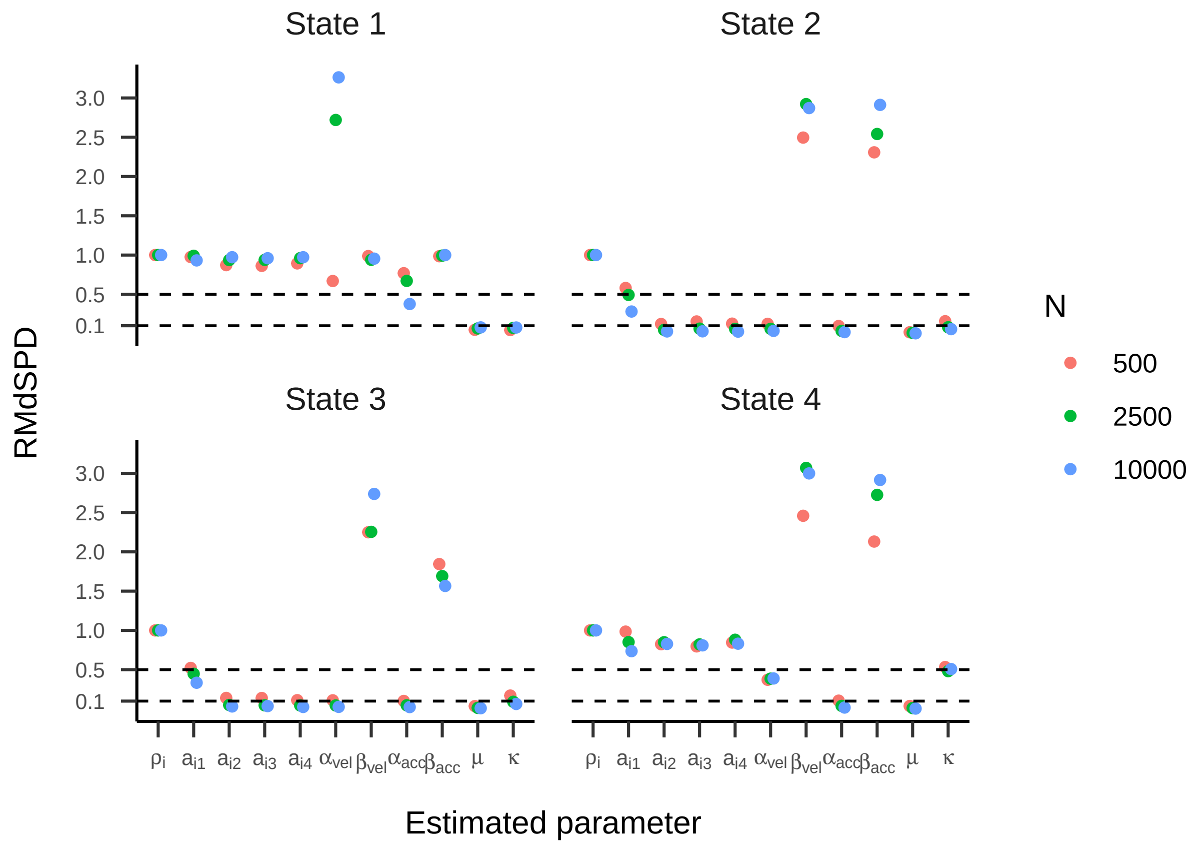

Figure 16. RMdSPD between true and estimated parameters of the three-state HMM in part two of the simulation. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

three states (fixation, saccade, and PSO), and for the dynamic conditions the model with four states (incl. smooth pursuit) to display the highest Schwarz weight. Regarding the Ehinger et al. (2019) data set, we assumed that the one-state model (only fixation) would show the highest weights for both tasks.

The algorithm was applied separately to every subject, condition, or task. For the Andersson et al. (2017) data set, all generative models were successfully fitted, whereas for the Ehinger et al. (2019) data set, it was only 780 out of 900 models (87%, 60 models per task).

Second, we compared gazeHMM to other algorithms and human coders. We applied

Table 3

Starting Values for the Response Model for Fitting gazeHMM to Benchmark Data

	Velocity		Acceleration		Rel. angle	
	Shape	Scale	Shape	Scale	Mean	Concentration
Fixation	10	10	10	10	-	-
Saccade	50	50	50	50	0	10
PSO	50	50	50	50	3.14	10
Pursuit	20	20	20	20	0	10
Event 5	20	20	50	50	0	10

Note. Starting values for velocity and acceleration signals are shown before scaling down by factor 100. Values for event 5 were chosen so that they approximately match plausible distributions for microsaccades.

our algorithm with a three-state generative model to the static condition in the Andersson et al. (2017) data set, and with a four-state model to the dynamic conditions. For comparison criteria, we followed Andersson et al. (2017): We calculated the RMSD of event durations and counts between all algorithms and the two human coders. Cohen's kappa was calculated for each event as the binary agreement between gazeHMM and the human coders. Lastly, the overall disagreement indicated which samples were classified differently by gazeHMM across all events.

Results

Model Comparison. Examining the Schwarz weights displayed in 17, we observed that the five-state generative model showed the highest weights in all three conditions.

Only in the moving dots condition, two subjects each displayed the highest weights for one- and three-state models. In sum, we concluded that the five-state generative model has most likely generated the Andersson et al. (2017) data, opposing our expectations. Because the Ehinger et al. (2019) data set showed a similar pattern, we included the results for this data in the supplementary material.

A recent model recovery study showed that the BIC tended to prefer overly complex HMMs when they were misspecified (e.g., the conditional independence assumption was violated; Pohle, Langrock, Beest, & Schmidt, 2017). Instead, the integrated completed likelihood (ICL) criterion (Biernacki, Celeux, & Govaert, 2000) performed better at choosing the correct data generating model. Therefore, we post hoc computed the weighted ICL criterion (analogous to Schwarz weights) for the models fitted to the Andersson et al. (2017) data set. Using the ICL as the model selection criterion yielded very similar results to the BIC (see supplementary material). The preference for the five-state generative model was even more consistent across conditions and subjects.

Comparison to Other Algorithms. As displayed in 4, gazeHMM showed a relatively low RMSD for fixations in the static condition compared to the other algorithms that were applied to the Andersson et al. (2017) data set. The lower RMSD for fixations indicated more similar classification to the human coders in terms of their mean and SD duration as well as the number of classified fixations. Oppositely, for fixations in the dynamic conditions, the RMSD of gazeHMM was the highest among the compared algorithms, suggesting substantial differences to the human coders. It is likely that these occured because gazeHMM classified a much larger number of fixations with very short durations. For saccades, gazeHMM had high RMSDs for the image and moving dots conditions, but for the video condition the RMSD was comparably low (see Table 5). Only two other algorithms classified PSOs (NH and LNS; Nyström & Holmqvist, 2010; Larsson et al., 2013), and gazeHMM showed consistently higher RMSDs than LNS (see Table 6). Compared to NH, the RMSD was higher in the image and video conditions but lower in

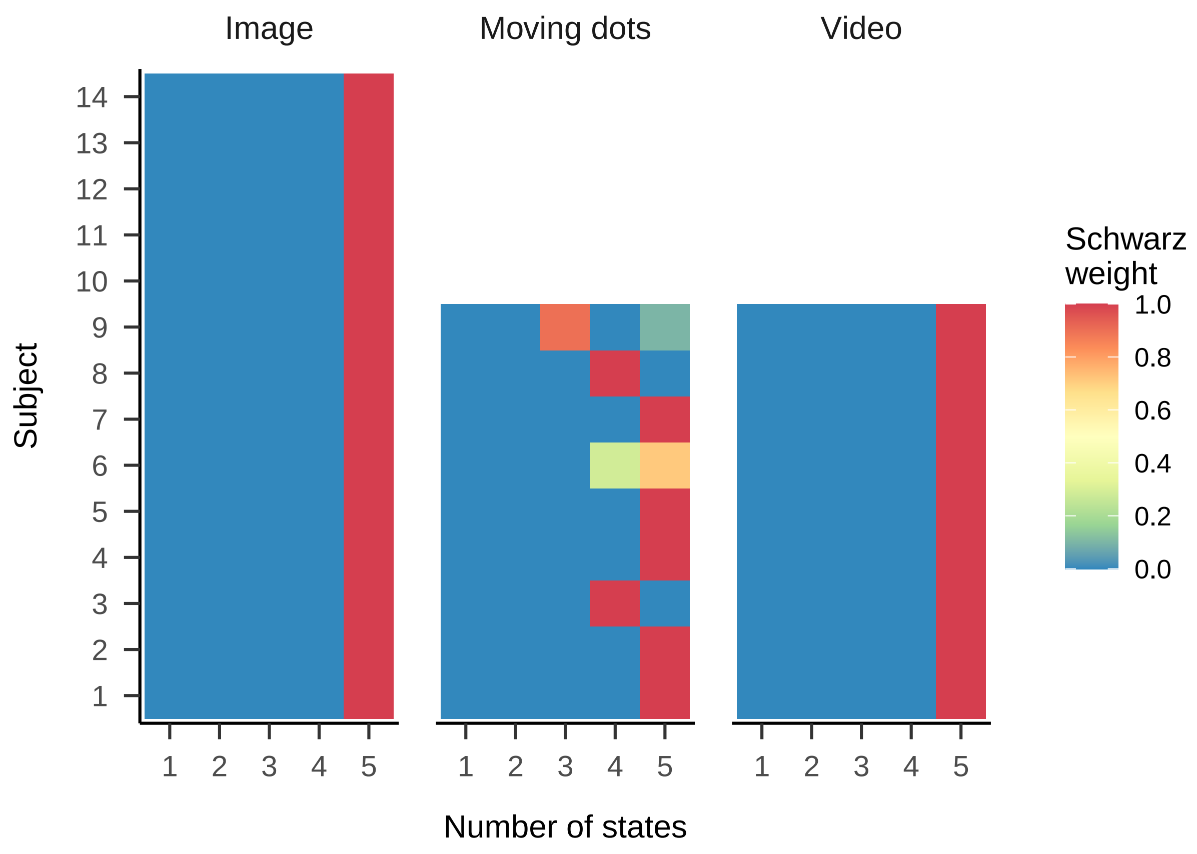

Figure 17. Schwarz weights displayed for each subject and generative models with different numbers of states. Top facet labels indicate the condition in the Andersson et al. (2017) data set. Higher weights indicate a better model fit.

the moving dots condition. No other algorithm parsed smooth pursuits, but the RMSD for gazeHMM was higher than among the human coders (see Table 7). Again, it classified a much larger number of smooth pursuits with short durations.

Table 8 contains the sample-to-sample agreement between the algorithms and human coders measured by Cohen's kappa. For fixations, gazeHMM shows one of the highest agreements for static, and *the* highest agreements for dynamic data. The absolute agreement is substantial for static and slight to fair (Landis & Koch, 1977). For saccades, the relative agreement for gazeHMM is low for the image, moderate for the moving dots, and high for the video condition. In absolute terms, the agreement is moderate to

Table 4

Fixation Duration Descriptives and RMSD Between Algorithms and Human Coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	275.07	285.43	403	0.02	185.64	93.35	11	0.02	338.17	303.11	82	0.16
coderRA	270.89	287.08	391	0.02	174.33	94.28	12	0.02	255.35	185.23	81	0.16
gazeHMM-3	164.60	256.28	578	0.77	-	-	NA	-	-	-	NA	-
gazeHMM-4	-	-	NA	-	15.72	16.58	381	1.13	21.80	24.70	1243	1.20
CDT	464.60	643.25	276	1.28	59.65	114.81	177	0.47	243.74	354.13	226	0.16
EK	-	-	NA	-	-	-	NA	-	-	-	NA	-
IDTk	488.09	579.48	263	1.24	968.00	1,170.95	17	1.82	984.36	1,596.41	62	1.75
IKF	189.58	257.87	518	0.74	269.53	202.90	51	0.16	286.16	315.18	173	0.18
IMST	359.92	471.26	351	0.75	911.67	1,158.97	18	1.75	694.60	1,058.51	87	1.08
IHMM	148.50	235.62	717	0.78	315.83	298.14	47	0.30	259.36	347.56	207	0.16
IVT	127.33	223.43	843	0.80	291.61	298.09	51	0.27	225.09	334.49	240	0.18
NH	282.00	318.49	297	0.53	392.39	336.04	31	0.46	462.25	380.79	89	0.41
BIT	230.34	161.84	439	0.84	193.85	107.64	66	0.14	263.69	225.34	183	0.25
LNS	-	-	NA	-	-	-	NA	-	-	-	NA	-

Note. Durations are displayed in seconds. gazeHMM-3 classified three and gazeHMM-4 classified four events. RMSD = root mean square deviation. Table adapted from Andersson et al. (2017).

substantial. Concerning PSOs, gazeHMM shows higher agreement than NH in the image and video conditions but consistently lower agreement compared to LNS. Absolutely, the agreement is slight to fair. Lastly, the

Additionally, I compared gazeHMM to the other algorithms by using Cohen's kappa as a measure of sample-to-sample agreement between algorithms and human coders (see Table 8). Absolute kappa values were interpreted according to Landis and Koch (1977).

For fixations, absolute kappa values in all three conditions indicated a slight to fair agreement between gazeHMM and human coders. Compared to the other algorithms, the agreement of gazeHMM was the lowest in the image condition but the highest in the moving dots and video condition.

For saccades, gazeHMM showed kappa values that correspond to a moderate to substantial agreement and were relatively high compared to the other algorithms. However,

Table 5

Saccade Duration Descriptives and RMSD Between Algorithms and Human Coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	32.22	15.97	377	0.09	23.21	10.60	38	0.07	27.19	11.78	117	0.07
coderRA	33.85	14.41	374	0.09	21.89	11.60	38	0.07	26.67	11.38	127	0.07
gazeHMM-3	38.99	29.27	657	0.65	-	-	NA	-	-	-	NA	-
gazeHMM-4	-	-	NA	-	26.70	12.21	46	0.26	29.83	20.01	153	0.52
CDT	-	-	NA	-	-	-	NA	-	-	-	NA	-
EK	27.26	23.94	787	0.85	17.90	12.91	59	0.53	22.24	17.73	252	0.97
IDTk	27.96	19.18	258	0.55	33.67	11.48	6	0.50	26.19	19.19	53	0.47
IKF	70.24	39.63	356	1.14	61.62	28.79	21	1.39	61.54	25.26	107	1.14
IMST	19.23	11.82	336	0.88	13.08	4.87	13	1.09	19.64	10.33	76	0.78
IHMM	53.99	28.31	370	0.43	41.05	18.91	19	0.57	45.96	18.00	109	0.39
IVT	46.37	24.56	375	0.16	34.80	14.15	20	0.28	39.66	17.12	112	0.20
NH	55.58	21.03	344	0.37	44.00	14.35	33	0.34	47.34	17.45	104	0.37
BIT	-	-	NA	-	-	-	NA	-	-	-	NA	-
LNS	32.55	13.41	390	0.60	27.05	11.45	42	0.26	30.11	10.40	122	0.59

Note. Durations are displayed in seconds. gazeHMM-3 classified three and gazeHMM-4 classified four events. RMSD = root mean square deviation. Table adapted from Andersson et al. (2017).

gazeHMM never reached the highest agreement.

For PSOs, absolute kappa values showed slight to fair agreement agreement to human coders. It was relatively low in the moving dots condition and second-rate in the image and video conditions compared to NH and LNS.

The agreement for smooth pursuits was moderate in the moving dots condition, fair in the video condition, and slight in the image condition. No other algorithm in the study was designed to detect smooth pursuits.

In sum, gazeHMM revealed a relatively high sample-to-sample agreement to human coders for saccades. For PSOs and smooth pursuits, I found fair to moderate agreement. The performance for fixations was absolutely and relatively low.

Disagreement and Confusion.

Table 6

PSO Duration Descriptives and RMSD Between Algorithms and Human Coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	25.36	13.63	313	0.52	14.42	5.44	24	0.50	23.31	13.26	97	0.30
coderRA	25.00	12.00	310	0.52	14.00	8.06	19	0.50	20.60	12.13	89	0.30
gazeHMM-3	14.25	15.33	518	1.29	-	-	NA	-	-	-	NA	-
gazeHMM-4	-	-	NA	-	6.29	7.98	21	1.18	17.86	14.40	101	1.42
CDT	-	-	NA	-	-	-	NA	-	-	-	NA	-
EK	-	-	NA	-	-	-	NA	-	-	-	NA	-
IDTk	-	-	NA	-	-	-	NA	-	-	-	NA	-
IKF	-	-	NA	-	-	-	NA	-	-	-	NA	-
IMST	-	-	NA	-	-	-	NA	-	-	-	NA	-
IHMM	-	-	NA	-	-	-	NA	-	-	-	NA	-
IVT	-	-	NA	-	-	-	NA	-	-	-	NA	-
NH	30.91	14.71	237	0.72	22.91	12.53	11	0.69	30.97	19.56	78	0.80
BIT	-	-	NA	-	-	-	NA	-	-	-	NA	-
LNS	29.74	14.81	319	0.53	19.62	7.79	21	0.66	29.52	19.18	87	0.78

Note. Durations are displayed in seconds. gazeHMM-3 classified three and gazeHMM-4 classified four events. RMSD = root mean square deviation. Table adapted from Andersson et al. (2017).

At last, I computed the overall disagreement[3] as the percentage of all samples taken together that were classified differently by gazeHMM than the human coders. Figure 18 shows that in the image condition, gazeHMM with four events had a higher overall disagreement by between 10 and 20% than all but two of the other algorithms. In the moving dots and video conditions, the disagreement was between 20 and 30% lower than for all other algorithms. Notably, gazeHMM's disagreement was similar across conditions (around 45%). In summary, gazeHMM had a lower overall disagreement than all other algorithms in the moving dots and video conditions but higher disagreement in the image condition.

[3] Even though I proposed to compute the *disagreement ratio* and Andersson et al. (2017) also used that term, the comparison technically involves percentages and not ratios. Thus, I decided to use only the term *disagreement* instead.

Table 7

Smooth Pursuit Duration Descriptives and RMSD Between gazeHMM and Human Coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	363.42	187.07	3	0.23	370.44	237.62	36	0.40	558.70	390.50	51	0.13
coderRA	299.12	180.06	17	0.23	344.82	338.39	39	0.40	516.40	376.41	70	0.13
gazeHMM-3	-	-	NA	-	-	-	NA	-	-	-	NA	-
gazeHMM-4	-	-	NA	-	22.53	22.24	400	2.01	21.45	23.37	1281	2.01
CDT	-	-	NA	-	-	-	NA	-	-	-	NA	-
EK	-	-	NA	-	-	-	NA	-	-	-	NA	-
IDTk	-	-	NA	-	-	-	NA	-	-	-	NA	-
IKF	-	-	NA	-	-	-	NA	-	-	-	NA	-
IMST	-	-	NA	-	-	-	NA	-	-	-	NA	-
IHMM	-	-	NA	-	-	-	NA	-	-	-	NA	-
IVT	-	-	NA	-	-	-	NA	-	-	-	NA	-
NH	-	-	NA	-	-	-	NA	-	-	-	NA	-
BIT	-	-	NA	-	-	-	NA	-	-	-	NA	-
LNS	-	-	NA	-	-	-	NA	-	-	-	NA	-

Note. Durations are displayed in seconds. gazeHMM-4 classified four events. RMSD = root mean square deviation. Table adapted from Andersson et al. (2017).

Contrary to my second hypothesis, applying gazeHMM to a benchmark data set revealed that it did not outperform all other algorithms in terms of RMSD and sample-to-sample agreement. The two criteria rather indicated a moderate performance of gazeHMM. However, the overall disagreement was lower for gazeHMM than for all other algorithms in the dynamic conditions but the highest in the static image condition.

Looking at the confusion matrix in Table 9, it can be seen that gazeHMM confused fixations and smooth pursuit to a large extend. In the static image condition, it overclassified smooth pursuit events, while they were underclassified in the dynamic conditions.

Table 8

Cohen's Kappa Between Human Coders and Algorithms for Different Conditions and Events

	Fixations			Saccades			PSOs			Smooth pursuits		
Algorithm	Image	Dots	Video	Image	Dots	Video	Image	Dots	Video	Image	Dots	Video
coderMN	0.84	0.84	0.65	0.91	0.79	0.87	0.76	0.57	0.65	0.34	0.81	0.66
coderRA	0.84	0.84	0.65	0.91	0.79	0.87	0.76	0.57	0.65	0.34	0.81	0.66
gazeHMM-3	0.67	0	0	0.36	0	0	0.19	0	0	0	0	0
gazeHMM-4	0	0.16	0.17	0	0.62	0.51	0	0.24	0.46	0	0.28	0.2
CDT	0.38	0.07	0.11	0	0	0	0	0	0	0	0	0
EK	0	0	0	0.64	0.73	0.67	0	0	0	0	0	0
IDTk	0.36	0	0.03	0.45	0.25	0.38	0	0	0	0	0	0
IKF	0.63	0.04	0.14	0.58	0.43	0.59	0	0	0	0	0	0
IMST	0.38	0	0.03	0.54	0.31	0.52	0	0	0	0	0	0
IHMM	0.67	0.03	0.13	0.69	0.58	0.71	0	0	0	0	0	0
IVT	0.67	0.03	0.13	0.75	0.59	0.76	0	0	0	0	0	0
NH	0.52	-0.23	0.01	0.67	0.58	0.68	0.24	0.14	0.25	0	0	0
BIT	0.67	0.02	0.14	0	0	0	0	0	0	0	0	0
LNS	0	0	0	0.81	0.75	0.81	0.64	0.56	0.63	0	0	0

Note. Negative values were set to zero. gazeHMM-3 classified three and gazeHMM-4 classified four events. Table adapted from Andersson et al. (2017).

References

Andersson, R., Larsson, L., Holmqvist, K., Stridh, M., & Nyström, M. (2017). One algorithm to rule them all? An evaluation and discussion of ten eye movement event-detection algorithms. *Behavior Research Methods*, *49*, 616–637. https://doi.org/10.3758/s13428-016-0738-9

Barzilai, J., & Borwein, J. M. (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, *8*, 141–148. https://doi.org/10.1093/imanum/8.1.141

Bellet, M. E., Bellet, J., Nienborg, H., Hafed, Z. M., & Berens, P. (2019). Human-level saccade detection performance using deep neural networks. *Journal of Neurophysiology*, *121*, 646–661. https://doi.org/10.1152/jn.00601.2018

Table 9

Confusion Matrix Between gazeHMM (Rows) and Human Coders (Columns) for Different Conditions

Event	Fixation	Saccade	PSO	Pursuit	Blink	Other
Image						
Fixations	0.88	0.08	0.08	0.72	0.07	0.03
Saccades	0.07	0.59	0.63	0.22	0.14	0.20
PSOs	0.04	0.08	0.23	0.03	0.05	0.16
Pursuits	0.00	0.00	0.00	0.00	0.00	0.00
Blinks	0.01	0.26	0.06	0.03	0.74	0.61
Moving dots						
Fixations	0.61	0.01	0.02	0.35	0.00	0.30
Saccades	0.01	0.78	0.60	0.03	0.00	0.48
PSOs	0.01	0.04	0.18	0.00	0.00	0.00
Pursuits	0.28	0.05	0.17	0.62	0.00	0.20
Blinks	0.10	0.12	0.04	0.00	0.00	0.02
Video						
Fixations	0.54	0.04	0.02	0.44	0.00	0.00
Saccades	0.02	0.57	0.35	0.02	0.14	0.82
PSOs	0.01	0.13	0.47	0.01	0.03	0.05
Pursuits	0.42	0.03	0.12	0.54	0.01	0.00
Blinks	0.00	0.23	0.04	0.00	0.82	0.14

Note. gazeHMM classified four events and blinks.

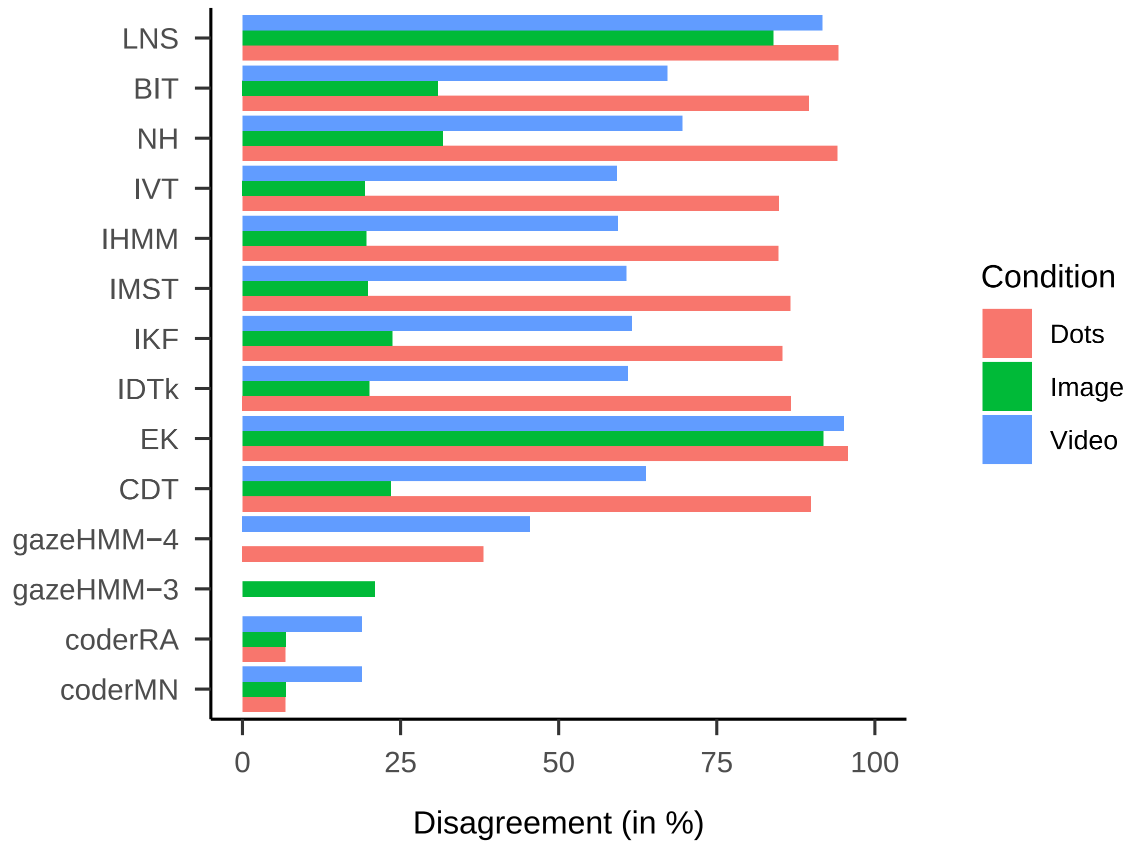

Figure 18. Disagreement between algorithms and human coders for different conditions (in %). gazeHMM-3 classified three and gazeHMM-4 classified four events. Figure adapted from Andersson et al. (2017).

Biernacki, C., Celeux, G., & Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(7), 719–726. https://doi.org/10.1109/34.865189

Birgin, E. G., Martinez, J. M., & Raydan, M. (2000). Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal of Optimization*, *10*, 1196–1211. https://doi.org/10.1137/S1052623497330963

Blignaut, P. (2009). Fixation identification: The optimum threshold for a dispersion algorithm. *Attention, Perception, & Psychophysics*, *71*(4), 881–895.

https://doi.org/10.3758/APP.71.4.881

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *19*(1), 1–38. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

Duchowski, A. T. (2017). *Eye tracking methodology* (3rd ed.). London: Springer. https://doi.org/10.1007/978-3-319-57883-5

Ehinger, B. V., Groß, K., Ibs, I., & Peter, K. (2019). A new comprehensive eye-tracking test battery concurrently evaluating the Pupil Labs glasses and the EyeLink 1000. *bioRxiv*. https://doi.org/10.1101/536243

Forney Jr, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268–278. https://doi.org/10.1109/PROC.1973.9030

Hein, O., & Zangemeister, W. H. (2017). Topology for gaze analyses - Raw data segmentation. *Journal of Eye Movement Research*, *10*(1), 1–25. https://doi.org/10.16910/jemr.10.1.1

Houpt, J. W., Frame, M. E., & Blaha, L. M. (2018). Unsupervised parsing of gaze data with a beta-process vector auto-regressive hidden Markov model. *Behavior Research Methods*, *50*, 2074–2096. https://doi.org/10.3758/s13428-017-0974-7

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Kasneci, E., Kasneci, G., Kübler, T. C., & Rosenstiel, W. (2014). The applicability of probabilistic methods to the online recognition of fixations and saccades in dynamic scenes. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 323–326. https://doi.org/10.1145/2578153.2578213

Komogortsev, O. V., Gobert, D. V., Jayarathna, S., Koh, D. H., & Gowda, S. M. (2010). Standardization of automated analyses of oculomotor fixation and saccadic behaviors. *IEEE Transactions on Biomedical Engineering*, *57*(11), 2635–2645. https://doi.org/10.1109/TBME.2010.2057429

Landis, R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, *33*(1), 159–174.

Larsson, L., Nystrom, M., & Stridh, M. (2013). Detection of saccades and postsaccadic oscillations in the presence of smooth pursuit. *IEEE Transactions on Biomedical Engineering*, *60*(9), 2484–2493. https://doi.org/10.1109/TBME.2013.2258918

Leigh, R. J., & Zee, D. S. (2015). *The neurology of eye movements* (5th ed.). Oxford University Press.

Ligges, U., Short, T., & Kienzle, P. (2015). signal: Signal processing. Retrieved from http://r-forge.r-project.org/projects/signal/

McLachlan, G. J., & Krishnan, T. (1997). *The EM algorithm and extensions.* New York: Wiley.

Mihali, A., Opheusden, B. van, & Ma, W. J. (2017). Bayesian microsaccade detection. *Journal of Vision*, *17*(1), 1–23. https://doi.org/10.1167/17.1.13

Nyström, M., & Holmqvist, K. (2010). An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods*, *42*(1), 188–204. https://doi.org/10.3758/BRM.42.1.188

Olsson, P. (2007). *Real-time and offline filters for eye tracking* (Master Thesis). Royal Institute of Technology Stockholm. Retrieved from http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=pubmed%7B/&%7Dcmd=Retrieve%7B/&%7Ddopt=AbstractPlus%7B/&%7Dlist%7B/_%7Duids=

12935495724606901081related:WUvHitMchLMJ

Pekkanen, J., & Lappi, O. (2017). A new and general approach to signal denoising and eye movement classification based on segmented linear regression. *Scientific Reports*, 1–13. https://doi.org/10.1038/s41598-017-17983-x

Pohle, J., Langrock, R., Beest, F. M. van, & Schmidt, N. M. (2017). Selecting the number of states in hidden Markov models: Pragmatic solutions illustrated using animal movement. *Journal of Agricultural, Biological, and Environmental Statistics*, *22*(3), 270–293. https://doi.org/10.1007/s13253-017-0283-8

R Core Team. (2020). R: A language and environment for statistical computing. Wien: R Foundation for Statistical Computing. Retrieved from https://www.r-project.org/

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 71–78. https://doi.org/10.1145/355017.355028

Savitzky, A., & Golay, M. J. E. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, *36*(8), 1627–1639. https://doi.org/10.1021/ac60214a047

Schulte-Mecklenbeck, M., Johnson, J. G., Böckenholt, U., Goldstein, D. G., Russo, J. E., Sullivan, N. J., & Willemsen, M. C. (2017). Process-tracing methods in decision making: On growing up in the 70s. *Current Directions in Psychological Science*, *26*(5), 442–450. https://doi.org/10.1177/0963721417708229

Shic, F., Scassellati, B., & Chawarska, K. (2008). The incomplete fixation measure. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 111–114. https://doi.org/10.1145/1344471.1344500

Stuart, S., Hickey, A., Vitorio, R., Welman, K., Foo, S., Keen, D., & Godfrey, A.

(2019). Eye-tracker algorithms to detect saccades during static and dynamic tasks: A structured review. *Physiological Measurement*, *40*(2), 1–26. https://doi.org/10.1088/1361-6579/ab02ab

Tafaj, E., Kasneci, G., Rosenstiel, W., & Bogdan, M. (2012). Bayesian online clustering of eye movement data. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 285–288. https://doi.org/10.1145/2168556.2168617

Varadhan, R., & Gilbert, P. (2009). BB: An R package for solving a large system of nonlinear equations and for optimizing a high-dimensional nonlinear objective function. *Journal of Statistical Software*, *32*(4), 1–26. Retrieved from http://www.jstatsoft.org/

Visser, I. (2011). Seven things to remember about hidden Markov models: A tutorial on Markovian models for time series. *Journal of Mathematical Psychology*, *55*(6), 403–415. https://doi.org/10.1016/j.jmp.2011.08.002

Visser, I., & Speekenbrink, M. (2010). depmixS4: An R package for hidden Markov models. *Journal of Statistical Software*, *36*(7), 1–21. https://doi.org/10.18637/jss.v036.i07

Visser, I., & Speekenbrink, M. (2019). *Mixture and hidden Markov models with R including latent class models*. Manuscript in preparation.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, *13*(2), 260–269. https://doi.org/10.1109/TIT.1967.1054010

Wagenmakers, E. J., & Farrell, S. (2004). AIC model selection using Akaike weights. *Psychonomic Bulletin and Review*, *11*(1), 192–196. https://doi.org/10.3758/BF03206482

Zemblys, R., Niehorster, D. C., Komogortsev, O., & Holmqvist, K. (2018). Using machine learning to detect events in eye-tracking data. *Behavior Research Methods*, 160–181. https://doi.org/10.3758/s13428-017-0860-3

Zucchini, W., MacDonald, I. L., & Langrock, R. (2016). *Hidden Markov models for time series - An introduction using R* (2nd ed.). Boca Raton, FL: Chapman & Hall. https://doi.org/10.1201/b20790