Keeping an Eye on Hidden Markov Models in Gaze Data Classification

Malte Valentin Lueken

Author Note

Affiliation: University of Amsterdam, Graduate School of Psychology, Department of Methodology and Statistics.

This manuscript constitutes the internship report as part of the Research Master's Psychology.

Student number: 12750166. Email address: malte.luken@student.uva.nl.

Specialization: Methodology and Statistics. Supervisor: dhr. dr. Ingmar Visser.

I would like to thank Simon Kucharsky and Ingmar Visser for their helpful advice on developing and implementing this research project.

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. I developed gazeHMM, an algorithm that improves on previous HMM algorithms. The development was guided by the question of whether HMMs are useful at describing eye movements and whether they improve the event classification. A simulation study showed that gazeHMM successfully recovered HMM parameters and hidden state sequences with only few exceptions. When applied to benchmark data, model comparisons for gazeHMM with different events did not match theoretical expectations. Compared to other algorithms, gazeHMM improved the event classification which was assessed by the similarity to human coding. The classification of smooth pursuits was found to be inadequate and is thus only recommended for exploration. Otherwise, I consider gazeHMM as appropriate for practical applications. In sum, gazeHMM improves classification but is not accurately describing eye movements, yet. I identify several explanations and discuss potential remedies for this result.

Keywords: keywords

Word count: X

Keeping an Eye on Hidden Markov Models in Gaze Data Classification

Introduction

For how long did humans seek to open a window into each other's minds, revealing their hidden cognitive processes? While many methods aiming to achieve this have been developed in the last decades, eye-tracking as one of them has become more and more reliable, accurate, and widely applied in cognitive psychology (Duchowski, 2017). 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 pursuit, and post-saccadic oscillations (PSOs). Classifying eye-tracking data reduces their complexity and is the first step towards cognitive interpretation (Andersson, Larsson, Holmqvist, Stridh, & Nyström, 2017; Salvucci & Goldberg, 2000).

The classification is normally done by algorithms, which is considered faster, more objective, and reproducible compared to human coding (Andersson et al., 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). As illustrated in Figure 1, most classification algorithms consist of four steps: First, the raw gaze position is transformed into a metric of position and/or time. Second, a filtering or smoothing procedure is applied to the data to separate the gaze signal from noise and artifacts (Spakov, 2012). Third, depending on the method and settings of the algorithm, each sample is labeled as a candidate for one of the predefined events. Fourth, the algorithm decides which candidates to accept, relabel, or merge (Hessels, Niehorster, Kemner, & Hooge, 2017; Komogortsev, Gobert, Jayarathna, Koh, & Gowda, 2010).

A frequently implemented method uses probabilistic models 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). Figure 2 illustrates the structure of HMMs: Parallel to the gaze signal evolves a sequence of distinct states that cannot be directly observed. Each gaze sample depends on its corresponding state. Each state depends on the previous but not on earlier states of the sequence (Cappé, Moulines, & Ryden, 2005; Visser & Speekenbrink, 2019).

In the context of eye movements, HMMs are suitable probabilistic models because the hidden states can be interpreted as events and gaze data are dependent time series. 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. Recently, 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 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 with a human coder (Andersson et al., 2017; Komogortsev et al., 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, p. @Pekkanen2017). 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. 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.

HMMs can also be used as a generative model to make inferences about the underlying processes of eye movements, a property that has been rarely used in the context of classification (cf. Mihali et al., 2017). For example, an HMM with four events could be compared to one with three events to see whether an event is present in the data. I propose to develop an algorithm that uses HMMs to classify eye movement data. It will address the weaknesses of previous HMM algorithms and utilize the generative property of HMMs.

Research Question

Can HMMs improve describing and classifying eye-tracking data? This is the general research question of my proposal and can be specified as following: Are HMMs useful generative models for recorded eye movements that improve classifying them into events? This research question will guide the development of the algorithm and leads to two hypotheses: First, I predict that HMMs are useful generative models for eye movements. This hypothesis will be supported if the algorithm yields events that match theoretical expectations for benchmark data. Other results will be treated as evidence against the hypothesis. Second, I hypothesize that HMMs will improve the classification performance of eye movements. I will treat it as support for this hypothesis if the algorithm outperforms algorithms using other methods when applied to benchmark data. I will treat other results as evidence against the hypothesis.

Operationalization

The proposed algorithm aims to improve the drawbacks of previous HMM algorithms. Therefore, its classification method is oriented to four major goals: (a) it does not require parameter settings through the user (e.g., thresholds) or (b) human-labeled data as input; (c) it covers the most relevant eye movement events, namely fixations, saccades, smooth pursuit, and PSOs; (d) it confirms rather than explores the presence of events in the data. Besides implementing the algorithm in the statistical programming language R R Core Team (2020), I will develop its concrete technical design exploratorily.

The first step will be choosing an appropriate metric that allows separating the gaze data into events. Zemblys, Niehorster, Komogortsev, and Holmqvist (2018) provide an overview of different gaze data metrics and compare how well they predict human-labeled events. The next step will choosing a filtering/smoothing method to reduce the noise in the gaze data (for a comparison of different gaze data filters, see Spakov, 2012). The criteria for choosing a metric and a filter will be its accuracy and not relying on user parameter settings.

The generative part of the algorithm will allow users to build an HMM which matches their theoretical assumptions about the gaze data. The algorithm will take a preprocessed gaze data metric (e.g., filtered velocity) as input and optimize the model parameters. The output of the model will be probabilities for hidden states. The states can be interpreted as event labels for each sample. For instance, the model could assign to a sample a probability of 0.6 for belonging to a fixation and 0.4 for belonging to a saccade. These probabilities can be used to compute the hidden state sequence (e.g., by choosing events with the highest probability). Moreover, the model will output the parameters of the distribution of labeled samples for each event. It will allow generating samples from these distributions to simulate gaze data.

Before testing the hypotheses, I will conduct a simulation study to estimate the parameter recovery of the generative model: Artificial eye movement data will be generated by the model. The parameters of the model will be varied across sensible ranges. Then, the model will be applied to the data. Finally, the estimated results of the model will be compared to the true parameters that have been used to generate the data.

The first hypothesis will be tested by comparing different generative models on benchmark data. The models will differ in their assumed number of events in the data. The model comparison will result in a preferred model which scores best on the evaluation criterium. The preferred model is expected to match the theoretically assumed number of present events in the benchmark data.

To test the second hypothesis, the proposed algorithm will be compared against other algorithms that have also been applied to benchmark data. The performance of the proposed algorithm will be assessed by metrics that are commonly used in classification (e.g., Cohen's Kappa).

Sample Characteristics

The hypotheses will be tested on two benchmark data sets that have been published in previous studies. The first set was published in Andersson et al. (2017) and contains eye-tracking data from 17 subjects. Three types of stimuli were presented on a screen (1024 × 768 pixels) to each subject: A static image, a linearly moving dot, and a video. Subjects were instructed to look freely at the images and follow the movements of the dots and the objects in the videos. Their eye movements were recorded with a Hi-Speed 1250 tower-mounted eye-tracker (SensoMotoric Instruments) and a sampling rate of 500 Hz. Only data from the right eye were used. The data were labeled by two human experts in eye movement research as fixations, saccades, smooth pursuits, PSOs, or blinks (Andersson et al., 2017, p. 621). The second data set, published by Ehinger, Groß, Ibs, and Peter (2019), consists of eye-tracking data from 15 subjects who performed six blocks of 10 different experimental tasks. Here, I will only consider data from Tasks 4 and 5 because they are qualitatively different from the first data set. In Task 4, subjects were instructed to fixate a central target on the screen (1920 × 1080 pixels) for 20 s. In Task 5, subjects also fixated a central target but were instructed to blink each time they heard a beep. Seven beeps with a duration of 100 ms and a 1.5 s interval in between were presented. For both tasks, eye-movements were recorded by two eye-trackers simultaneously (EyeLink 1000, SR Research; and Pupil Labs glasses, Pupil Labs) with a sampling rate of 250-500 Hz and 240 Hz (interpolated), respectively (Ehinger et al., 2019, pp. 3–11).

Data Analysis

I will treat the parameter recovery of the generative model as satisfactory when it yields a 95% CI covering the true parameters. The median rate of correct classifications should be a minimum of 90%. The first hypothesis will be tested using a model comparison approach: For both data sets, models with one, two, three, four, and five eye movement events plus a noise event will be compared. The noise event will represent blinks. For the first data set and static images, I expect the model with three events (excl. noise) to be the preferred model since smooth pursuit movements are not assumed to be present. For moving dots and video stimuli, I expect the model with four events (excl. noise) to be preferred, since smooth pursuits are expected in the data. For the second data set, I expect a model with one event (excl. noise) to be preferred because only fixations should be present in the data from both tasks. For all comparisons, the preferred model will be chosen according to the highest Schwarz weight (Wagenmakers & Farrell, 2004). I will test the second hypothesis using the metrics that have been used in the original study of the first data set: Andersson et al. (2017) compared ten algorithms to human-labeled data by the root mean squared error (RMSE) of event distribution descriptives, the sample-to-sample agreement indicated by Cohen's Kappa, and the disagreement ratio across all samples indicated by the confusion matrix. I expect the algorithm with a generative model assuming four events (excl. noise) to achieve either a lower RMSE, higher Cohen's Kappa, or lower disagreement ratio than all the other algorithms which are evaluated in the study.

Intended Results

Preferred generative models that match the theoretical assumptions on the number of events in the data would support the notion that HMMs describe gaze data appropriately. If a generative model with fewer events than expected is preferred, this would mean that the model is not able to distinguish the events well enough. If more events than expected are preferred, this would indicate that the model is describing noise or an unexpected event is present in the data. Outperforming other algorithms in agreement to human-coded data would demonstrate that the proposed algorithm classifies events similar to human coders. However, Hooge, Niehorster, Nyström, Andersson, and Hessels (2018) observed considerable differences in how humans code gaze data. They concluded that human coders cannot be seen as a gold standard for event classification. The proposed algorithm would be the first allowing the user to conduct model comparisons as part of the classification. This comparison could be used to investigate the data more thoroughly on a qualitative (e.g., assumed present events) or quantitative (e.g., event sample distributions) level. In contrast to many other classification algorithms, the proposed algorithm would not only classify gaze data but also explain why it is classified this way by using a generative model. This outstanding property of the algorithm would guarantee its competitiveness against other popular algorithms, especially the recently developed machine learning approaches which do not make generative assumptions (e.g., Bellet et al., 2019). It could spark a shift from pure classification output towards understanding eye movements on a more fundamental level.

Algorithm Development

As part of answering my research questions, I developed an algorithm named gazeHMM to classify gaze data into discrete eye movement events. The following section describes the development of gazeHMM and its final version that has been used to obtain results. Technical details can be found in Appendix B.

Eye Movement Metrics

Many different metrics can be used to describe gaze data and eye movement events (Zemblys et al., 2018). Here, the goal is to find those metrics that separate the gaze data samples belonging to different events the best. However, many metrics 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; Zemblys et al., 2018). This can be problematic because these parameters are often set without theoretical justification and they differ substantially between metrics. Belonging to the most basic metrics, which do not require parameter settings, I used velocity, acceleration, and sample-to-sample angle (similar to relative or change in angle; Larsson, Nystrom, & Stridh, 2013) in gazeHMM.

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 after high velocity saccades (Larsson et al., 2013). They are specifically distinguished by their change in direction clustered around 180 degrees (Pekkanen & Lappi, 2017). 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).

Fitering and Smoothing

As with metrics, many methods can be used to filter or smooth gaze data before, while, or after computing eye movement metrics. Their purpose is to remove noise or artifacts from the gaze data that could distort the classification (Duchowski, 2017). Filtering (smoothing) methods that are applied before computing the eye movement metrics target the recorded gaze position. While removing noise, filters might also erase differences in the metrics between samples. For instance, tremor movements during fixations could be filtered out and in consequence, the samples would follow the same direction. This could complicate separating fixations from smooth pursuits. The same problem applies to PSOs and saccades where filtering could erase oscillations (see Figure 3).

Instead of filtering before computing the metrics, I decided to combine both in one step. Previous algorithms have used two methods that both filter and compute derivatives of the gaze position (i.e., velocity and acceleration). Houpt et al. (2018) compute the first and second discrete derivative (similar to a Sobel and Laplace filter, respectively) of the gaze position to be used by their algorithm. Similarly, Nyström and Holmqvist (2010) use a Savitzky-Golay (SG) filter Savitzky and Golay (1964) to estimate velocity and acceleration from a gaze signal. Figure 4 displays velocity and acceleration signals obtained by both methods. They follow a similar trend, but the SG velocity signal is less noisy and the acceleration signal is consequently lower than the Laplace acceleration signal. Therefore, I chose to implement the SG filter to compute velocity and acceleration signals, because it filters out more noise than the discrete derivatives but still preserves the edges in the signal to distinguish between events. To preserve motor noise in the sample-to-sample angle signal, gazeHMM computes the absolute angle with the backward difference and the first discrete derivative as the forward difference (see Appendix B).

The Generative Model

The generative model underlying gazeHMM is a multivariate hidden Markov model (HMM). It can contain between two and four states that correspond to different eye movement events: The first state always represents fixations, the second saccades, the third PSOs, and the fourth smooth pursuits. Thus, users can choose wether they would like to classify only fixations and saccades, or additionally PSOs and/or smooth puruits.

In general, HMMs consist of three submodels: An initial state model, a transition model, and a response model (Visser & Speekenbrink, 2019). In gazeHMM, the response model consists of three response variables which are the velocity and acceleration signals obtained by the SG filter and the change in angle signal. The response variables are treated as conditionally independent on the states. I acknowledge that 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 identifyable.

Previous algorithms using HMMs 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) and due to noise above zero. 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, I use gamma distributions with a shape and scale parametrization for this purpose. It has to be noted that the gamma was choosen out of convenience and the best fitting distribution might be different between eye-trackers, subjects, and tasks.

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

The initial state model and the transition model use multinomial distributions (with a category for each state). For all three submodels of the HMM, no time-varying covariates were included.

Missing Data and Blinks

HMMs can estimate parameters and hidden states despite missing data. They either treat data as missing at random or as missing depending on states (Visser & Speekenbrink, 2019). In gazeHMM, I assume data to be missing at random. However, most of the missing data in eye movement classification are due to blinks. Therefore, the user can indicate in gazHMM 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, I decided to remove them heuristically. Before classifying the samples, gazeHMM 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.

When evaluating the likelihood of the HMM given parameters and the joint density of response distributions for missing data, the model integrates over all possible values. The resulting density for missing samples is therefore set to one (Visser & Speekenbrink, 2019).

Optimization and Classification

The paramters of the HMM are estimated through maximum likelihood using an expectation-maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977; McLachlan & Krishnan, 1997). The EM 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 (velocity, acceleration, and sample-to-sample angle) and the expected hidden states (eye movement events; Visser & Speekenbrink, 2019). 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) were 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 steplengths (Barzilai & Borwein, 1988).

Postprocessing

After classifying gaze samples into states, gazeHMM applies a postprocessing routine to the estimated state sequence. I implemented this rountine 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 did not explicitly control the duration of events in the HMM which ocassionally 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 were relabeled as the state of the previous event. Finally, samples initially indicated as missing were labeled as noise (including blinks) and event metrics were computed (e.g., fixation durations).

Implementation

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, I 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). The major part of this section is a direct copy of the preregistration. When appropriate, additional explanations were added and tenses adapted to make the design easier to read and understand.

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

Design

Parameter Variation. The simulation study was divided into four parts. In the first part, I varied the parameters of the HMM. For models with $k \in \{2, 3, 4\}$ states, $q \in \{10, 15, 20\}$ parameters were varied 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 was varied 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). I did not manipulate the initial state probabilities because these are usually irrelevant in the context of eye movement classification. For the transition probabilities, I only simultaneously varied the probabilities for staying in the same state (diagonals of the tranistion matrix) to reduce the complexity of the simulation. The left over probability mass was split evenly between the probabilites for switching to a different state (per row of the transition matrix). Moreover, I 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.

I defined approximate ranges for each response variable and chose true parameter intervals and default values so that they produced samples that roughly corresponded to these ranges. Table 1 shows the assumed ranges for each event and Tables 2 and 3 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. I 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 A1.

Sample Size and Noise Variation. In the second part, I varied 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 $N \in \{500, 2500, 10000\}$, I 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, I 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 drawn from moderately skewed distributions with modes equal to the original values. To angle, I 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 varied simultaneously in equidistant steps in their intervals.

Variation of Starting Values. In the third part, I increased the variation in the starting values used for parameter estimation. The model parameters were set to their default values. For the shape, scale, and concentration parameters, I simultaneously increased the scale parameters of the starting value gamma distributions: For $k \in \{2, 3, 4\}$ states and $\beta_{start} = (\psi_{true}/2)\tau_{start}$ with $\tau_{start} \in \{1, 2, 3\}$, 100 data sets with $N = 2500$ samples were generated each ($100 \times 3 \times 3 = 900$ recoveries).

Missing Data. In the last part, I set intervals of the generated data to be missing. The model parameters were set to their default values. For $k \in \{2, 3, 4\}$ states and $m \in \{1, 3, 5\}$ intervals, 100 data sets with $N = 2500$ samples were generated ($100 \times 3 \times 3 = 900$ recoveries). The length of the missing data interval $l \in [1, 200]$ samples varied in equidistant steps between the data sets.

Starting Values

The HMM always started with a uniform distribution to estimate the initial state and state transition probabilities. To generate random starting values for the estimation of shape, scale, and concentration parameters, I used 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.

Data Analysis[1]

For each parameter, I 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[2]:

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

I treated $\text{RMdSPD} < 0.1$ as good, $0.1 \leq \text{RMdSPD} < 0.5$ as moderate, and $\text{RMdSPD} \geq 0.5$ as bad recovery of a parameter. By taking the median, I reduced the influence of potential outliers in the estimation and using proportions enabled me to compare RMdSPD values across parameters and data sets.

Additionally, I 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, I 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. Higher kappa values were interpreted as better model accuracy. I adopted the ranges proposed by Landis and Koch (1977) to interpret kappa values. Models that could not be fitted were excluded from the recovery.

[1] Note that this section deviates from my proposed analysis plan. However, since I preregistered the changed analysis plan, I treated the results of the simulation as confirmatory.

[2] 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, I used $\psi_{true} = 2\pi$ instead.

Results[3]

Simulation Study

Parameter Variation.

Two States.

In the first part of the simulation, I examined how varying the parameters in the HMM affects the deviation and accuracy between the true and estimated parameters and state sequence. Figure 5 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, that is, the median deviation never exceeded 10% of the true parameters.[4] The regressions between manipulated true and estimated parameters are shown in Figures 6 and 7. With one outlier at parameter $\alpha_{acc;1}$, the estimated parameters matched the true parameters very closely. The deviation seemed to increase slighlty with parameter magnitude. Considering accuracy, Figure 8 displays Cohen's kappa between true and estimated hidden state sequences. With two exceptions, kappa values were almost one, indicating nearly perfect agreement. In sum, the HMM with two states recovered parameters and hidden states very well and outliers only occured rarely.

Three States.

[3] The design and analysis plan of the simulation study have not been included in my proposal. However, since I preregistered both the design and analysis, I treated the results as confirmatory. In general, the evaluation of an algorithm can hardly be proposed in detail before the algorithm has been developed. Therefore, I chose to report a general results section. Nevertheless, I indicated whenever results were obtained through exploratory analyses that were not planned beforehand.

[4] Note that the intial state probability ρ_i has also been estimated. Since the HMM only simulated one state sequence, this parameter is always either zero or one (leading to an RMdSPD of one). Therefore, I decided not to include it in the analysis.

For the simulation with three states, the RMdSPD is shown in Figure 9. When response parameters (other than $a_{i=j}$) were manipulated, the RMdSPDs for a_{12} and a_{31} were consistently below 0.5. Varying κ in states two and three led to RMdSPDs below 0.5 in the respective states. Otherwise, RMdSPDs were consistently lower than 0.1, indicating a median deviation between true and estimated parameters below 10% of the true parameter. Inspecting the regressions between manipulated true and estimated parameters (see Figures 10 and 11) revealed strong and unbiased linear relationships (intercepts close to zero and slopes close to one). Again, the deviation seemed to increase with true parameter magnitude (reverse for kappas). In contrast to the two-state HMM, larger deviatons and more outliers were observed. Cohen's kappa values for the three-state HMM are presented in Figure 12. For most estimated models, the kappas between true and estimated state sequences were above 0.95, indicating almost perfect agreement. However, for some models, I observed kappas clustered around zero or -0.33, suggesting that state labels switched. To summarize, the three-state HMM had a slightly worse parameter recovery and more outliers than the two-state model but shows a similar good accuracy.

Four States.

The RMdSPDs for the four-state HMM is illustrated in Figure 13. For estimated transition probabilities and α_{vel} and β_{vel} parameters in states one and four, RMdSPDs were below 0.5. Estimated kappa parameters in smooth pursuits were also often below 0.5 when paramteres in states two, three, and four were varied. Otherwise, RMdSPDs were below 0.1, indicating a median deviation below 10% of the true parameter. Looking at the regressions between true and estimated paramters, Figures 14 and 15 show strong and unbiased relationships. However, there were larger deviations and more outliers than in the previous models, especially for states one and four. Accuracy measured by Cohen's kappa ranged between 0.6 and 0.9 for the majority of models, meaning moderate to almost perfect agreement between true and estimated state sequences (see Figure 16). Here, some kappa values clustered around 0.25 and zero, which again can be interpreted as the result of label switching. Summarizing, the parameter recovery for the four-state HMM was slightly worse with even more outliers compared to the three-state model. Especially the recovery of transition parameters in states corresponding to fixations and smooth pursuits decreased.

Overall, simulations in part one demonstrated that the HMM recovered parameters at least moderately well when parameters were manipulated (all RMdSPDs below 0.5). The HMM estimated state sequences very accurately. Adding states to the model decreased the accuracy and recovery slightly, leading to more outliers, especially when smooth pursuits were added.

Sample Size and Noise Variation.

Two States.

In the second part, I 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 17). The other estimated parameters showed RMdSPDs close to or below 0.1. Increasing the sample size seemed to improve RMdSPDs 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 18, left plot). To conclude, adding noise did not affect the parameter recover or accuracy of the two-state HMM, whereas increasing the sample size improved the recovery slightly.

Three States.

The RMdSPDs for the β_{vel} and β_{acc} were above 0.5 in all three states (see Figure 19). 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, higher sample sizes had lower RMdSPDs. The accuracy of the estimated models was almost perfect with most kappa values above 0.95 (see Figure 18, middle plot). Several outliers clustered around kappas of zero and -0.33, signaling label switching. For the three-state HMM, adding noise led to a slightly worse recovery and accuracy overall. However, scale parameters of gamma distributions in all three states were only badly recovered.

Four States.

RMdSPDs regarding the four-state HMM are displayed in Figure 20. For states one and four, values for most parameters (including all transition probabilites) were above 0.5. Similarly, RMdSPDs for β_{vel} and β_{acc} in states two and three were above 0.5. 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 18, right plot). Multiple kappa values clustered around 0.25 or zero, which can be explained by label switching. In summary, adding noise to data generated by the four-state HMM heavily decreased the recovery for most parameters in states corresponding to fixations and smooth pursuits. For saccade and PSO states, only scale parameters of gamma distributions were badly recovered, but increasing the sample size slightly improved the recovery.

In general, the HMM recovered parameters well despite noise being added to the data. However, when smooth pursuits were added to the model, the parameter recovery for fixation and smooth pursuit states substantially decreased. When PSOs or smooth pursuits were included, scale parameters of gamma distributions were badly recovered. Increasing the samples in the HMM slightly improved the recovery of most parameters. The accuracy of the model was slightly lowered when including more states, but it was neither affected by the noise variability τ_{noise} nor the sample size.

Variation of Starting Values. The third part of the simulation investigated how increasing the variation in generating starting values affected parameter recover and accuracy of the HMM. For the two-state HMM, all parameters displayed RMdSPDs lower than 0.1 (see Figure 21). Cohen's kappa values were slightly lower than 1, indicating almost perfect accuracy (see Figure 22). For the three-state HMM, RMdSPDs were lower than 0.1 except for a_{12} and a_{31}, which were below 0.5 (see Figure 23). As in previous parts, Cohen's kappa values were mostly above 0.95 (almost perfect accuracy) with exceptions clustering around zero and -0.33 (see Figure 22). Regarding the four state-HMM, RMdSPS of transition probabilities for states one and four were clustered above 0.1, other estimated parameters in the model showed values below 0.1 (see Figure 24). The HMM showed substantial to almost perfect accuracy, as Cohen's kappa values were mostly above 0.8 with a few values clustering around 0.6, 0.25, and zero (see Figure 22). In conclusion, the parameter recovery of the model worked very well. State sequences were accurately estimated and more states being included in the model slightly decreased the accuracy. Increasing the variation in starting values for parameter estimation did neither affect the parameter recovery nor accuracy.

In this part, I explored post-hoc whether clusters of low Cohen's kappa values were due to label switching. According to Visser and Speekenbrink (2019), label switching occurs when exchanging the order of the states leads to equally likely models. Thus, the model is accurate but has oppositely estimated states compared to the true states. In this case, an HMM with three states that is perfectly accurate but has one label swtiched would have a Cohen's kappa of 0. To test if label switching occured, I manually switched one or two labels post-hoc (see Figure 25). It can be seen that this approach resolved low accuracy clusters for three and four states.

Missing Data. In the last part, data intervals of varying length were set to be missing. Regarding parameter recovery, RMdSPDs were almost exactly mirroring those in the previous part for two, three, and four states (see Figures 26, 27, and 28, respectively). Looking at accuracies, Figure 29 illustrates an interaction between the length of and amount of missing data intervals. Cohen's kappa values linearly decreased with the length of missing data intervals and the decrease became linearly steeper when more intervals were included. For two and three state models, kappas started at almost one and decreased to 0.6 for five missing intervals, indicating substantial to almost perfect accuracy. For the four-state model, kappas started around 0.85 and decreased to 0.5 for five missing intervals, suggesting moderate to substantial accuracy. As in previous parts, a several kappa values clustered around zero and -0.33 for the three-state model and around 0.6, 0.25, and zero for the four-state model. In total, missing data did not affect the parameter recovery but linearly decreased the accuracy of the model from an nearly perfect to a moderate extent.

With a few exceptions, the simulation study revealed that the HMM recovered parameters well and accurately estimated the true state sequences. The most critical decrease in recovery occured when noise was added to the data generated by models including smooth pursuit. In the noise condition, scale parameters of gamma distributions were often badly recovered. Higher sample sizes slightly improved parameter recovery, but neither variation in starting values nor missing data affected it. In contrast, the accuracy of the model was linearly descreasing with more data missing but not influenced by manipulating parameters, noise, or starting value variation. Adding more states to the HMM generally decreased the parameter recovery and the accuracy.

Validation

Starting Values and Model Fitting. I applied gazeHMM on two benchmark data sets to test my hypotheses. Note that, for the Ehinger et al. (2019) data set, I only fitted models to the Eyelink data to reduce the complexity of the analyses. For both data sets, I used the same set of starting values (initial state model: $\rho = 1/k$; transition model: $a_{i=j} = 0.9$ and $a_{i \neq j} = 0.1/k$; response model: see Table 4) to estimate the parameters of the HMM. In contrast to the simulation study, generating random starting value 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 (so were the starting values for their gamma distributions). The algorithm was applied separately for every participant and every condition, task, or block. For the data set by Andersson et al. (2017), the algorithm was successfully fitted for every participant and condition. An example for event classification by gazeHMM is displayed in Figure

@(fig:plot-example-classification) and examples for parameter estimates are presented in Tables A1 and A2.

Model Comparison. To examine whether HMMs are accurately describing eye movements, I applied gazeHMM with one, two, three, four, and five states on two benchmark data sets and compared the Schwarz weights of the HMMs. Schwarz weights are the result of transforming a set of BIC values. They can be interpreted as the probability for a model to have generated the data it was fitted to. For the Andersson et al. (2017) data set, I expected the highest weights for the three-state models in the image condition. For the moving dots and video conditions, I predicted the highest weights for the four-state model. Figure 30 shows the Schwarz weights for different subjects and models in the image condition. For all subjects, the five-state model displays the highest weight, suggesting that it hast most likely generated the data. In the moving dots condition, the five-state model has the highest weights for most subjects, but the one-, three, and four-state models exhibit the highest weights for two subjects each (see Figure 31). The video condition has the same pattern as the image condition, as the five-state model consistently has the highest weights (see Figure 32). In contrast to my hypothesis, applying different HMMs to the Andersson et al. (2017) data suggests that the five-state model has most likely generated the data. Except for the moving dots condition, there was little variation in the Schwarz weights, indicating large differences in the model likelihoods. The model comparison for the Ehinger et al. (2019) data set yielded similar results (see Figures A2 and A3). 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 in choosing the correct data generating model. Therefore, I computed the weighted ICL criterion for the HMMs 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 Figures A4, A5, and A6). The preference for the five-state model was even more consistent across conditions and subjects.

Comparison to Other Algorithms.

Event Durations.

The hypothesis that HMMs improve the classification performance of eye movement events was investigated by comparing gazeHMM against other algorithms. Therefore, I applied gazeHMM with four events to the Andersson et al. (2017) data set. Therefore, I calculated the RMSD[5] of event durations as described in the original article by Andersson et al. for all algorithms included in the study plus gazeHMM: First, the mean and standard deviation of event durations as well as the number of events were normalized to [0,1] for each coder and algorithm. Second, the sum of the root square deviation between each algorithm and the average of the two human coders is computed. This metric can only be relatively interpreted (Andersson et al., 2017).

Table 5 shows that, in all three conditions, gazeHMM has a higher RMSD than all the the other algorithms for classified fixations. The high RMSD values were mostly due to a large amount of very short fixations.

For saccades, gazeHMM has a relatively low RMSD of the algorithms in the image and video condition and a relatively high RMSD in the moving dots condition (see Table 6). Here, gazeHMM overestimated the duration of saccades, as indicated by high mean and SD durations.

For PSOs, gazeHMM had the highest RMSD in the image and video condition (see Table 7). In the moving dots condition, its RMSD was lower than for NH (Nyström & Holmqvist, 2010) but higher than for LNS (Larsson et al., 2013). Here, gazeHMM underestimated the number of PSOs and their duration compared to the human coders.

No other algorithm in the study was designed to classify smooth pursuits, but

[5] In my proposal, I declared to compute the RMSE instead of RMSD. However, since I conducted an in sample comparison, the term deviation is more appropriate. Andersson et al. (2017) also referred to it as RMSD.

gazeHMM classified a large number of short smooth pursuits, causing a substantially higher RMSD than the human coders (see Figure 8).

In sum, gazeHMM did not outperform all other algorithms regarding the RMSD of event durations compared to human coders. The duration of fixations and smooth pursuits was largely underestimated and the number of these two events largely overestimated. For saccades and PSOs, gazeHMM estimated relatively accurate durations and event numbers, but didn't reach the best algorithms.

Sample-to-sample Agreement.

Moreover, 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 9). Cohen's kappa indicates the agreement between to classifiers accounting for the agreement due to chance. Absolute 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 were relatively high compared to the other algorithms. However, 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 mediocre in the image and video conditions compared to NH and LNS.

The agreement for smooth puruits was moderate in the moving dots condition but slight in the image and video conditions. 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 mediocre agreement. The performance for fixations was absolutely and relatively low.

Disagreement and Confusion.

At last, I computed the overall disagreement[6] as the percentage of all samples taken together that were classified differently by gazeHMM than the human coders. Figure 33 shows that in the image condition, gazeHMM with four events had a higher overall disagreement than all but two of the other algorithms. In the moving dots and video conditions, the disagreement was considerably lower than for all the 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.

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 mediocre 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 10, it can be seen that gazeHMM confused fixations and smooth pursuit to a large extend. In the image condition, it overclassified smooth pursuit events, while they were underclassified in the dynamic conditions.

gazeHMM With Three States. Even though it was not confirmed by the model comparison, applying gazeHMM with three states to static data seems theoretically more appropriate. Including only three states could prevent gazeHMM from overclassifying smooth pursuits in the image data and yield better validation results. Therefore, I decided

[6] 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.

to apply gazeHMM with three states to the image dat and repeat the comparison to other algorithms exploratorily.

Table 11 shows the RMSD between gazeHMM and human coders only for the image condition. For fixations, gazeHMM shows a comparably low RMSD but it was relatively high for saccades and PSOs. The sample-to-sample agreement between gazeHMM with three states and the human coders measured by Cohen's kappa is displayed in Table 12 (only for the image condition). For fixations, the absolute kappa indicated substantial agreement and was relatively high compared to the other algorithms. The absolute kappa for saccades corresponded to moderate agreement. Compared to the other algorithms, it was rather low. For PSOs, the absolute agreement was fair, but relatively low compared to the other algorithms.

Figure 33 illustrates the overall disagreement between gazeHMM with three events and the human coders compared to the other algorithms. In all three conditions, the disagreement was slightly lower for gazeHMM than for all the other algorithms. Compared to gazeHMM with four states, the RMSD decreased while the sample-to-sample agreement increased for fixations. For saccades and PSOs, the opposite pattern was found. The overall disagreement substantially decreased. Overall, applying gazeHMM with three states to static data seems more appropriate, since the increase in fixation classification outweighs the decrease in classification of saccades and PSOs. There was even some support for my second hypothesis since gazeHMM had lower overall disagreement to human coders than all other algorithms.

Discussion

In this report, I presented gazeHMM, a novel algorithm for classifying gaze data into eye movement events. The algorithm models velocity, acceleration, and sample-to-sample angle signals with gamma distributions and a mixture of von Mises and a uniform distribution. An HMM classifies the gaze samples into fixations, saccades, and optionally PSOs, and/or smooth pursuits. A simulation study revealed that the generative model of gazeHMM recovered parameters and hidden state sequence well with a few exceptions. Adding smooth pursuits to the model and noise to the generated data were the most critical factors for decreasing recovery. In contrast, higher sample sizes yielded a better parameter recovery. The variation in starting values did not affect the recovery but the amount of missing data decreased the recovery of hidden states. I applied gazeHMM with different numbers of states to benchmark data and compared the model fit, showing that a five-state HMM had consistently most likely generated the data. Thus, the model comparison provided evidence against my first hypothesis that the events in gazeHMM will match theoretical expectations.

When comparing gazeHMM with four events against other algorithms and human coders, it did not show the best performance for the duration and number of classified events nor the sample-to-sample agreement across all conditions. However, it had a lower overall disagreement for dynamic data but higher disagreement for static data. I argued that four states might not be appropriate for static data and additionally applied gazeHMM with three states to them. That led to substantially better classifications for fixations, but slightly worse performance for saccades and PSOs. With three states, gazeHMM had a lower overall disagreement than all other algorithms. I treat this pattern as moderate evidence for my second hypothesis that gazeHMM improves the classification of eye movements. Taken together, I only found some evidence for one part of my research question, that gazeHMM improves classifying gaze data but not describing them.

Considering the results of the simulation study, it seems reasonable that adding the smooth pursuit state to the HMM decreased parameter and state recovery: It is the event that is overlapping most closely with another event (fixations) in terms of velocity, acceleration, and sample-to-sample angle. The overlap can cause the HMM to confuse parameters and hidden states. The decrease in parameter recovery (especially for scale parameters) due to noise shows that the overlap is enhanced by more dispersion in the data. The scale parameters might be particularly vulnerable to extreme data points. The decrease in state recovery for long and many intervals of missing data can be explained by the tendency of HMMs to stay in the same state when no information is available from the data. However, the decrease was rather small, leading to the conclusion that the HMM can deal well with missing data. Despite these drawbacks, the recovery of the generative model in gazeHMM seems very promising. The simulation study gives also an approximate reference for the maximum recovery of hidden states that can be achieved by the HMM (Cohen's kappa values of ~1 for two, ~0.95 for three, and ~0.8 for four events).

The model comparison on the benchmark data clearly provided evidence against HMMs being useful generative models for eye movement data. Even using a model selection criterion that identified correct models succesfully in previous model comparison studies (Pohle et al., 2017), did not yield the expected pattern. There are several explanations for this result:

There were more eye movement events present in the data than I expected. Eye movement events can be divided into subevents. For example, fixations consist of drift and tremor movements (Duchowski, 2017) and PSOs encompass dynamic, static, and glissadic over- and undershoots (Larsson et al., 2013). A study on a recently developed HMM algorithm supports this explanation: Houpt et al. (2018) applied the unsupervised BP-AR-HMM algorithm to the Andersson et al. (2017) data set and classified more distinct states than the human coders. Some of the states classified by BP-AR-HMM matched the same event coded by humans. Since the subevents are usually not interesting for users of classification algorithms, the ability of HMMs to classify might limit their ability to describe eye movements here.

Model selection criteria are generally not appropriate for comparing HMMs with differnt numbers of states. This argument has been discussed in the field of ecology (see Li & Bolker, 2017), where studies found that selection criteria preferred models with more states than expected (similar to the result of this study; e.g., Langrock, Kneib, Sohn, & Deruiter, 2015). Li and Bolker (2017) explain this bias with the simplicity of the submodels in HMMs: Initial state, transition, and response models for each state are usually relatively simple. When they do not describe the processes in the respective states accurately, the selection criteria compensate for that by preferring a model with more states. Thus, there are not more latent states present in the data, but the submodels of the HMM are misspecified or too simple. Correcting for model misspecifications, led to better model recovery in studies on animal movements (Langrock et al., 2015; Li & Bolker, 2017). However, Pohle et al. (2017) showed in simulations that the ICL identified the correct model despite several misspecifications. It has to be noted that the study by Pohle et al. (2017) only used data generating models with two states.

The submodels of gazeHMM were misspecified. Pohle et al. (2017) identified two scenarios in which model recovery using the ICL did not give optimal results: Outliers in the data and inadequate distributions in the response models. Both situations could apply to gazeHMM and eye movement data. Outliers occur frequently in eye-tracking data due to measurement error. Choosing adequate response distributions in HMMs is usually difficult and can depend on the individual and task the data is obtained from (Langrock et al., 2015). Moreover, gazeHMM only estimated intercepts for all parameters and thus, no time-varying covariates were included (cf. Li & Bolker, 2017). This aspect could indeed oversimplify the complex nature of eye movement data.

Comparing gazeHMM to other algorithms on benchmark data showed that gazeHMM moderately improved the event classification in agreement with human coders. However, the evaluation criteria (RMSD of event durations, sample-to-sample agreement, and overall disagreement) yieled different results. The fact that gazeHMM outperformed all other algorithms regarding the overall disagreement can be because it is the only algorithm classifying five events. Of course, that makes it still a useful tool for classification. Nevertheless, Cohen's kappa values of 0.72 (saccades - video - four states) or 0.68 (fixations - image - three states) indicate substantial agreement to human coders, especially in light of the maximum reference of ~0.8 from the simulation study. At this point it is important to mention that human coding should not be considered a gold standard in event classification: Hooge et al. (2018) observed substantial differences between coders and within coders over time. Despite these differences, they recommend comparisons with human coding to demonstrate the performance of new algorithms and finding errors in their design.

Advantages of gazeHMM

In view of the four proposed goals that gazeHMM should fulfill, I can draw the following conclusions: Even though gazeHMM does require some parameter settings (in the pre- and postprocessing), they only have minor influence on the classification and are merely included to compensate for some drawbacks of the generative model. Their default values should be appropriate for most applications. A major advantage of gazeHMM is that it does not require human labeled data as input. Instead, it estimates all parameters and hidden states from the data. Since human coding is quite laborious, difficult to reproduce, and by times inconsistent (as noted earlier, Hooge et al., 2018), this property makes gazeHMM a good alternative to other recently developed algorithms that require human coded input (Bellet et al., 2019; Pekkanen & Lappi, 2017; Zemblys et al., 2018). This could also explain why the agreement to human coding is lower for gazeHMM than for algorithms that learn from human labeled data. Another advantage of gazeHMM is its ability to classify four eye movement events, namely fixations, saccades, PSOs, and smooth pursuit. Whereas most algorithms only parse fixations and saccades (Andersson et al., 2017), few are able to classify PSOs (e.g., Zemblys et al., 2018), and even less categorize smooth pursuits (e.g., Pekkanen & Lappi, 2017). However, including smooth pursuits in gazeHMM led to some undesirable classifications on benchmark data, resulting in rapid switching between fixation and smooth pursuit events. Therefore, I recommend using gazeHMM with four events only for exploratory purposes. Without smooth pursuits, I consider gazeHMM's classification as appropriate for application. Lastly, its implementation in R using depmixS4 (Visser & Speekenbrink, 2010) should make gazeHMM a tool that is easy to use and customize for individual needs.

Future Directions

Despite its advantages, there are several aspects in which gazeHMM can be improved: First, a multivariate distribution could be used to account for the correlation between velocity and acceleration signals (for an example, see Balakrishnan & Lai, 2009). Potential problem of this approach might be choosing the right distribution and convergence issues (due to a large number of parameters). Another option to model the correlation could be to include one of the response variables as a covariate of the other.

Second, instead of the gamma being the generic (and potentially inappropriate) response distribution, a non-parametric approach could be used: Langrock et al. (2015) use a linear combination of standardized B-splines to approximate response densities, which led to HMMs with less states being preferred. This approach could potentially combat the problem of unexpectedly high-state HMMs being preferred for eye movement data.

Third, one solution to diverging results when comparing gazeHMM with different events could be model averaging: Instead of using the maximum posterior state probability of each sample from the preferred model, the probabilities could be weighted according to a model selection criterion (e.g., Schwarz weight) and averaged. Then, the maximum averaged probability could be used to classify the samples into events. This approach could lead to a more robust classification because averaging might reduce the inaccuracy of each competing model. However, the model comparison for gazeHMM often showed extreme weights for a five-state model, which would lead to a very limited influence of the other models in the averaged probabilities. Fourth, including covariates of the transition probabilities and response parameters could improve the fit of gazeHMM on eye movement data. As pointed out earlier, just estimating intercepts of parameters could be too simple to model the complexity of eye movements. As candidate covariates I suggest periodic functions of time (Li & Bolker, 2017) which could, for instance, capture the specific pattern of saccades. Whether covariates are improving the fit of submodels to eye movement data could in turn be assessed by inspecting pseudo-residuals and auto-correlation functions (see Zucchini, MacDonald, & Langrock, 2016). Fifth, to avoid rapid switching between fixations and smooth pursuits as well as unreasonably short saccades, gazeHMM could explicitly model the duration of events. This can be achieved by setting the diagonal transition probabilities to zero and assign a distribution of state durations to each state (Bishop, 2006). Consequently, the duration distributions of fixations and smooth pursuits could differ from saccades and PSOs. This extension of the HMM is also called hidden semi-Markov model and has been successfully used by Mihali et al. (2017) to classify microsaccades. A drawback of the extension would be higher computational costs.

Lastly, allowing constrained parameters in the HMM could replace some of the postprocessing steps in gazeHMM. This could potentially be achieved by using different response distributions or parameter optimization methods. Moreover, switching from the maximum likelihood to the Markov chain Monte Carlo (Bayesian) framework could help avoiding convergence problems with constrained parameters, but would also open new research questions about suitable priors for HMM parameters in the eye movement domain.

Conclusion

Both the simulation and validation studies showed that gazeHMM is a suitable algorithm for classifying eye movement events. For smooth pursuits, the classification is not optimal and thus not recommended, yet. On one hand, the algorithm has some advantages over concurrent event classfication algorithms, not relying on human-labeled input being the most important one. On the other hand, it is not able to perform model comparisons as expected and still poses difficulties regarding good model specifications for different eye movement events. These difficulties highlight general challenges for HMMs in general, but also specifically for eye movements. Revisting my research question, I conclude that HMMs are suitable models for classifying eye movement events but still lack the ability to accurately describe them.

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

Balakrishnan, N., & Lai, C.-D. (2009). *Continuous bivariate distributions* (2nd ed., Vol. 53). New York: Springer. https://doi.org/10.1007/b101765

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

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

Bishop, C. M. (2006). *Pattern recognition and machine learning.* New York: Springer.

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

Cappé, O., Moulines, E., & Ryden, T. (2005). *Inference in hidden Markov models.* New York: Springer.

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

Hessels, R. S., Niehorster, D. C., Kemner, C., & Hooge, I. T. C. (2017). Noise-robust fixation detection in eye movement data: Identification by two-means clustering (I2MC). *Behavior Research Methods*, *49*, 1802–1823. https://doi.org/10.3758/s13428-016-0822-1

Hooge, I. T. C., Niehorster, D. C., Nyström, M., Andersson, R., & Hessels, R. S. (2018). Is human classification by experienced untrained observers a gold standard in fixation detection? *Behavior Research Methods*, *50*, 1864–1881. https://doi.org/10.3758/s13428-017-0955-x

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.

Langrock, R., Kneib, T., Sohn, A., & Deruiter, S. L. (2015). Nonparametric inference in hidden Markov models using P-splines. *Biometrics*, *71*(2), 520–528. https://doi.org/10.1111/biom.12282

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.

Li, M., & Bolker, B. M. (2017). Incorporating periodic variability in hidden Markov models for animal movement. *Movement Ecology*, *5*(1), 1–12. https://doi.org/10.1186/s40462-016-0093-6

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* (PhD 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

Spakov, O. (2012). Comparison of eye movement filters used in HCI. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 281–284. https://doi.org/10.1145/2168556.2168616

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., & 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*. Berlin: Springer.

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

Table 1

Approximate ranges of response variables used to generate parameter values

Event	Resp.variable	Range
Fixation	Velocity	0-50
Fixation	Acceleration	0-50
Fixation	Angle	uniform
Saccade	Velocity	50-1000
Saccade	Acceleration	50-500
Saccade	Angle	~0
PSO	Velocity	20-100
PSO	Acceleration	10-90
PSO	Angle	~pi
Smooth pursuit	Velocity	20-100
Smooth pursuit	Acceleration	0-30
Smooth pursuit	Angle	~0

Note. Units are deg/s (velocity), deg/s^2 (acceleration), and radians (angle). ~ indicates that the distribution has a peak at this value. Velocity ranges are based on event velocities reported in Larsson et al. (2013). Since I would also like to test the algorithm on extreme data distributions, I extended the ranges beyond those found in typical eye movement data.

Table 2

Intervals and default parameter values for the response models of the HMM 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*.

Table 3

Intervals and default parameter values for the response models of the HMM 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 $= 2\pi$.

Table 4

Starting values for response models in fitting gazeHMM to benchmark data sets

	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.

Table 5

Mean and standard deviations of fixation durations, number of fixations, and RMSD between fixation durations classified by algorithms and human coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	0.275	0.285	403	0.02	0.19	0.09	12	0.089	0.338	0.303	82	0.305
coderRA	0.271	0.287	391	0.02	0.167	0.092	21	0.089	0.255	0.185	81	0.305
gazeHMM	0.024	0.032	2457	2.076	0.017	0.018	455	1.657	0.023	0.026	1124	1.757
CDT	0.397	0.559	251	0.914	0.06	0.127	165	0.773	0.213	0.297	211	0.344
EM	-	-	-	-	-	-	-	-	-	-	-	-
IDT	0.399	0.328	242	0.485	0.323	0.146	8	0.592	0.554	0.454	48	0.779
IKF	0.174	0.239	513	0.406	0.217	0.184	72	0.526	0.258	0.296	169	0.219
IMST	0.304	0.293	333	0.124	0.268	0.14	12	0.412	0.526	0.825	71	1.169
IHMM	0.133	0.216	701	0.644	0.214	0.286	67	0.83	0.234	0.319	194	0.316
IVT	0.114	0.204	827	0.774	0.203	0.282	71	0.796	0.202	0.306	227	0.391
NH	0.258	0.299	292	0.112	0.38	0.333	30	1.355	0.429	0.336	83	0.366
BIT	0.209	0.136	423	0.467	0.189	0.113	67	0.212	0.248	0.215	170	0.211
LNS	-	-	-	-	-	-	-	-	-	-	-	-

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

Table 6

Mean and standard deviations of saccade durations, number of saccades, and RMSD between saccade durations classified by algorithms and human coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	0.032	0.016	377	0.1	0.023	0.01	47	0.054	0.027	0.012	117	0.071
coderRA	0.034	0.014	374	0.1	0.022	0.011	47	0.054	0.027	0.011	127	0.071
gazeHMM	0.04	0.024	388	0.501	0.035	0.026	59	1.151	0.039	0.021	122	0.547
CDT	-	-	-	-	-	-	-	-	-	-	-	-
EM	0.025	0.022	787	1.209	0.017	0.014	93	0.837	0.02	0.016	252	0.906
IDT	0.025	0.015	258	0.408	0.032	0.014	10	0.803	0.024	0.053	41	1.426
IKF	0.062	0.037	353	1.494	0.06	0.026	29	1.736	0.055	0.02	107	1.025
IMST	0.017	0.01	335	0.625	0.013	0.005	18	0.822	0.018	0.01	76	0.496
IHMM	0.048	0.026	368	0.747	0.041	0.017	27	0.931	0.042	0.018	109	0.618
IVT	0.041	0.022	373	0.434	0.036	0.014	28	0.671	0.036	0.016	112	0.395
NH	0.05	0.02	344	0.615	0.043	0.016	42	0.745	0.044	0.018	104	0.696
BIT	-	-	-	-	-	-	-	-	-	-	-	-
LNS	0.029	0.012	390	0.235	0.026	0.011	53	0.16	0.028	0.012	122	0.039

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

Table 7

Mean and standard deviations of PSO durations, number of PSOs, and RMSD between PSO durations classified by algorithms and human coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	0.025	0.014	313	0.229	0.015	0.005	33	0.786	0.023	0.013	97	0.744
coderRA	0.025	0.012	310	0.229	0.015	0.008	28	0.786	0.021	0.012	89	0.744
gazeHMM	0.02	0.02	213	2.208	0.004	0.006	23	1.193	0.025	0.019	66	1.944
NH	0.028	0.013	237	1.079	0.024	0.012	17	2.068	0.028	0.013	78	1.333
LNS	0.025	0.009	319	0.457	0.02	0.009	31	0.595	0.024	0.01	87	0.755

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

Table 8

Mean and standard deviations of smooth pursuit durations, number of smooth pursuits, and RMSD between smooth pursuit durations classified by algorithms and human coders

	Image				Moving dots				Video			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
coderMN	0.363	0.187	3	0.235	0.363	0.236	48	0.329	0.559	0.391	51	0.135
coderRA	0.299	0.18	17	0.235	0.367	0.333	45	0.329	0.516	0.376	70	0.135
gazeHMM	0.02	0.022	2648	2.882	0.022	0.022	493	2.835	0.025	0.026	1178	2.933

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

Table 9

Cohen's kappa between human coders and algorithms for different conditions and events

	Fixations			Saccades			PSOs			Smooth pursuit		
Algorithm	Image	Dots	Video	Image	Dots	Video	Image	Dots	Video	Image	Dots	Video
coderMN	0.92	0.81	0.83	0.95	0.91	0.94	0.88	0.82	0.83	0.00	0.00	0.00
coderRA	0.92	0.84	0.82	0.95	0.91	0.94	0.88	0.80	0.81	0.00	0.00	0.00
gazeHMM	0.28	0.24	0.18	0.68	0.58	0.72	0.33	0.14	0.40	0.01	0.56	0.21
CDT	0.38	0.06	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
EM	0.00	0.00	0.00	0.64	0.66	0.67	0.00	0.00	0.00	0.00	0.00	0.00
IDT	0.36	0.00	0.03	0.45	0.26	0.38	0.00	0.00	0.00	0.00	0.00	0.00
IKF	0.63	0.03	0.14	0.58	0.46	0.59	0.00	0.00	0.00	0.00	0.00	0.00
IMST	0.38	0.00	0.03	0.54	0.30	0.52	0.00	0.00	0.00	0.00	0.00	0.00
IHMM	0.67	0.03	0.13	0.69	0.60	0.71	0.00	0.00	0.00	0.00	0.00	0.00
IVT	0.67	0.03	0.13	0.75	0.63	0.76	0.00	0.00	0.00	0.00	0.00	0.00
NH	0.52	0.00	0.01	0.67	0.60	0.68	0.24	0.20	0.25	0.00	0.00	0.00
BIT	0.67	0.03	0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LNS	0.00	0.00	0.00	0.81	0.75	0.81	0.64	0.59	0.63	0.00	0.00	0.00

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

Table 10

Confusion matrix between gazeHMM (rows) and human coders (columns) for different events and conditions

Event	Fixations	Saccades	PSOs	Pursuits	Blinks	Other
Image						
Fixations	51947	523	402	1400	324	2
Saccades	1107	8909	2758	179	877	94
PSOs	1055	562	1845	45	203	10
Pursuits	44353	1033	1560	1422	606	14
Blinks	632	260	79	41	5407	51
Moving dots						
Fixations	1718	7	5	6082	11	13
Saccades	34	946	314	743	36	21
PSOs	15	6	40	31	0	0
Pursuits	850	88	87	9843	31	9
Blinks	292	23	12	282	8776	8841
Video						
Fixations	11544	34	34	12448	1	0
Saccades	244	2727	810	367	149	19
PSOs	235	278	642	135	34	1
Pursuits	9795	51	261	17334	5	0
Blinks	85	8	0	22	810	2

Note. gazeHMM classified four events and blinks.

Table 11

Mean and standard deviation of event durations, number of events, and RMSD between event durations classified by algorithms and human coders for the image condition

	Fixations				Saccades				PSOs			
Algorithm	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD	Mean	SD	Events	RMSD
MN	0.28	0.29	403	0.04	0.03	0.02	377	0.1	0.03	0.01	313	0.24
RA	0.27	0.29	391	0.04	0.03	0.01	374	0.1	0.03	0.01	310	0.24
gazeHMM	0.2	0.28	503	0.45	0.05	0.04	511	1.45	0.02	0.02	280	1.6
CDT	0.4	0.56	251	1.33	-	-	-	-	-	-	-	-
EM	-	-	-	-	0.02	0.02	787	1.19	-	-	-	-
IDT	0.4	0.33	242	0.81	0.02	0.02	258	0.41	-	-	-	-
IKF	0.17	0.24	513	0.66	0.06	0.04	353	1.43	-	-	-	-
IMST	0.3	0.29	333	0.23	0.02	0.01	335	0.61	-	-	-	-
IHMM	0.13	0.22	701	1.18	0.05	0.03	368	0.72	-	-	-	-
IVT	0.11	0.2	827	1.49	0.04	0.02	373	0.41	-	-	-	-
NH	0.26	0.3	292	0.26	0.05	0.02	344	0.6	0.03	0.01	237	1.35
BIT	0.21	0.14	423	0.62	-	-	-	-	-	-	-	-
LNS	-	-	-	-	0.03	0.01	390	0.23	0.02	0.01	319	0.48

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

Table 12

Cohen's kappa between human coders and algorithms for different events in the image condition

Algorithm	Fixations	Saccades	PSOs
coderMN	0.92	0.95	0.88
coderRA	0.92	0.95	0.88
gazeHMM-4	0.28	0.68	0.33
gazeHMM-3	0.67	0.50	0.26
CDT	0.38	0.00	0.00
EM	0.00	0.64	0.00
IDT	0.36	0.45	0.00
IKF	0.63	0.58	0.00
IMST	0.38	0.54	0.00
IHMM	0.67	0.69	0.00
IVT	0.67	0.75	0.00
NH	0.52	0.67	0.24
BIT	0.67	0.00	0.00
LNS	0.00	0.81	0.64

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

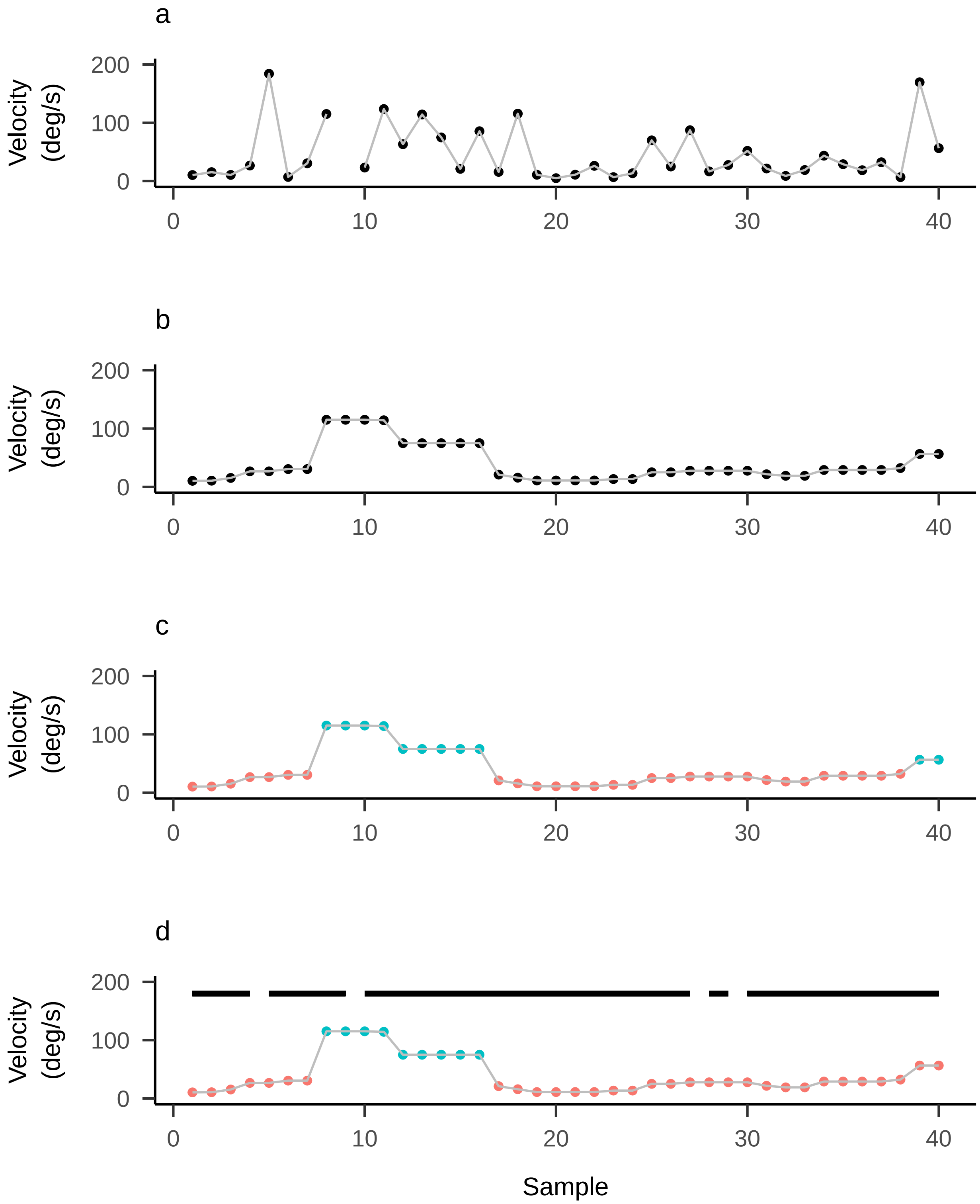

Figure 1. Example workflow for eye movement event classification algorithms: (a) the raw gaze signal is transformed into a velocity signal (in deg/s); (b) the velocity signal is smoothed and filtered; (c) samples are labeled as events (indicated by colors), and (d) relabeled. Sequences of samples belonging to the same event are merged (indicated by black segments).

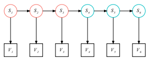

Figure 2. Example structure of HMMs: The observed velocity signal V depends on hidden states S (displayed for six samples). Each state depends on the previous state. Colors indicate different states. Figure inspired by Visser and Speekenbrink (2019, p. 133).

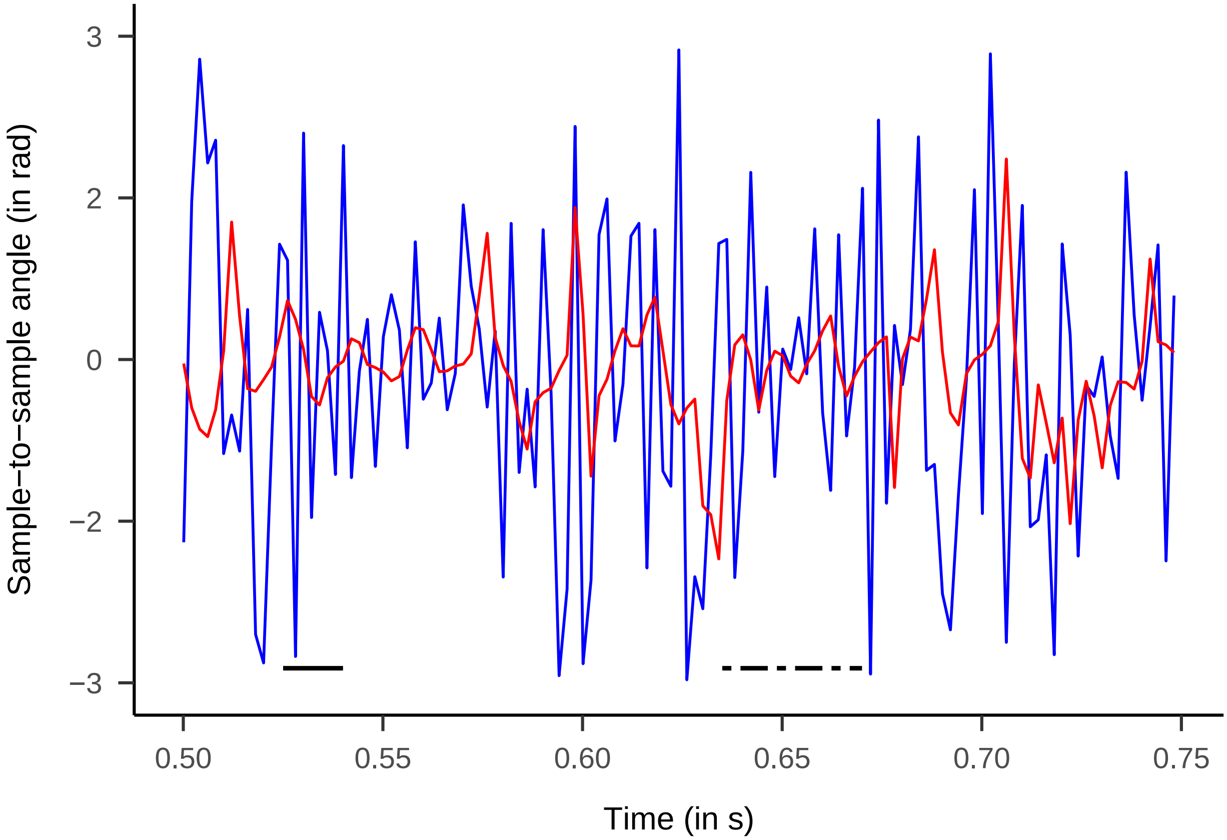

Figure 3. Example data from Andersson et al. (2017) displayed as sample-to-sample angle (in rad) over time (in s). Line colors indicate whether the positional data has been filtered before computing the sample-to-sample angle (red) or not (blue). For this example, I used a Butterworth filter of order three and a normalized cutoff frequency of 0.3. The full line segment marks where potential oscillations are filtered out. The dashed line segment illustrates where potentially random change in angle is filtered out.

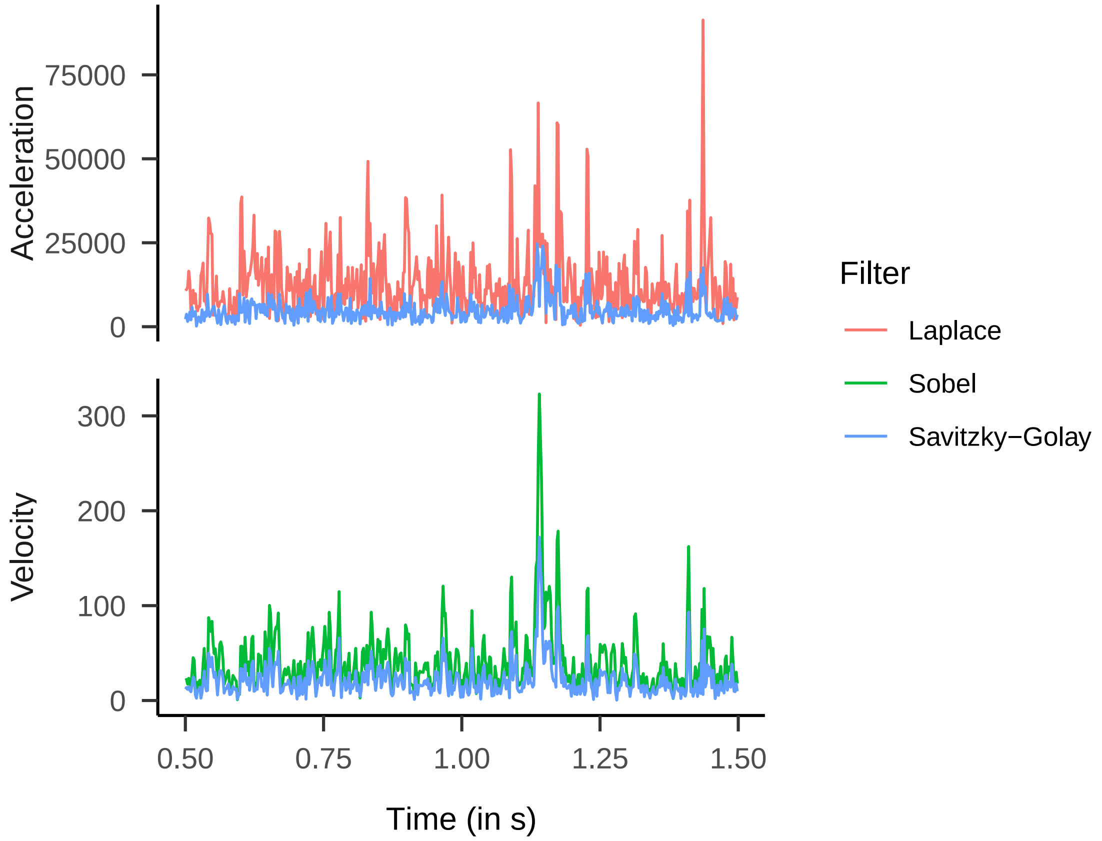

Figure 4. Example data from Andersson et al. (2017) displayed as velocity (deg/s) and acceleration (in deg/s^2) over time (in s). Line colors indicate which filter has been used to derive the signal. Savitzky-Golay filters were applied with order three and length five (corresponding to 10 ms).

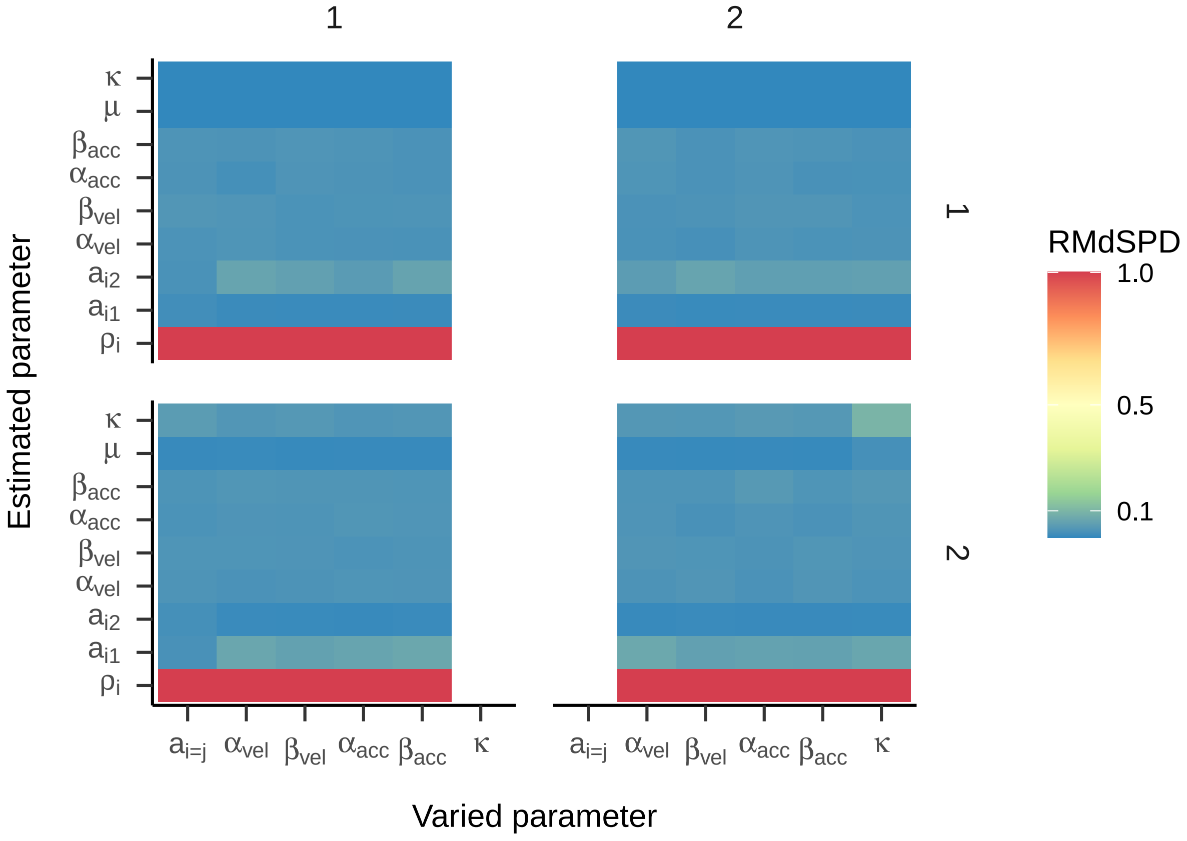

Figure 5. RMdSPD between true and estimated parameters of the two-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which state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

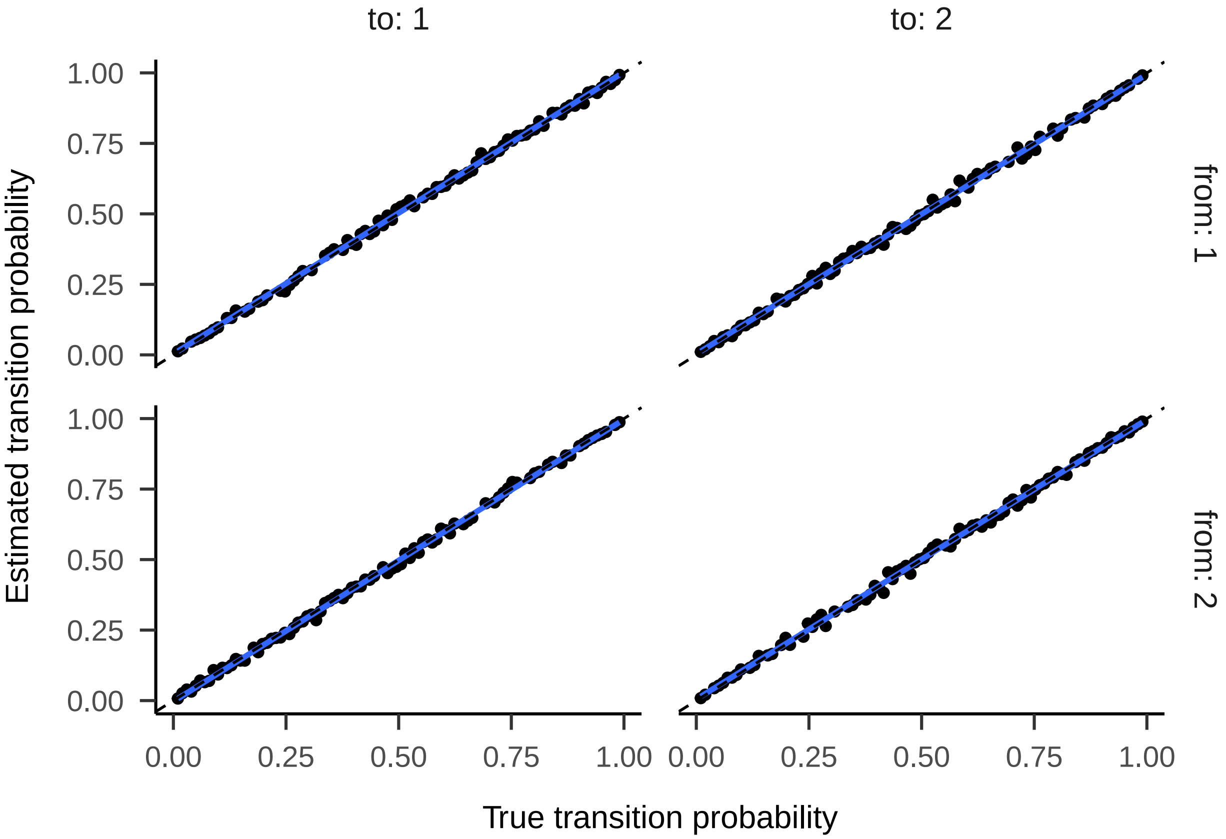

Figure 6. Regression lines between true and estimated transition probabilities for the two-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving.

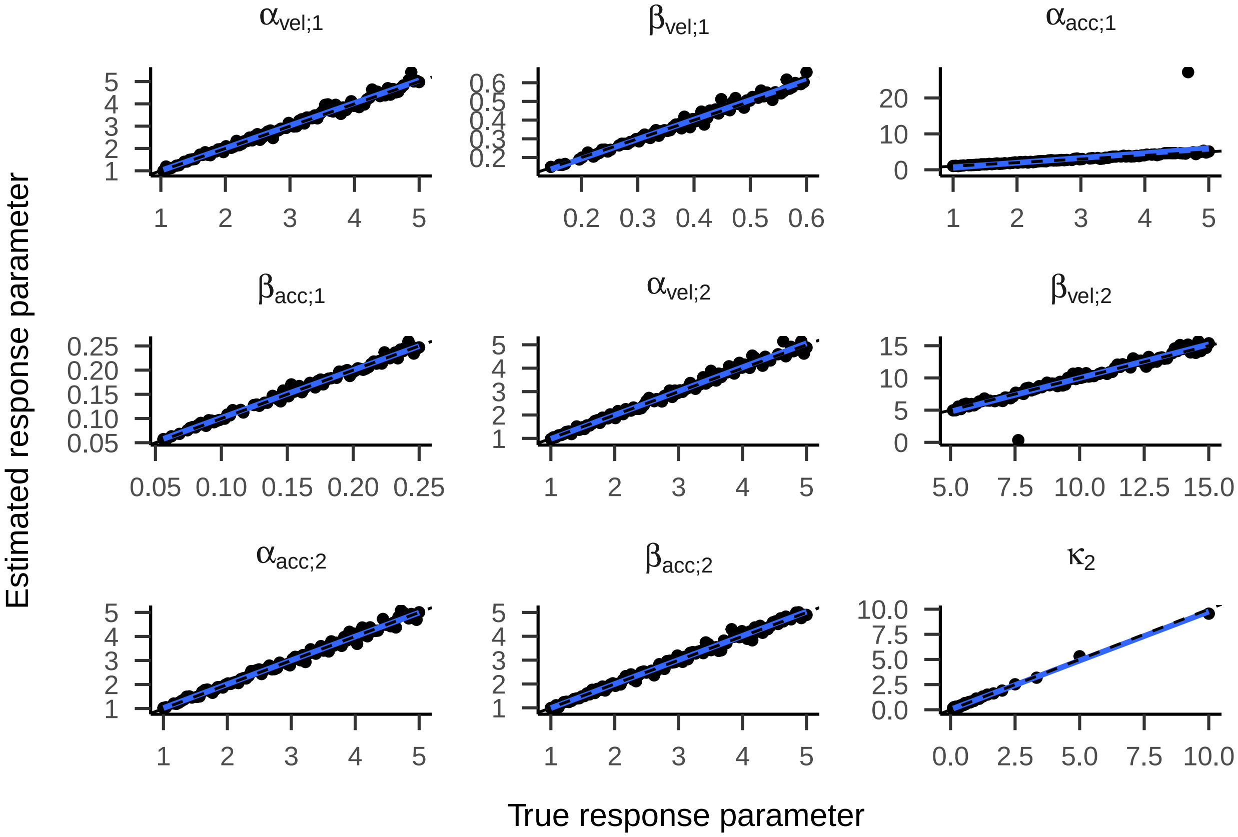

Figure 7. Regression lines between true and estimated response parameters of the two-state HMM in part one. Top facet labels indicate response parameters.

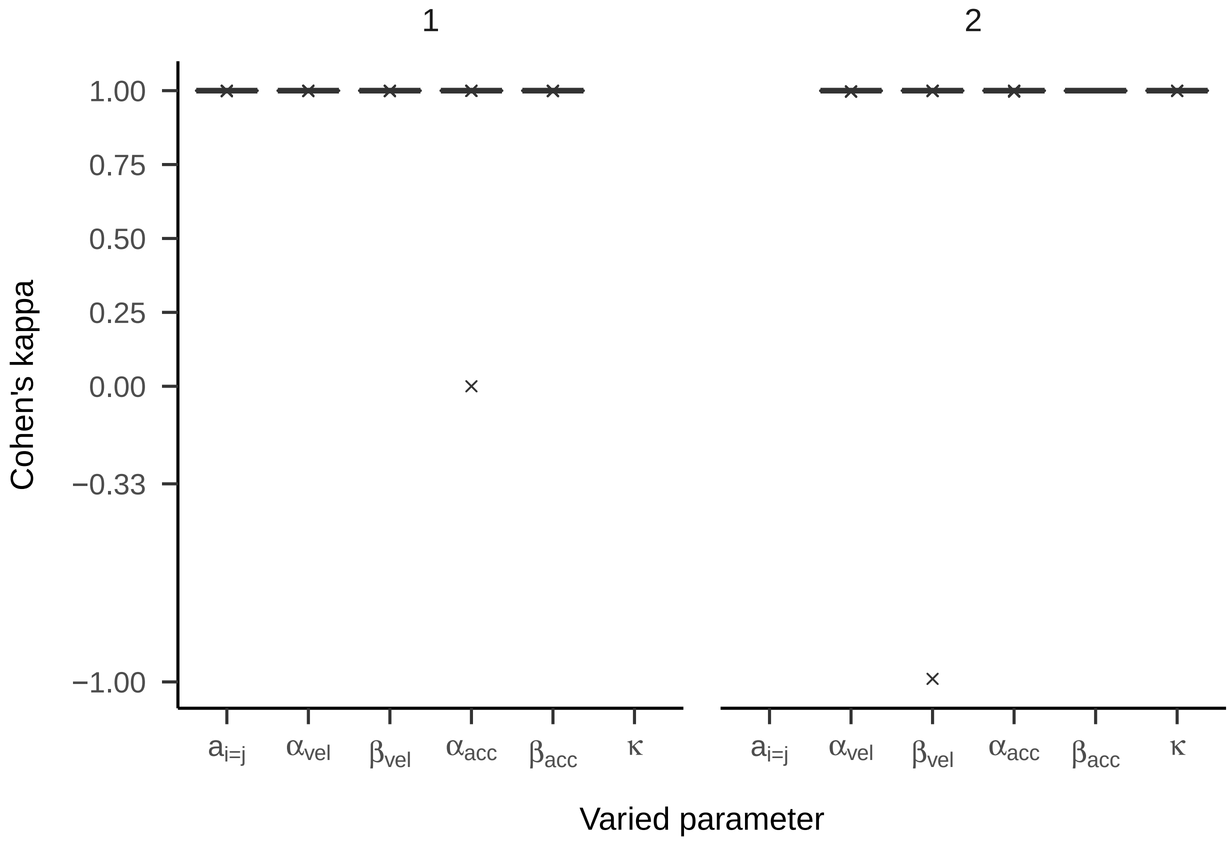

Figure 8. Boxplots displaying Cohen's kappa depending on which parameter of the two-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians. Crosses represent outliers (distance to first/third quartile higher than 1.5 times the inter-quartile range [IQR]).

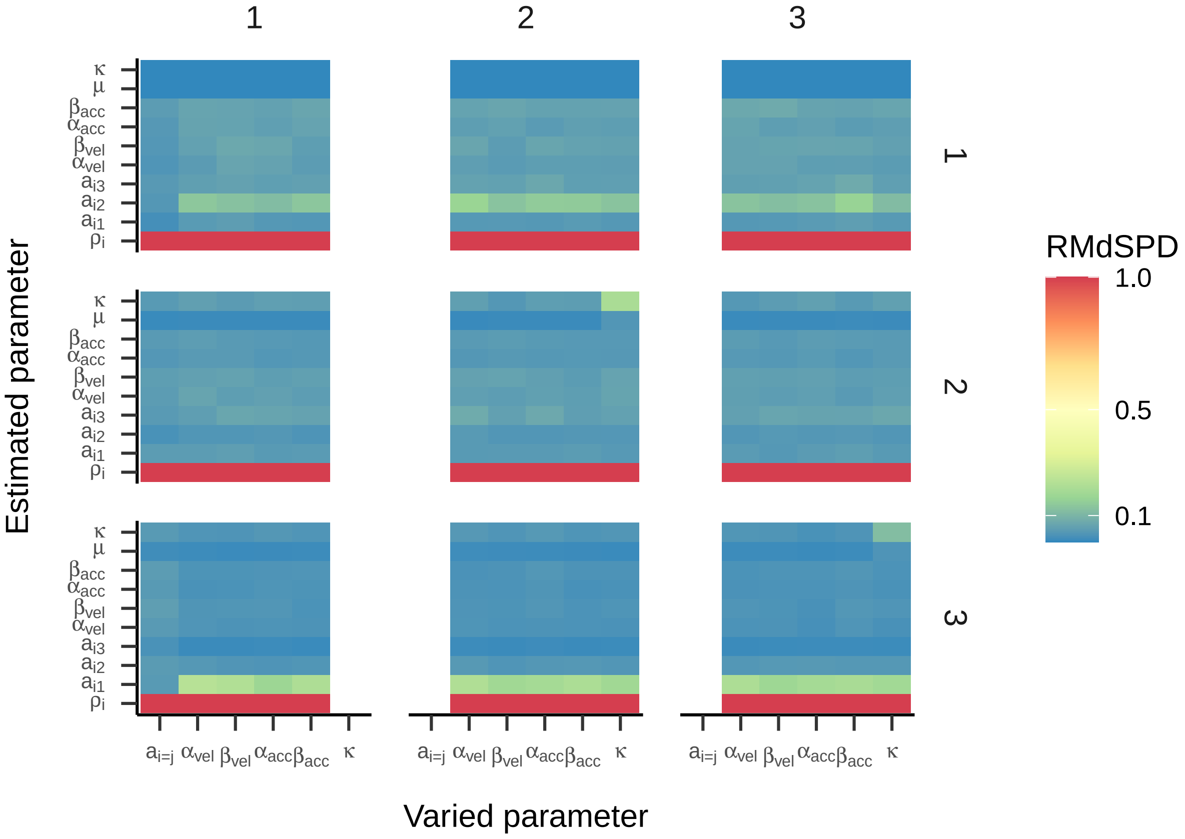

Figure 9. RMdSPD between true and estimated parameters of the three-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which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

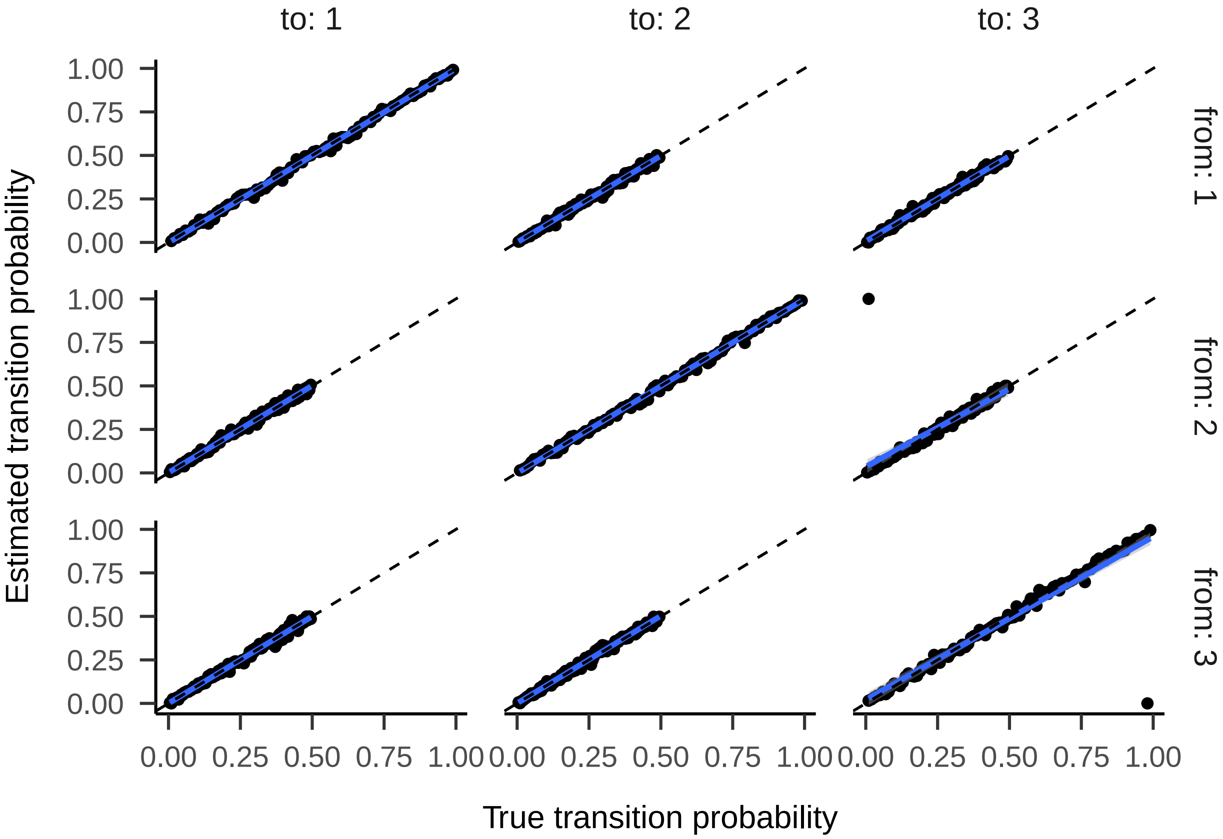

Figure 10. Regression lines between true and estimated transition probabilities for the three-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving.

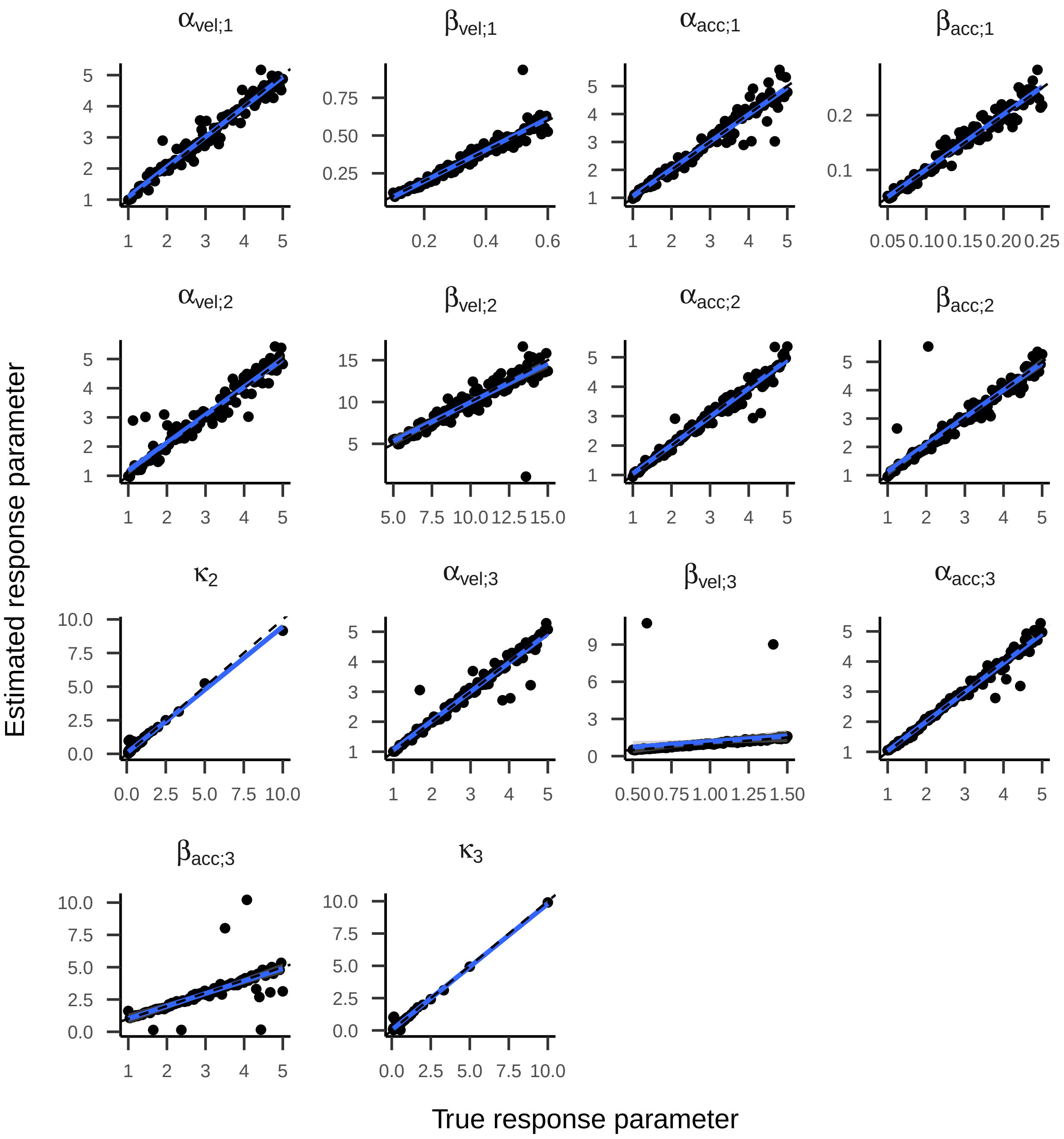

Figure 11. Regression lines between true and estimated response parameters of the three-state HMM in part one. Top facet labels indicate response parameters.

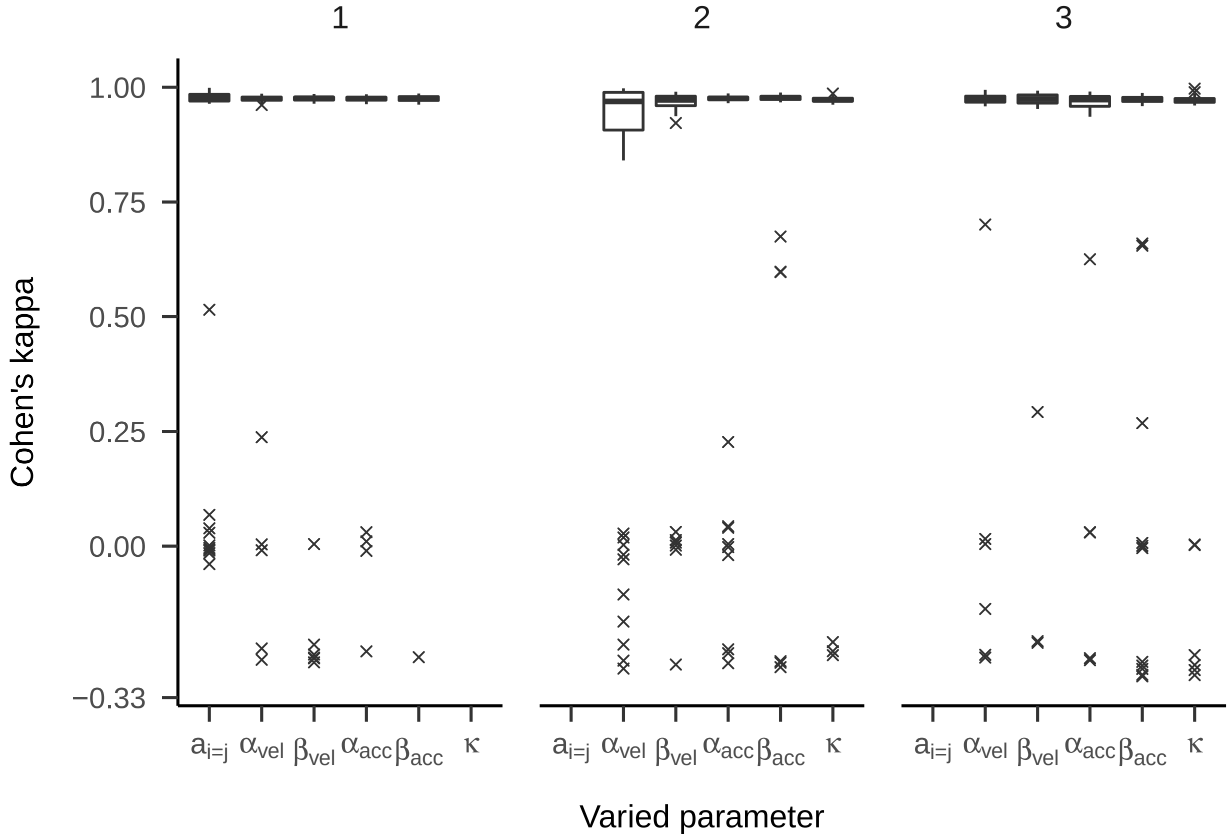

Figure 12. Boxplots displaying Cohen's kappa depending on which parameter of the three-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians and hinges the first and third quartile. Whiskers range from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

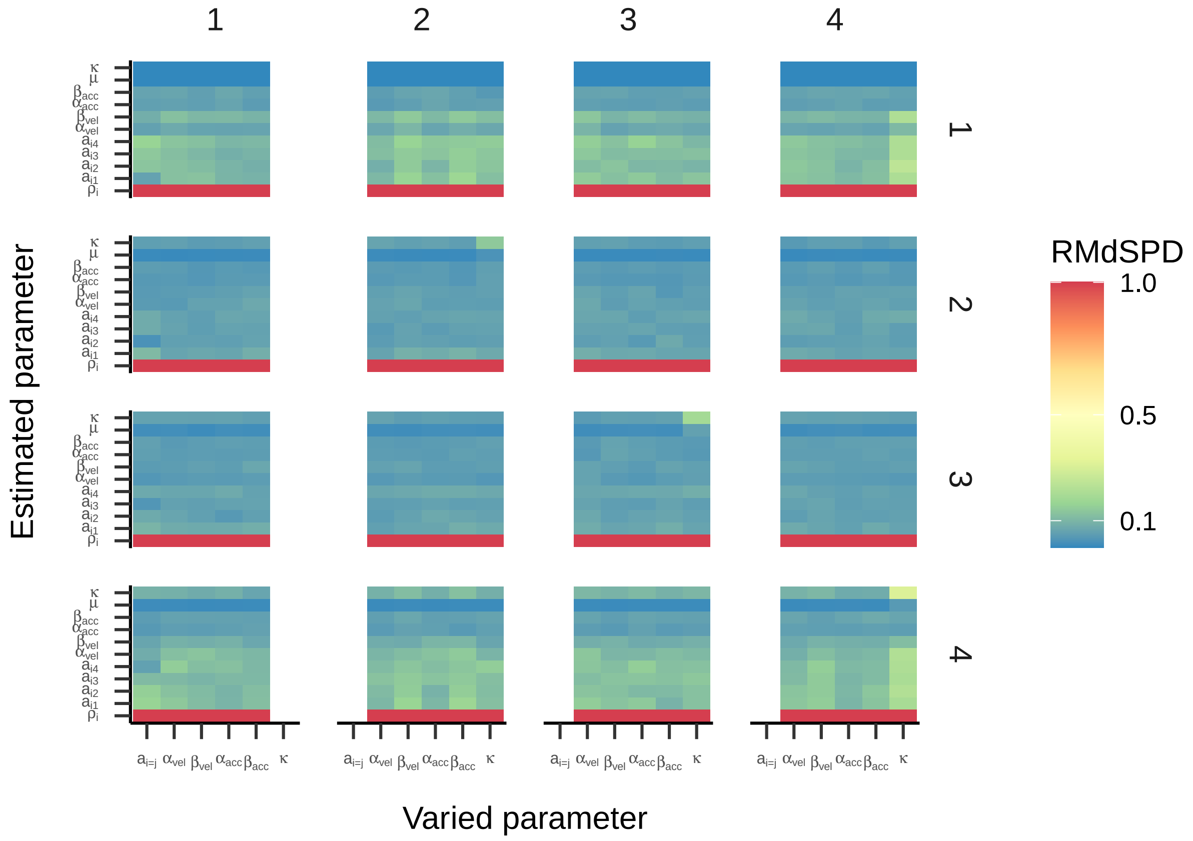

Figure 13. RMdSPD between true and estimated parameters of the four-state HMM in part one of the simulation. Labels on the x-axis indicate which true parameters have been manipulated and labels on the y-axis show for which estimated parameter the RMdSPD is displayed. Top facet labels specify in which state the parameters have been varied and right facet labels denote to which state estimated parameters belong.

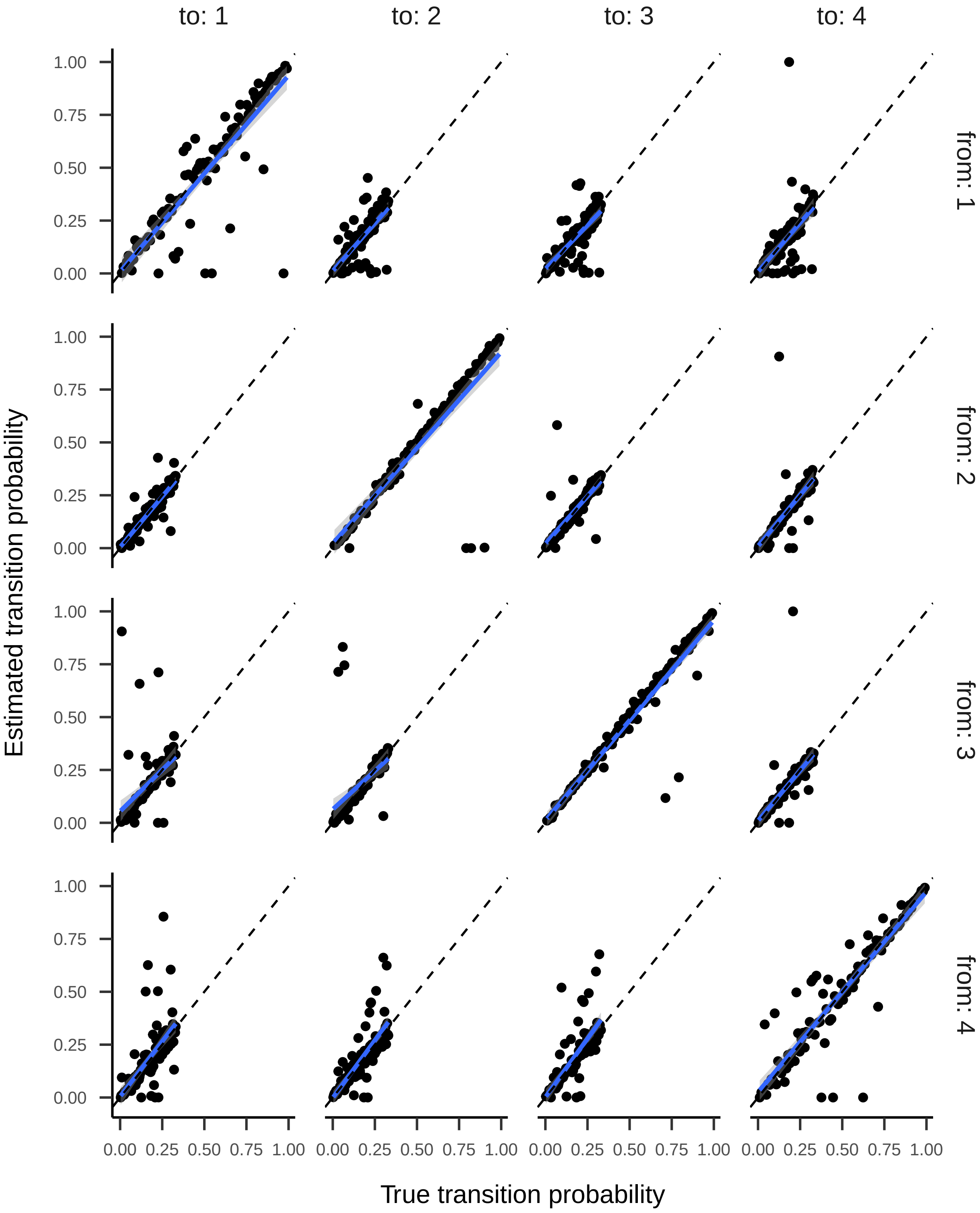

Figure 14. Regression lines between true and estimated transition probabilities for the four-state HMM in part one. Top facet labels show to and right facet labels show from which state the HMM is moving.

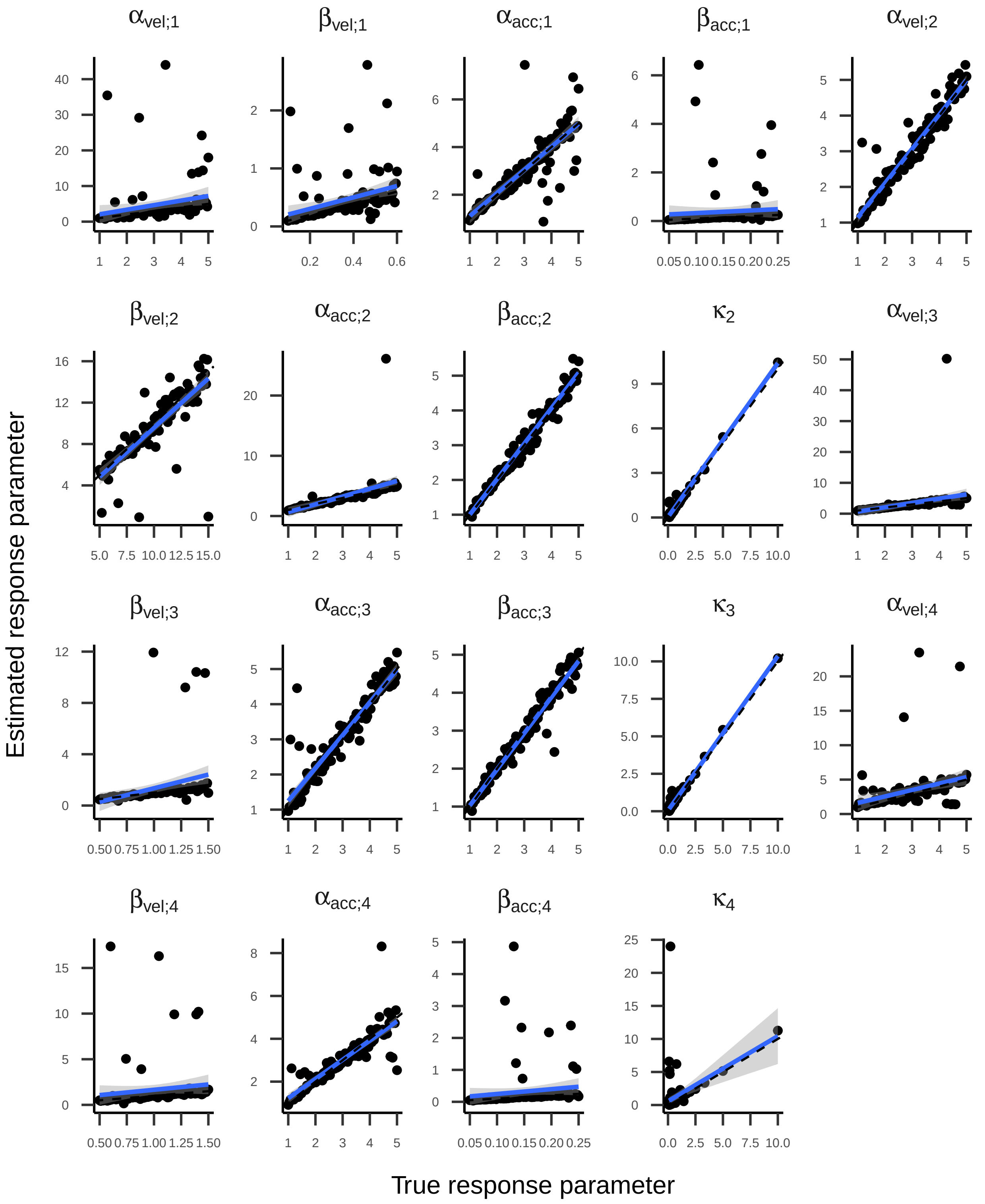

Figure 15. Regression lines between true and estimated response parameters of the four-state HMM in part one. Top facet labels indicate response parameters.

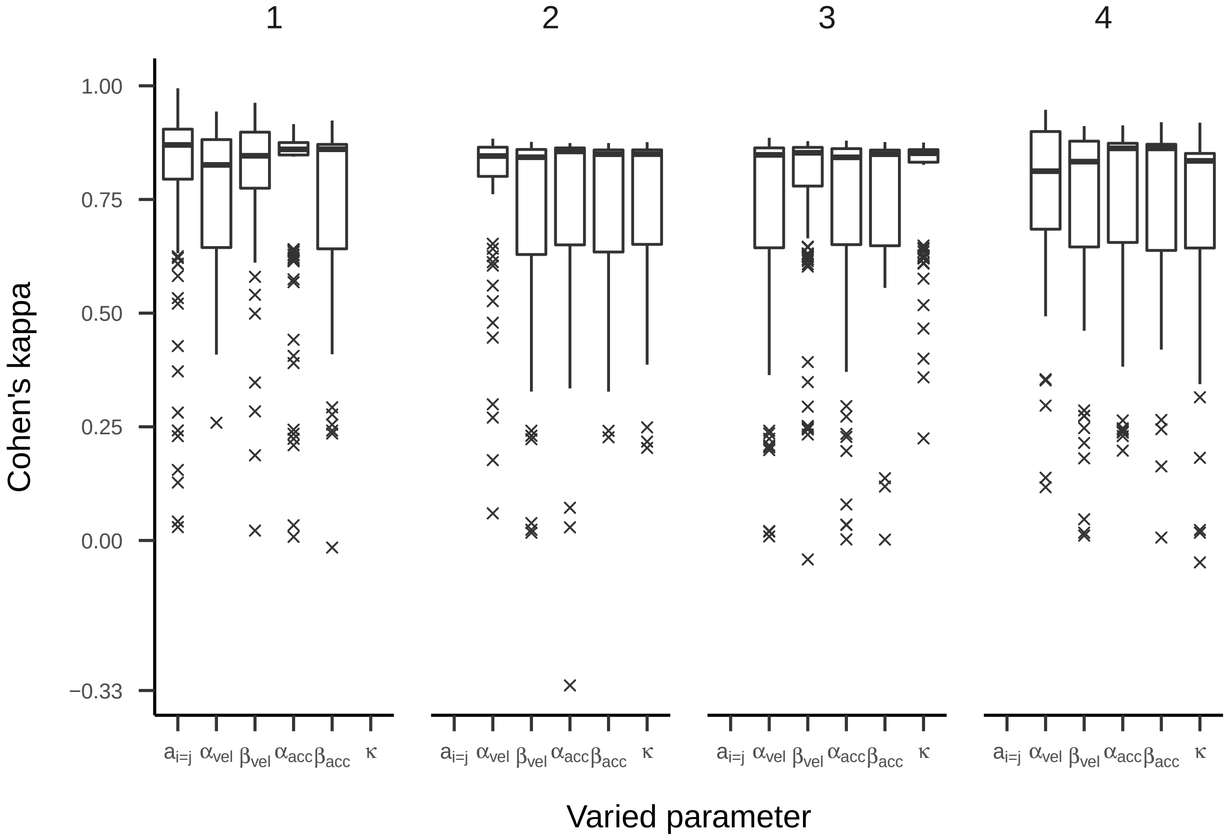

Figure 16. Boxplots displaying Cohen's kappa depending on which parameter of the four-state HMM has been manipulated in part one. Top facet labels indicate for which state parameters have been manipulated. Black solid lines symbolize medians and hinges the first and third quartile. Whiskers range from hinges to lowest/highest value within 1.5 times the IQR. Crosses represent outliers.

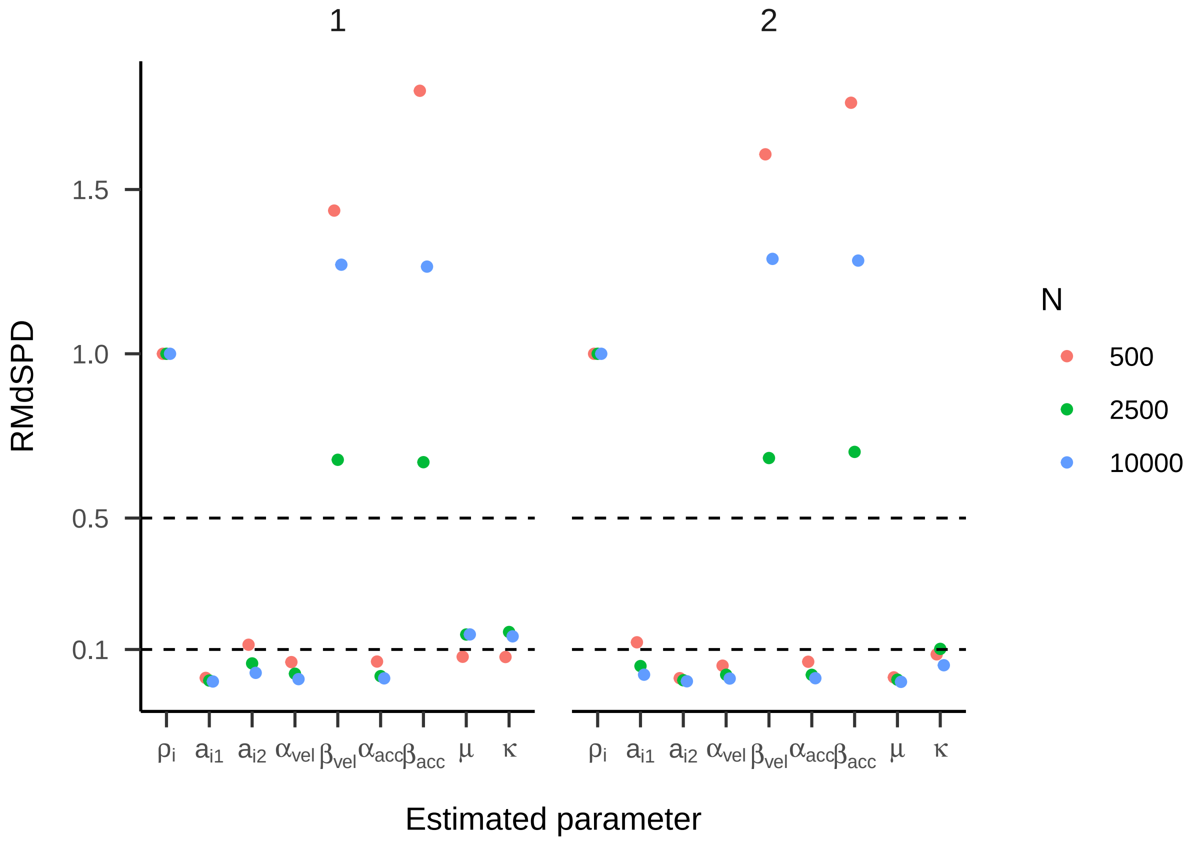

Figure 17. RMdSPD between true and estimated parameters of the two-state HMM in part two of the simulation. Colours indicate different sizes of generated data. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

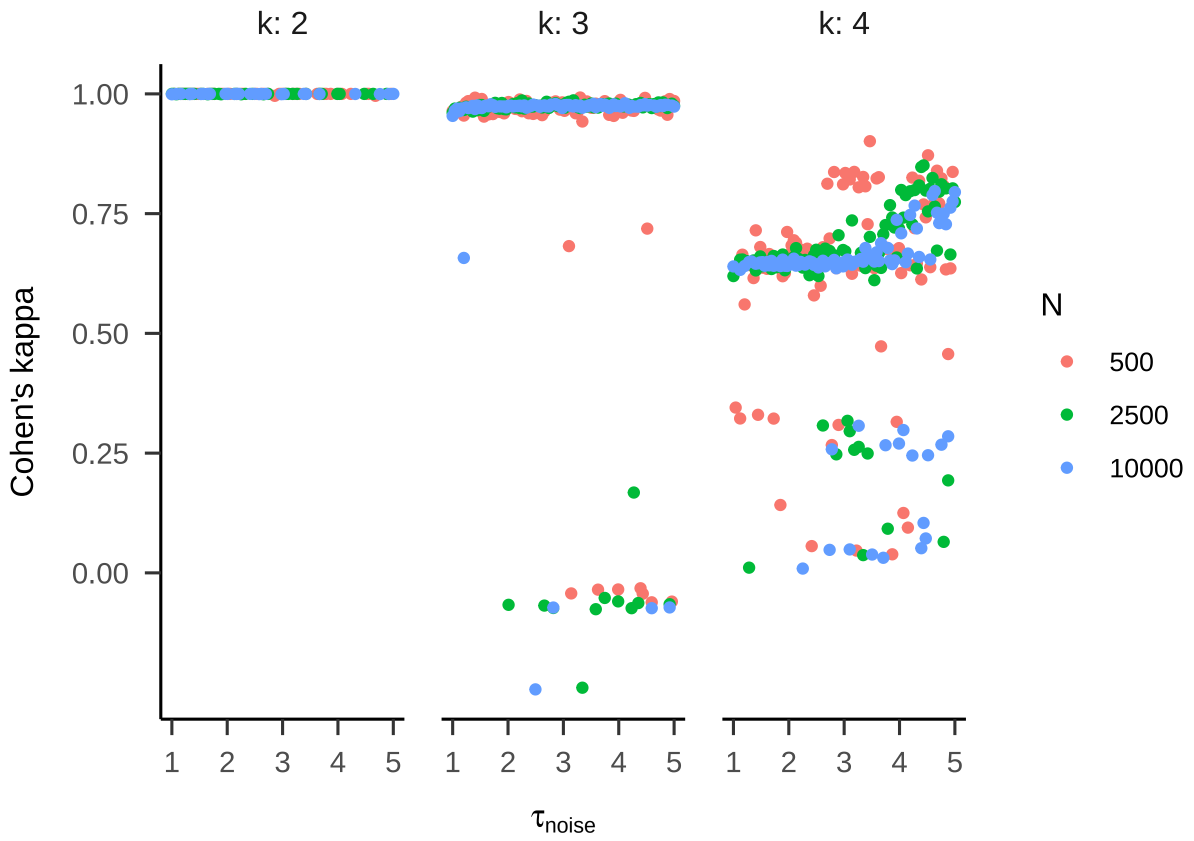

Figure 18. Cohen's kappa depending on the variation of noise added to the data generated by the HMM. Colours indicate different sizes of generated data. Top facet labels indicate the number of states in the HMM.

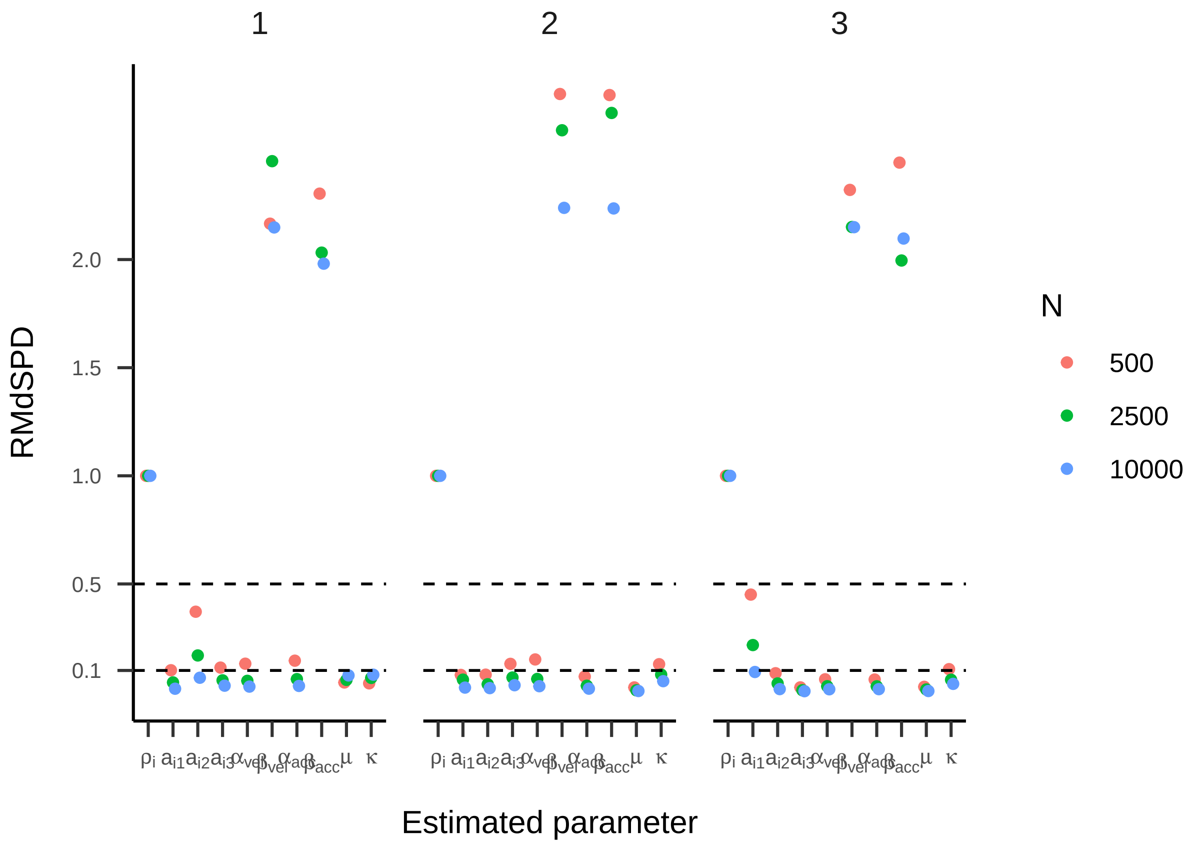

Figure 19. RMdSPD between true and estimated parameters of the three-state HMM in part two of the simulation. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

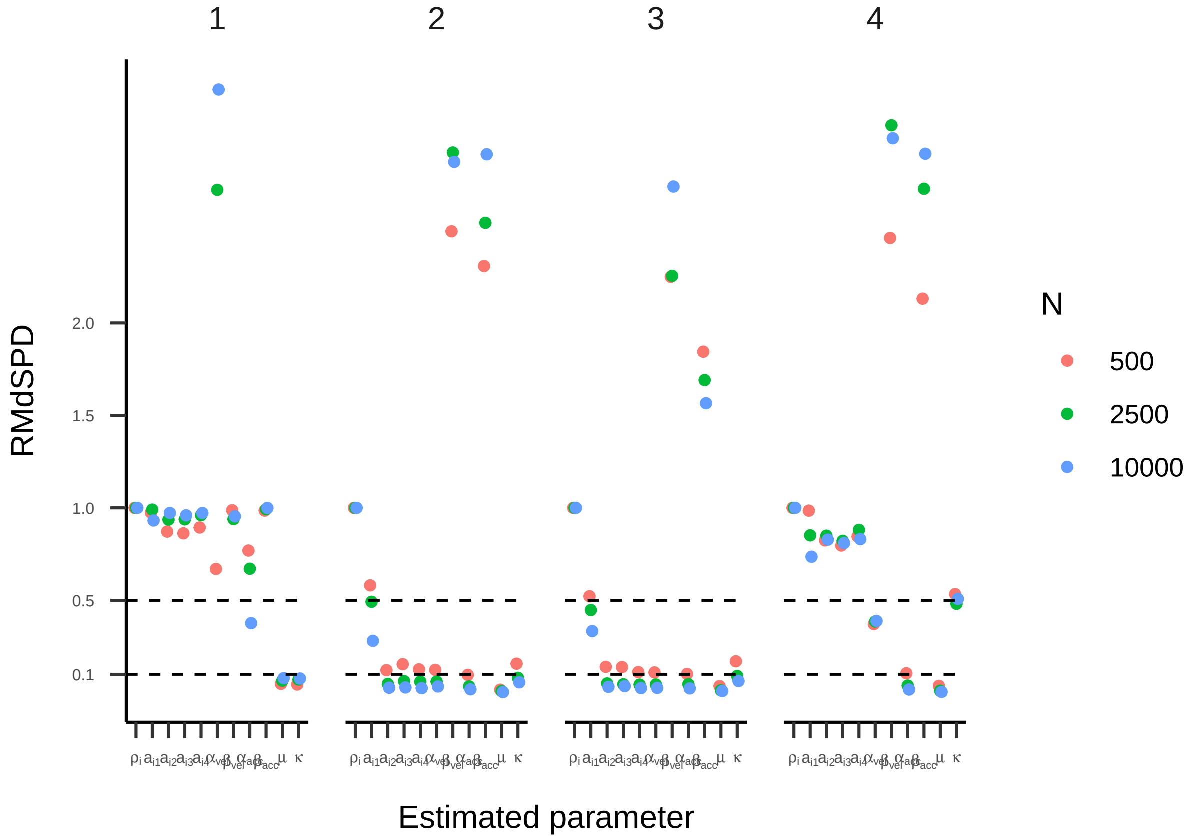

Figure 20. RMdSPD between true and estimated parameters of the three-state HMM in part two of the simulation. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

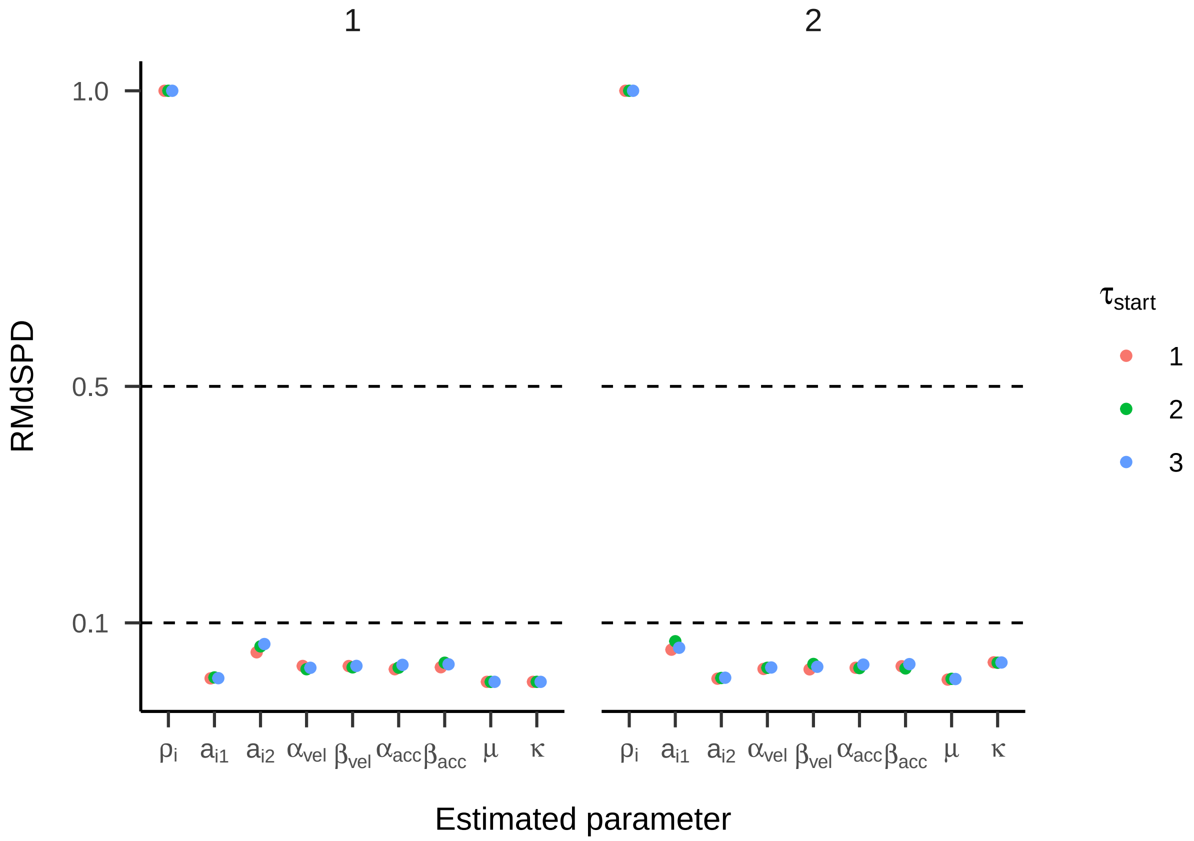

Figure 21. RMdSPD between true and estimated parameters of the two-state HMM in part three of the simulation. Colours indicate the variation in starting values used to estimate parameters. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

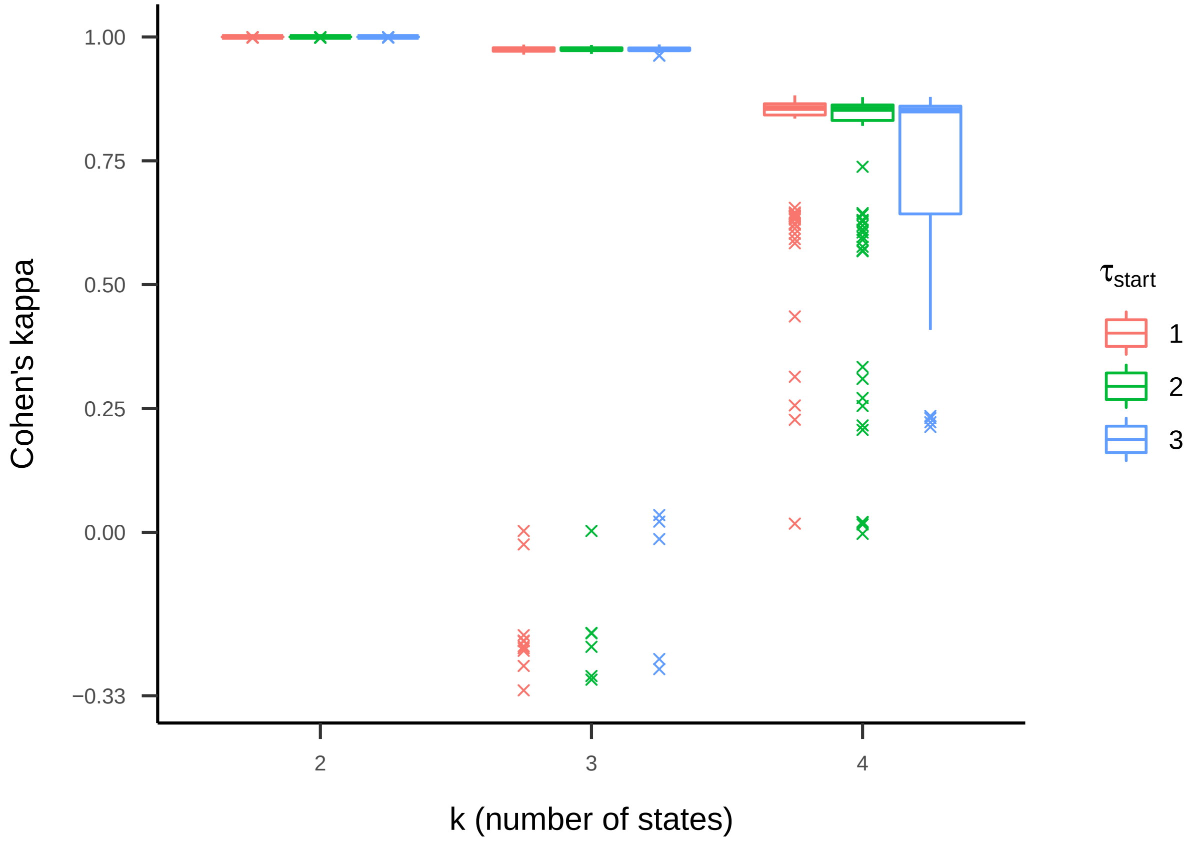

Figure 22. Boxplots displaying Cohen's kappa depending on the number of states in the HMM in part three. Colours indicate the variation in starting values used to estimate parameters. Solid vertical lines symbolize medians and hinges the first and third quartile. Whiskers range from hinges to lowest/highest value within 1.5 times the IQR. Crosses represent outliers.

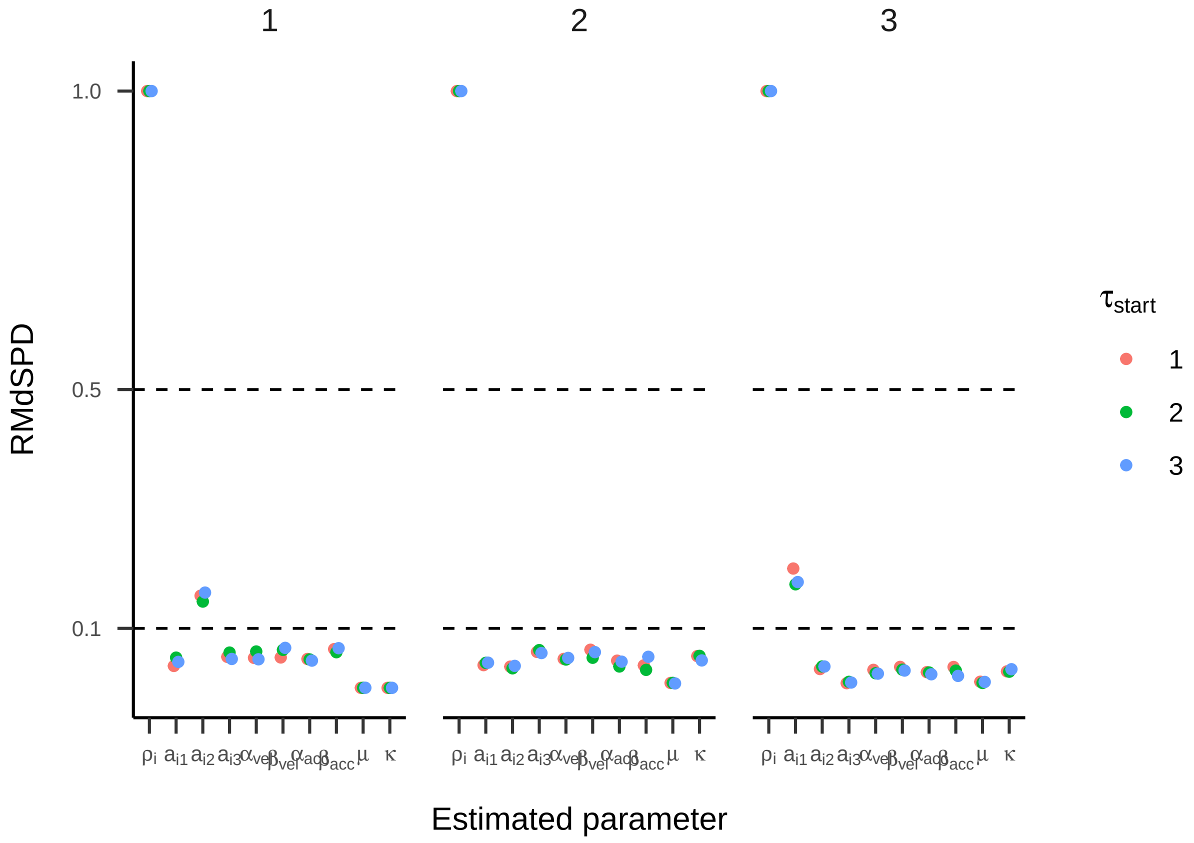

Figure 23. RMdSPD between true and estimated parameters of the three-state HMM in part three of the simulation. Colours indicate the variation in starting values used to estimate parameters. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

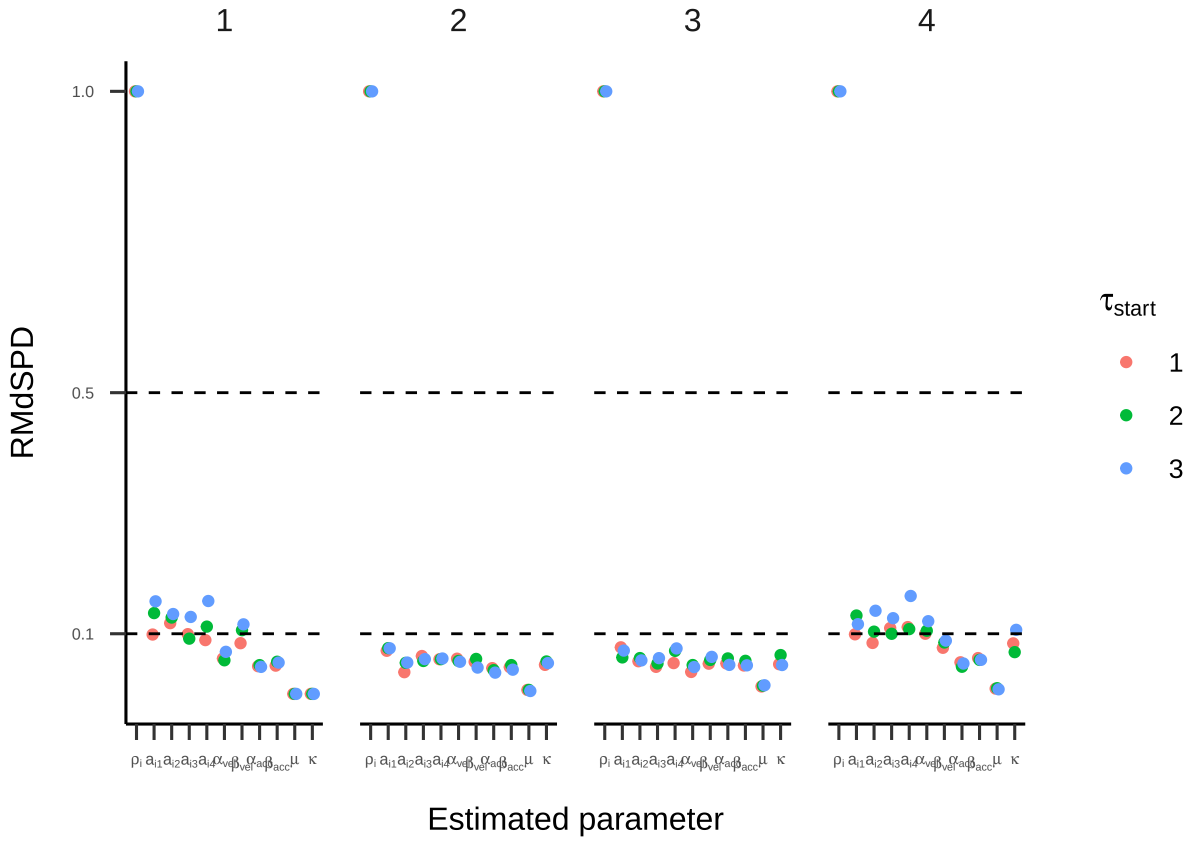

Figure 24. RMdSPD between true and estimated parameters of the four-state HMM in part three of the simulation. Colours indicate the variation in starting values used to estimate parameters. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

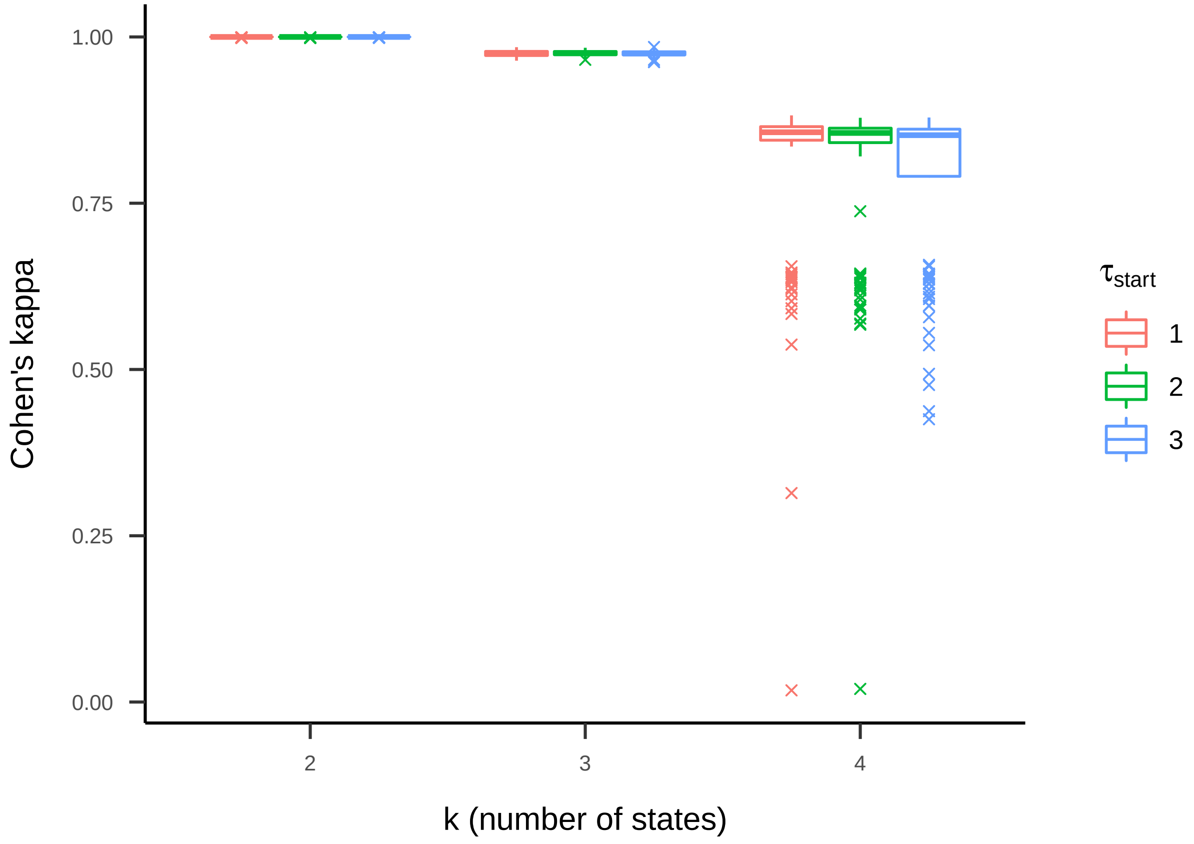

Figure 25. Boxplots displaying Cohen's kappa depending on the number of states in the HMM in part three after state labels were switched post-hoc (exploratory analysis). Colours indicate the variation in starting values used to estimate parameters. Solid vertical lines symbolize medians and hinges the first and third quartile. Whiskers range from hinges to lowest/highest value within 1.5 times the IQR. Crosses represent outliers.

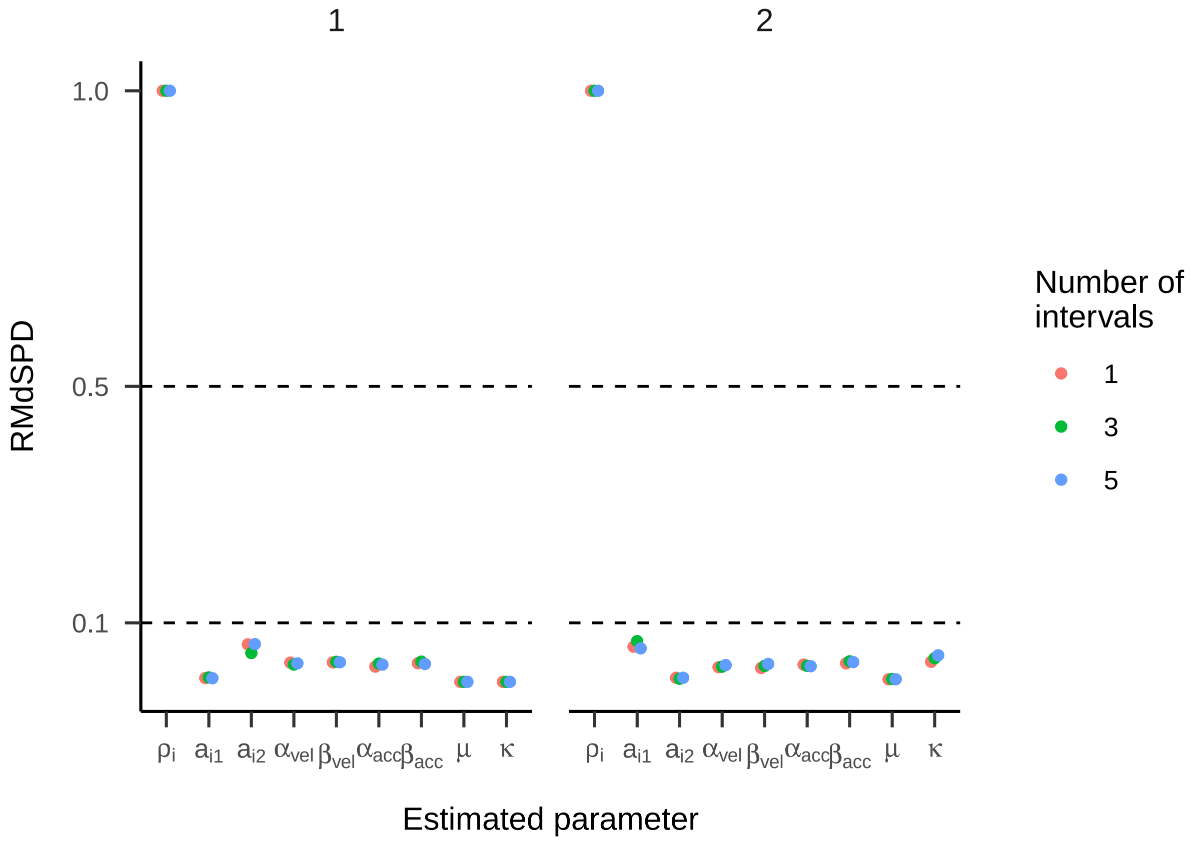

Figure 26. RMdSPD between true and estimated parameters of the two-state HMM in part four of the simulation. Colours indicate the number of missing data intervals. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

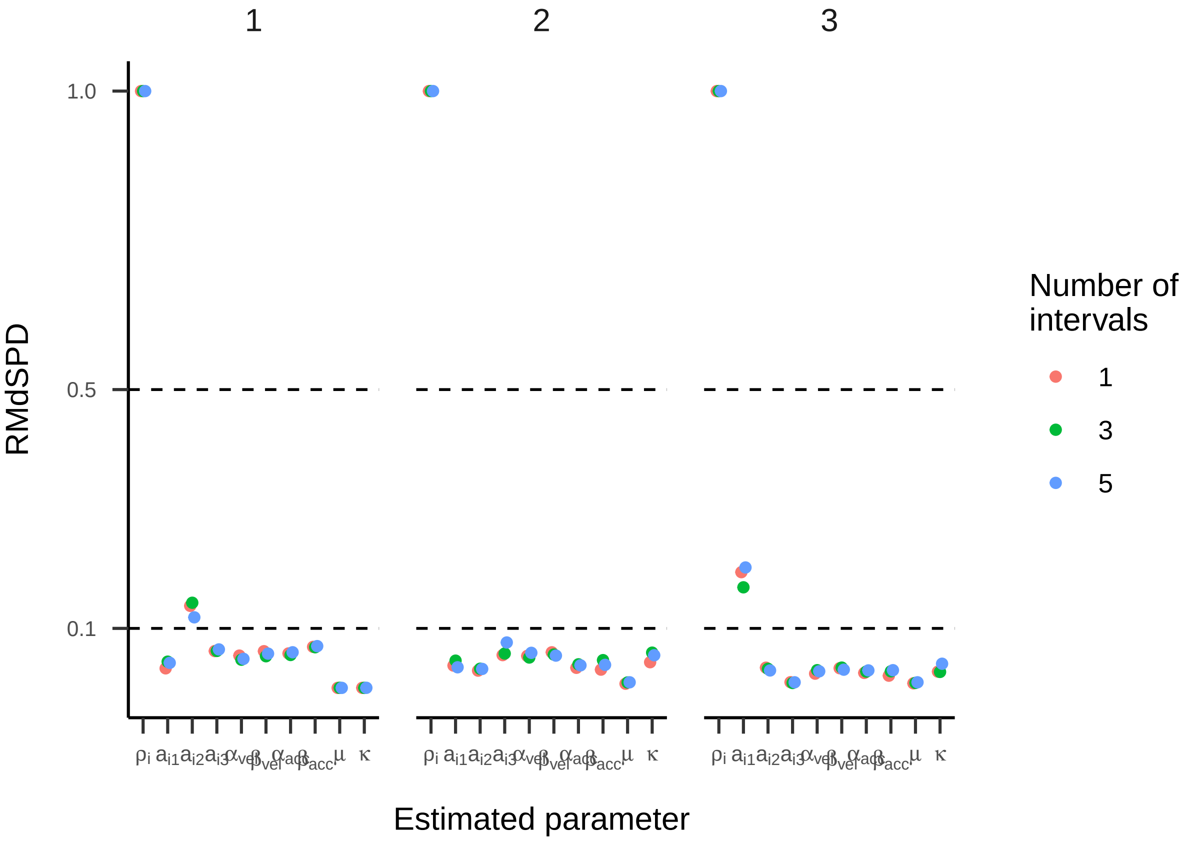

Figure 27. RMdSPD between true and estimated parameters of the three-state HMM in part four of the simulation. Colours indicate the number of missing data intervals. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

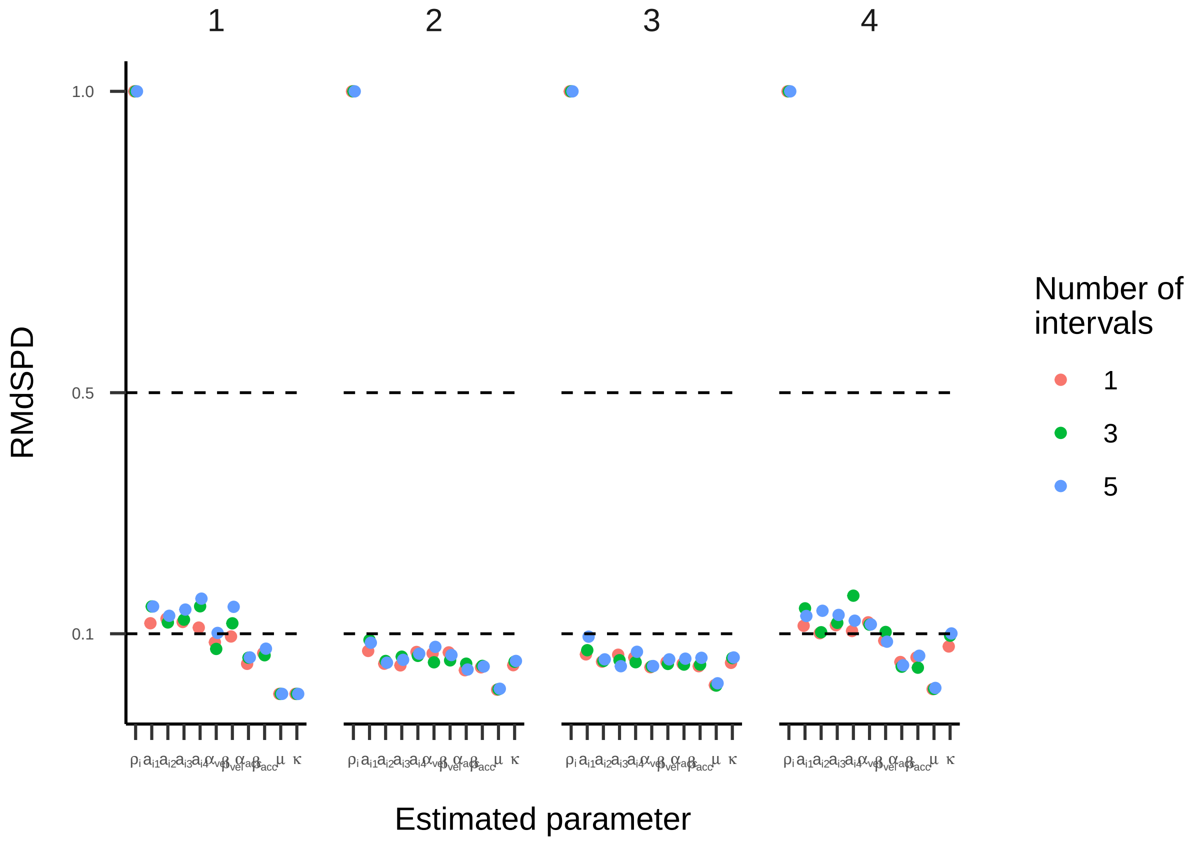

Figure 28. RMdSPD between true and estimated parameters of the four-state HMM in part four of the simulation. Colours indicate the number of missing data intervals. Labels on the x-axis indicate for which estimated parameter the RMdSPD is displayed. Top facet labels denote to which state estimated parameters belong.

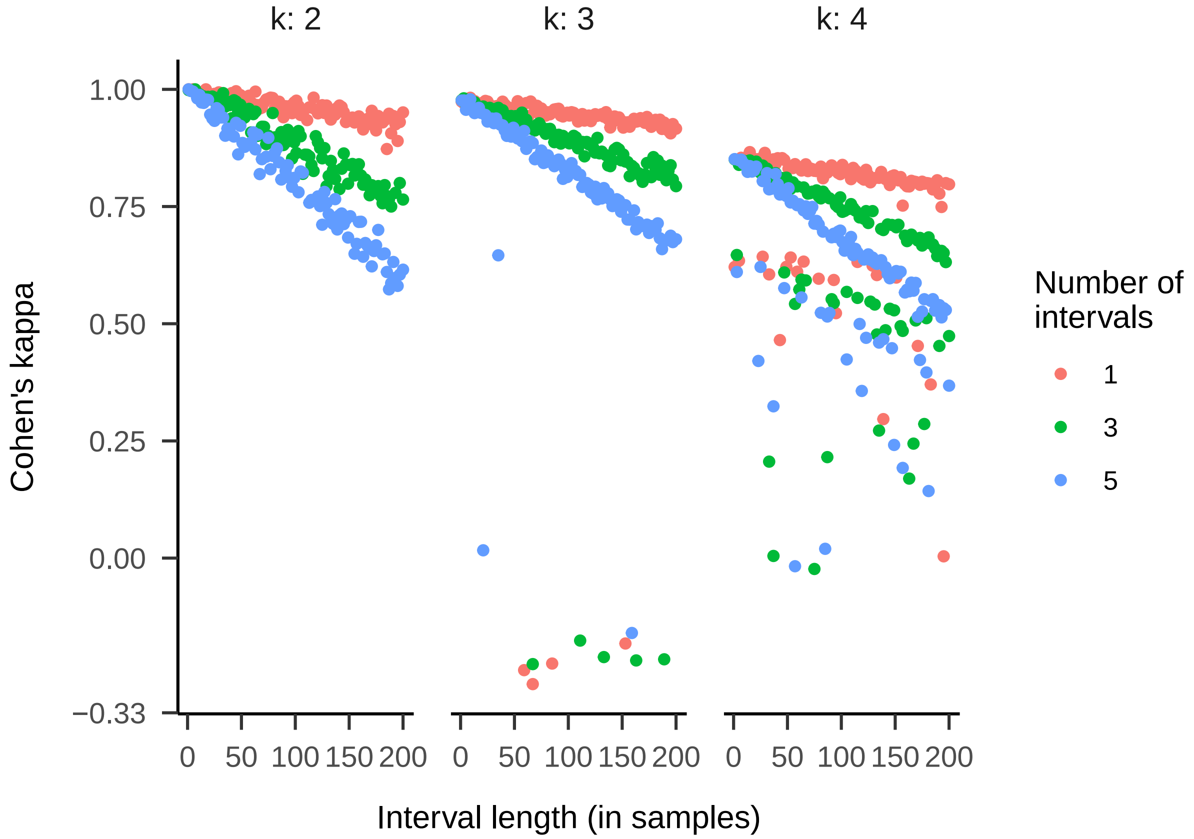

Figure 29. Cohen's kappa depending on the length of missing data intervals. Colours indicate the number of missing data intervals. Top facet labels indicate the number of states in the HMM.

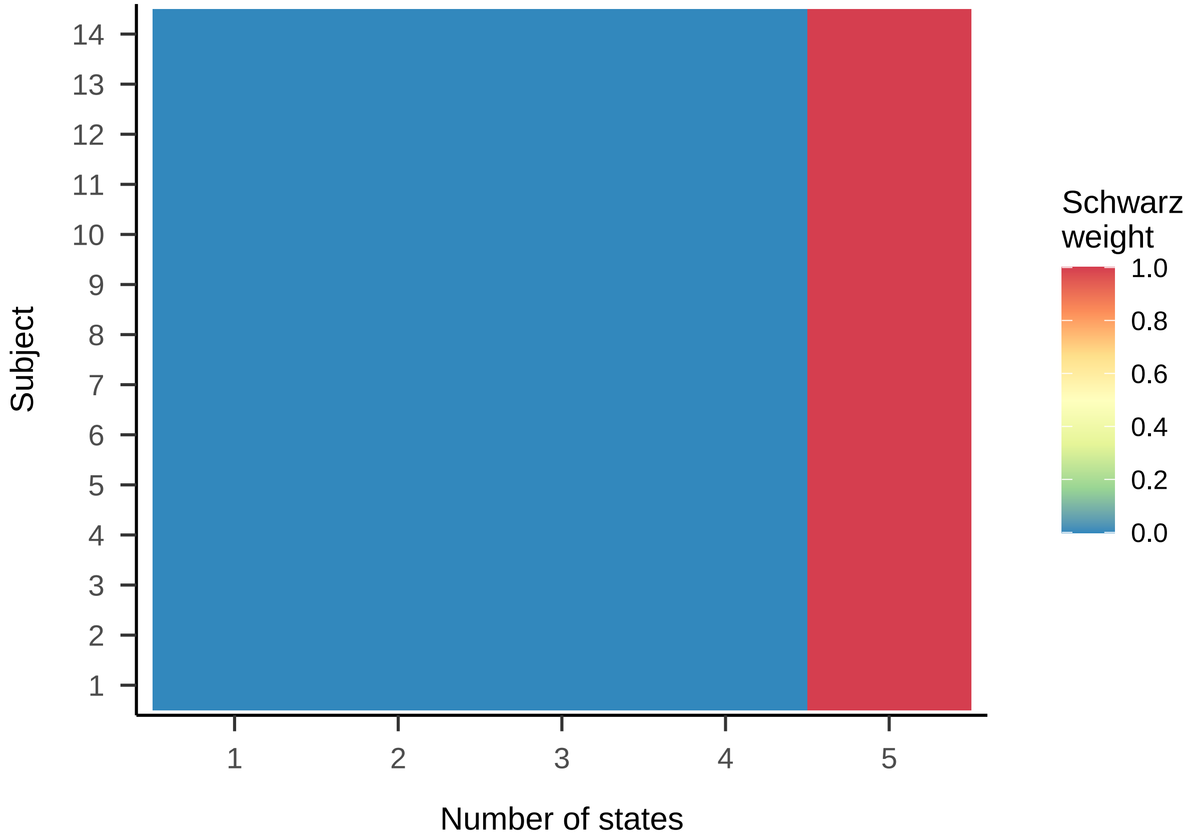

Figure 30. Schwarz weights displayed for each subject and HMMs with different numbers of states. Models were applied to the image condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

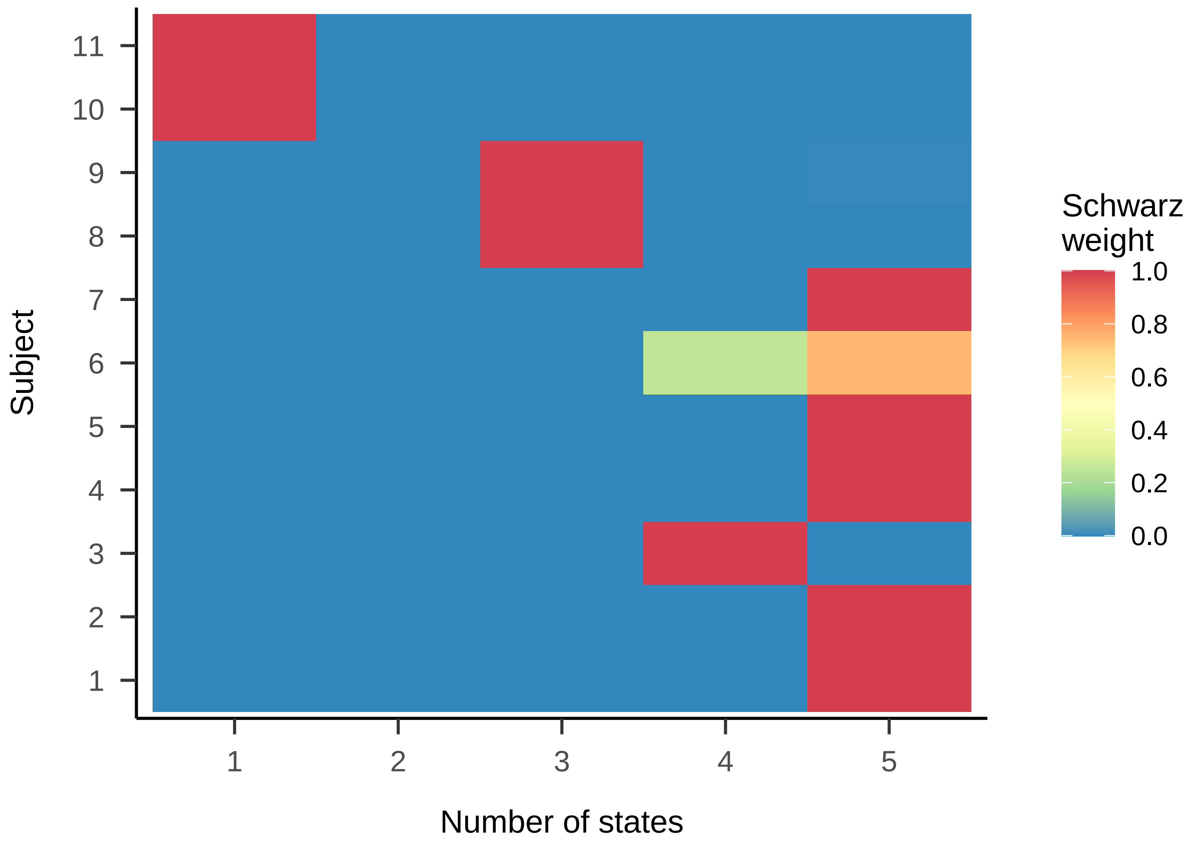

Figure 31. Schwarz weights displayed for each subject and HMMs with different numbers of states. Models were applied to the moving dots condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

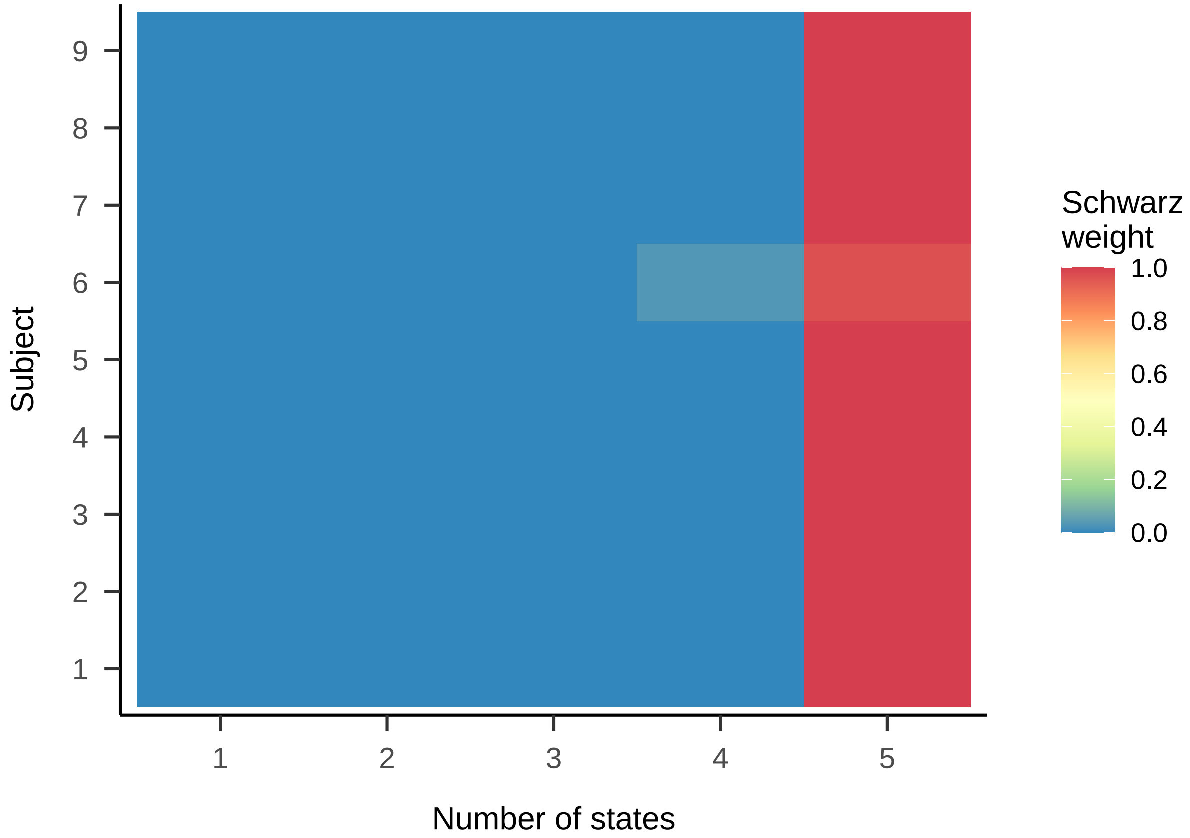

Figure 32. Schwarz weights displayed for each subject and HMMs with different numbers of states. Models were applied to the video condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

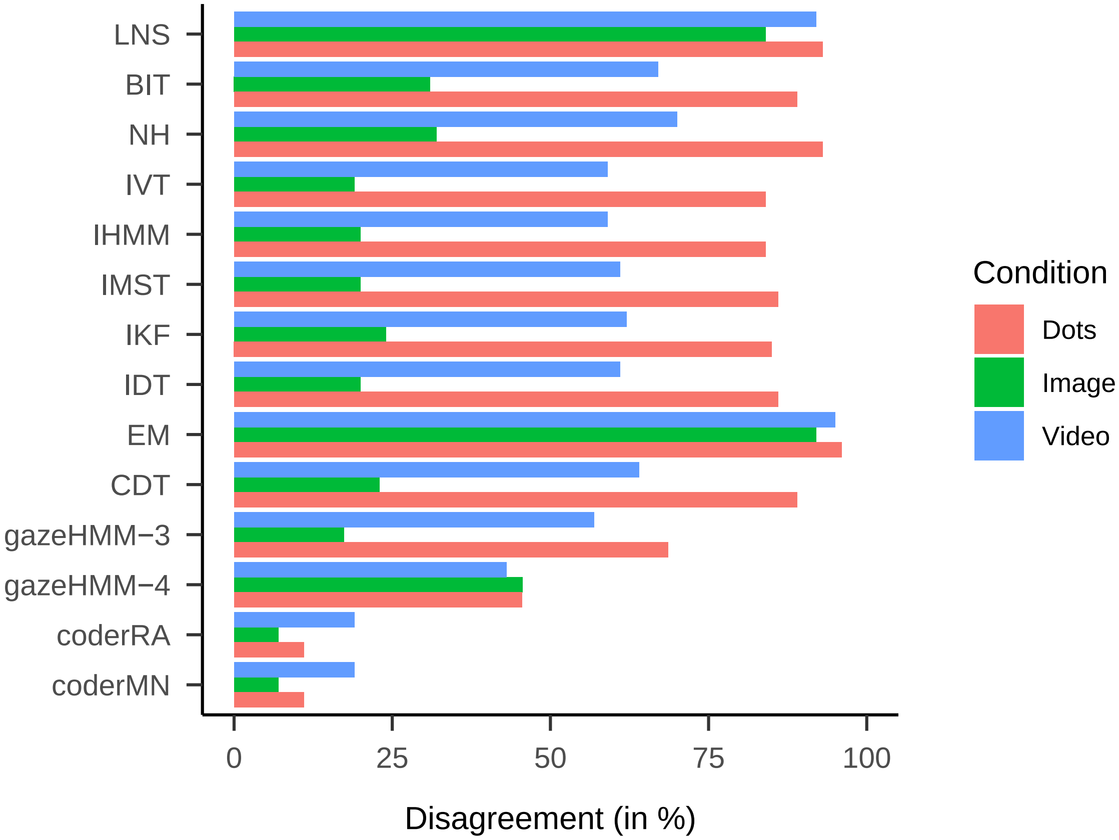

Figure 33. Disagreement between algorithms and human coders for different conditions (in %). gazeHMM-3 classified three and gazeHMM-4 classified four events. Figure adapted from Andersson et al. (2017).

Appendix A

Additional Figures and Tables

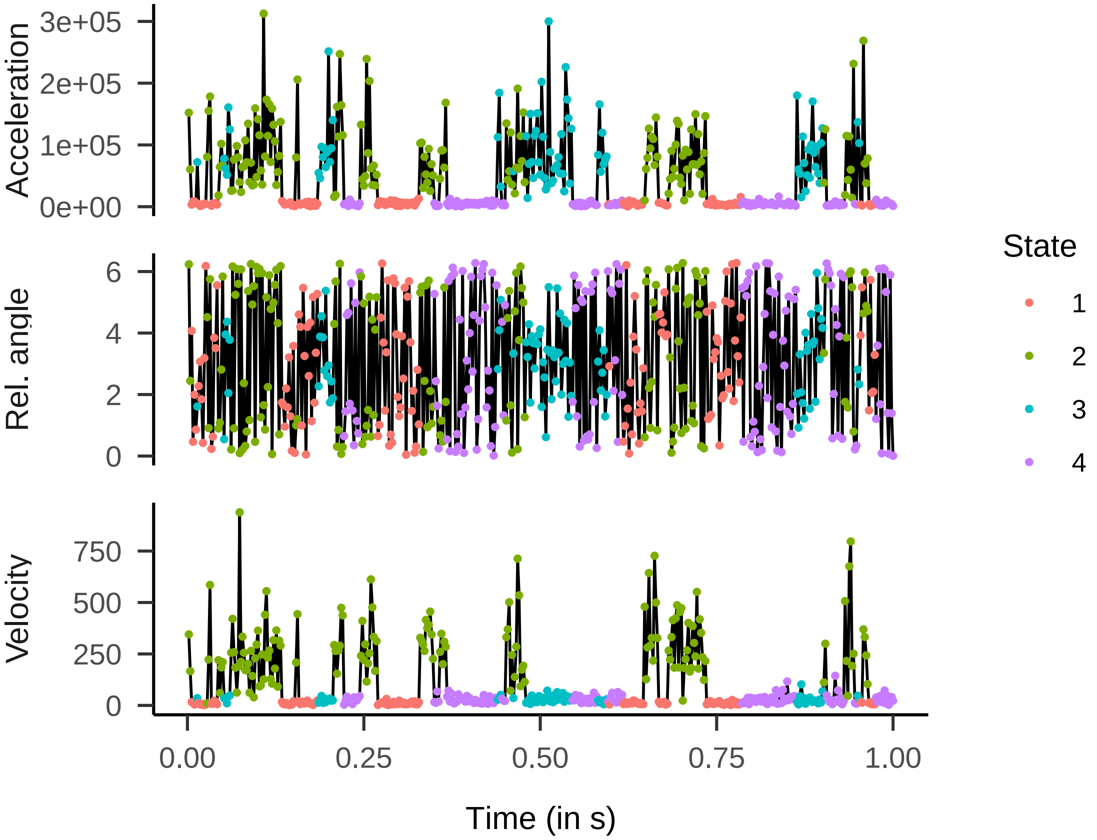

Figure A1. Simulated data from the HMM with the default parameters of the simulation study. Displayed are acceleration (in deg/s^2, multiplied by 10000), sample-to-sample (relative) angle (in rad), and velocity (in deg/s, multiplied by 10) over time (in s). Acceleration and velocity were multiplied back to their natural scale. Note that PSOs (State 3, blue dots) do not only occur after saccades because the HMM parameters were unconstrained.

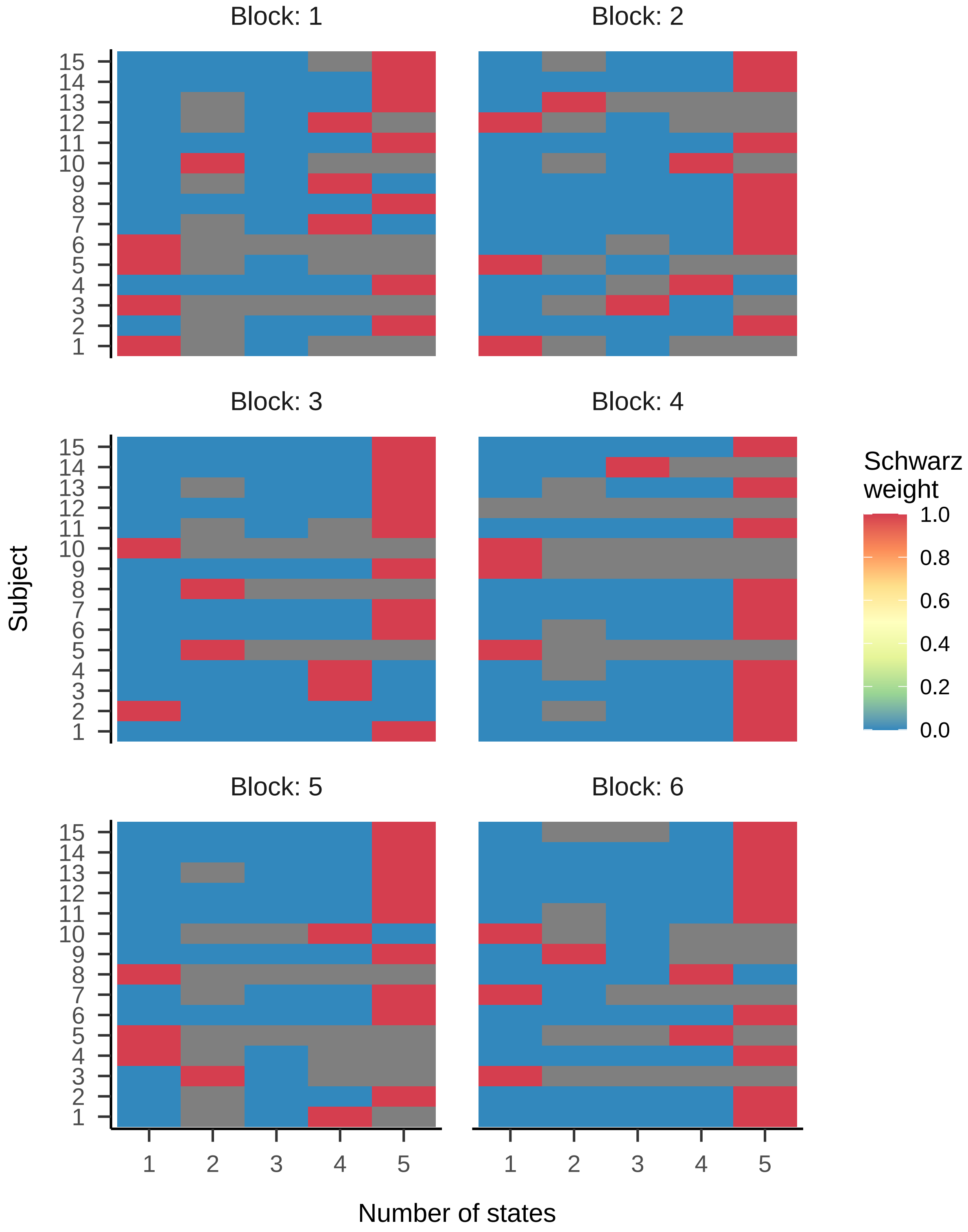

Figure A2. Schwarz weights displayed for each subject and HMMs with different numbers of states. Models were applied to task 4 of the Ehinger et al. (2019) data set. Higher weights indicate better model fit. Grey tiles indicate erroneous model fits.

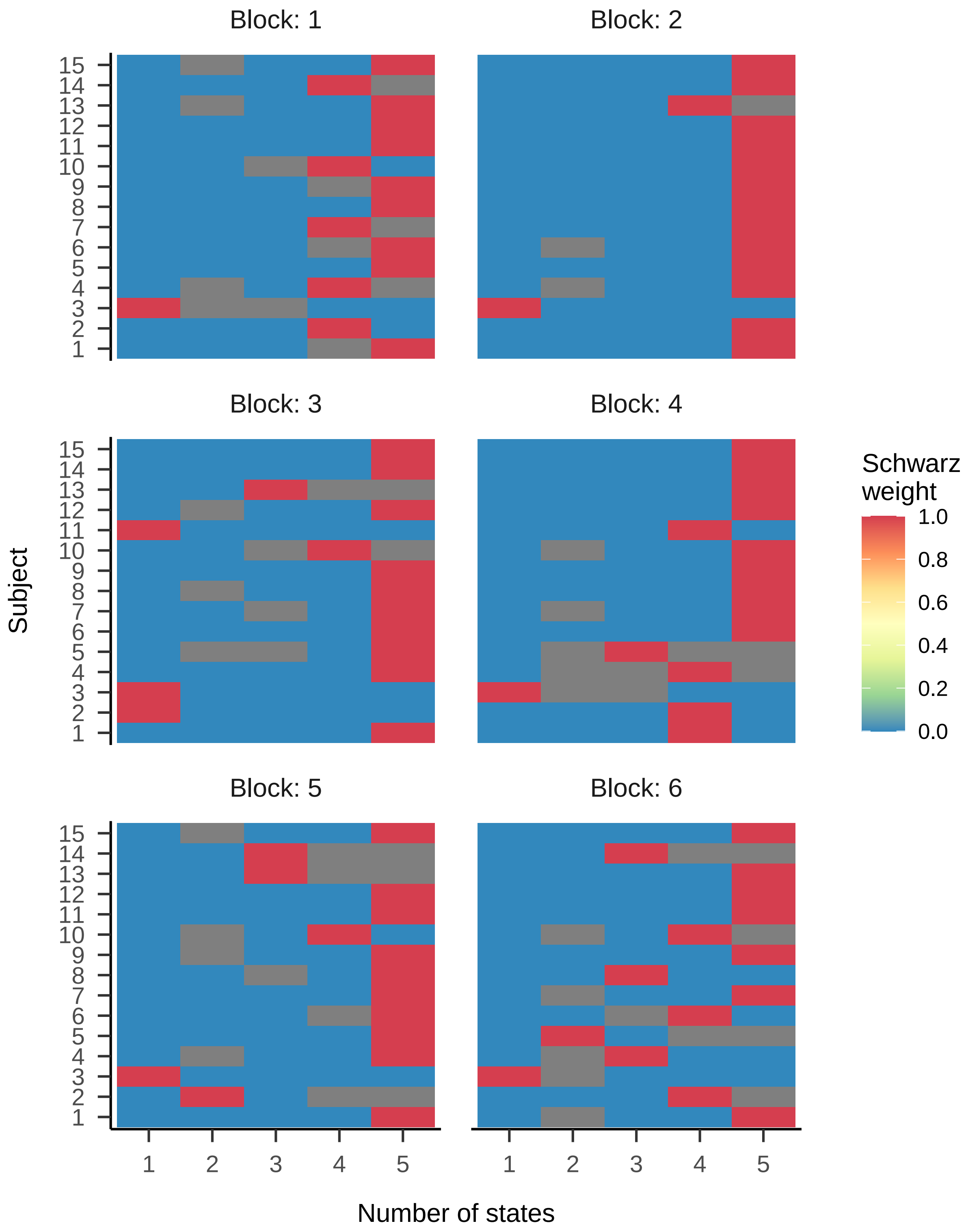

Figure A3. Schwarz weights displayed for each subject and HMMs with different numbers of states. Models were applied to task 5 of the Ehinger et al. (2019) data set. Higher weights indicate better model fit. Grey tiles indicate erroneous model fits.

Table A1

Transition probabability estimates for example data from Andersson et al. (2017) (ref:tab-example-trans-note) Rows indicate from which state and columns into which state the model transitions.

To state			
1	2	3	4
0.89	0.00	0.02	0.09
0.00	0.88	0.12	0.00
0.00	0.03	0.87	0.10
0.14	0.00	0.04	0.82

Note. (ref:tab-example-trans-note)

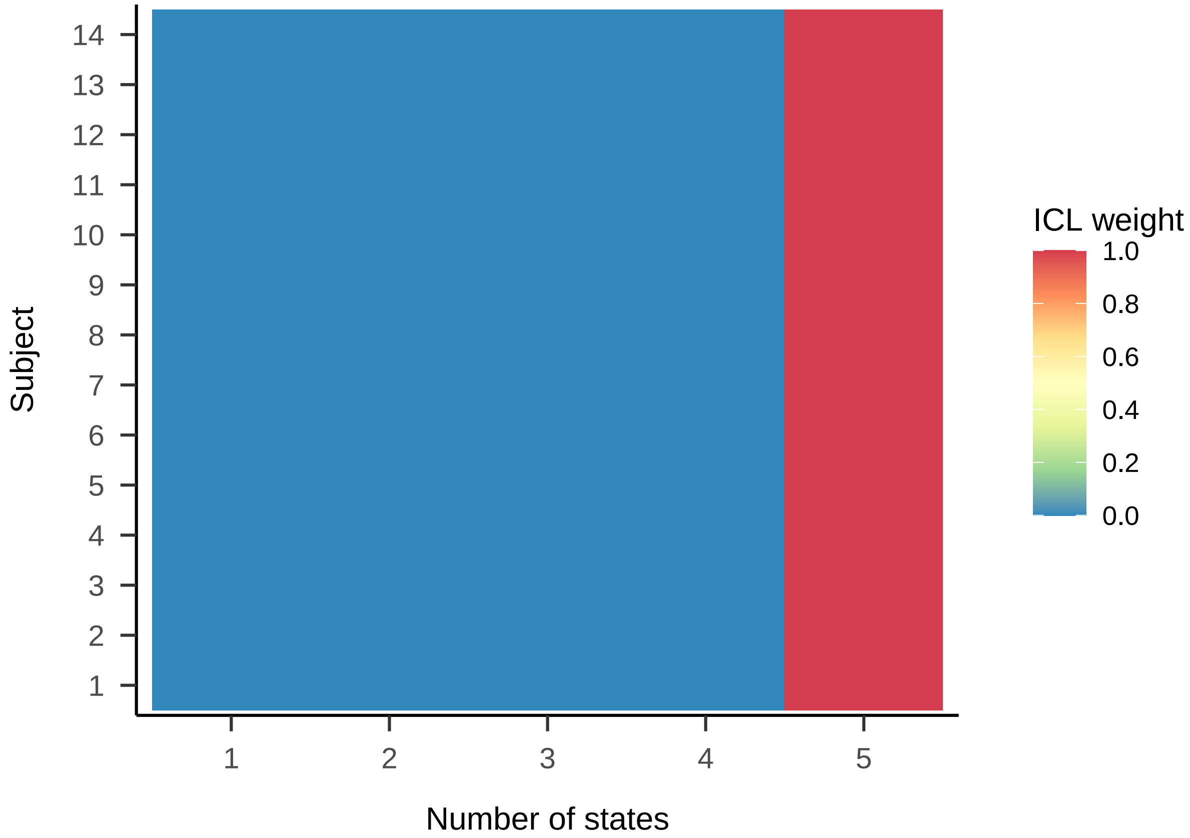

Figure A4. ICL weights displayed for each subject and HMMs with different numbers of states. Models were applied to the image condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

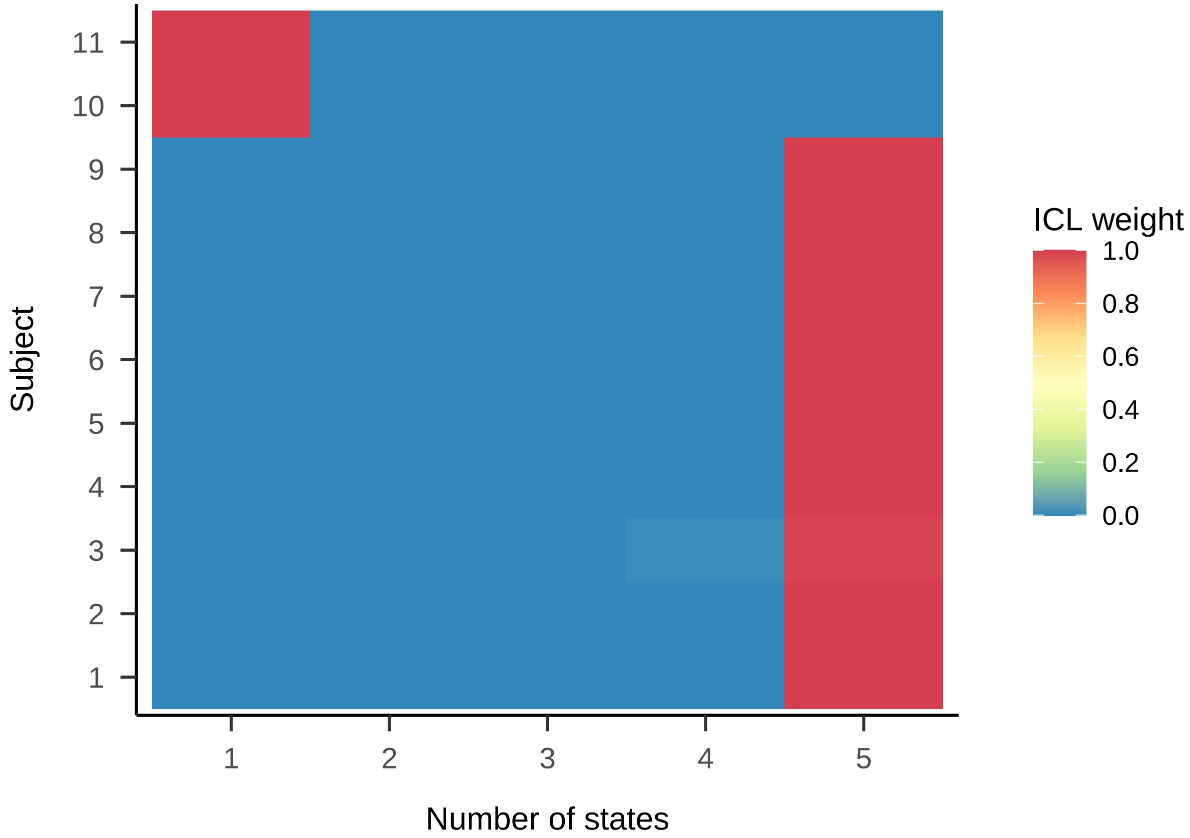

Figure A5. ICL weights displayed for each subject and HMMs with different numbers of states. Models were applied to the moving dots condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

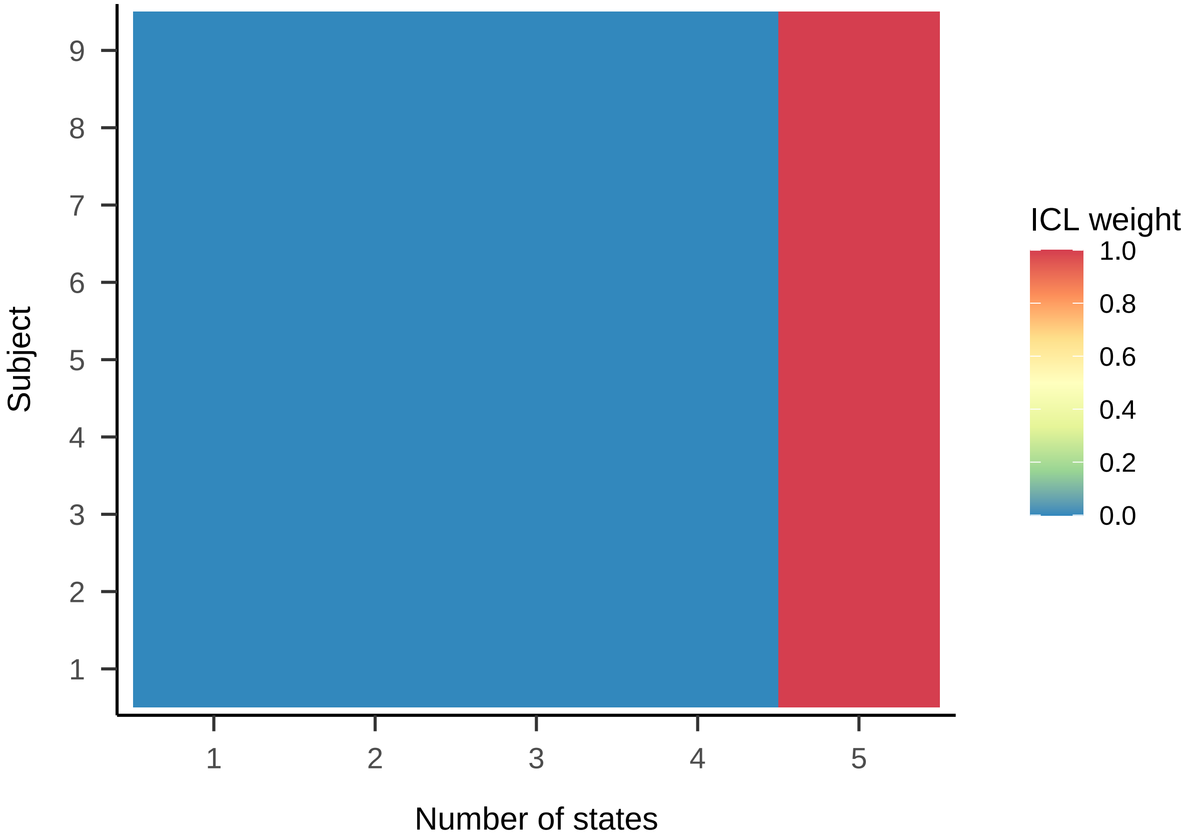

Figure A6. ICL weights displayed for each subject and HMMs with different numbers of states. Models were applied to the video condition of the Andersson et al. (2017) data set. Higher weights indicate better model fit.

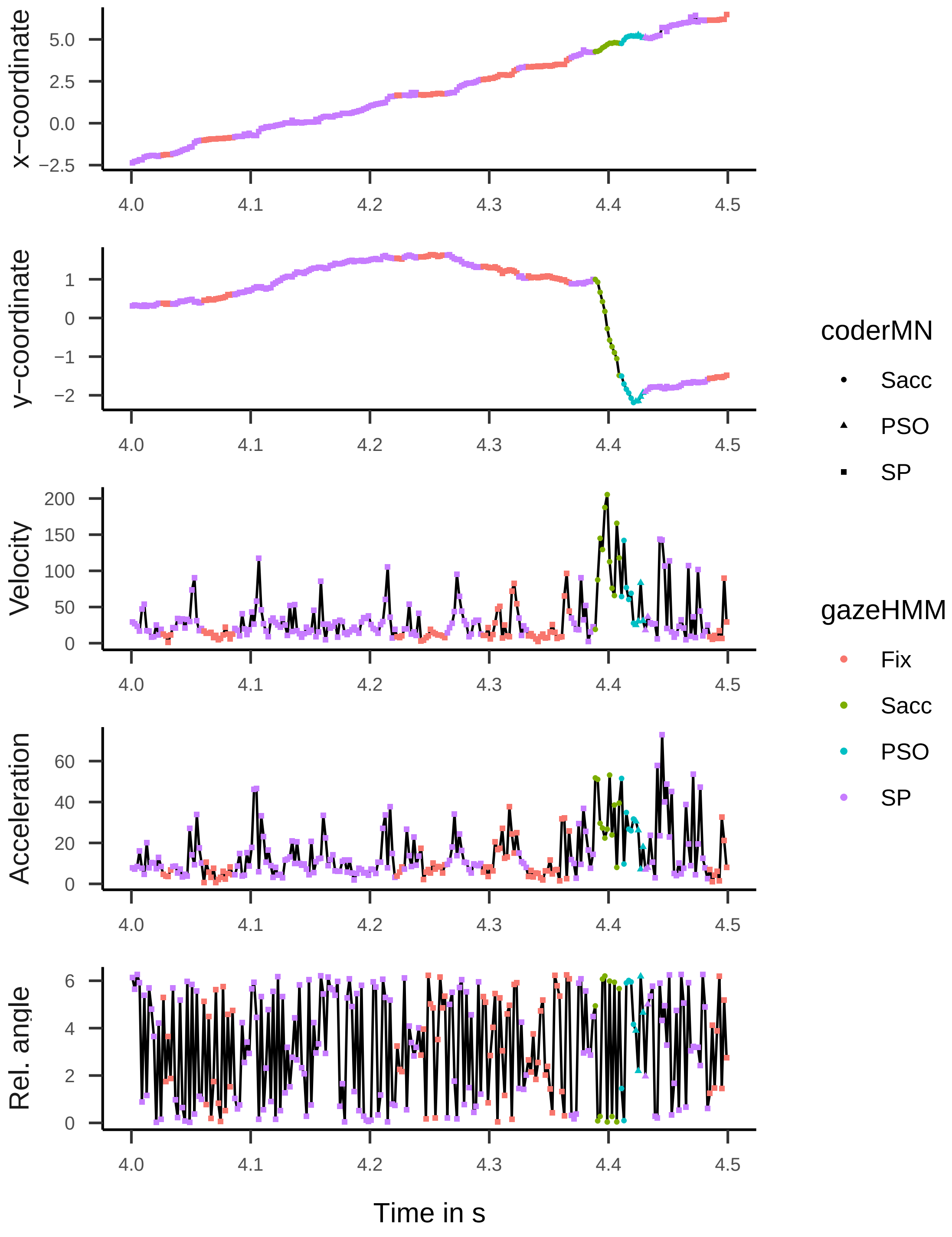

Figure A7. Example data from Andersson et al. (2017) displayed as x-, and y-coordinates, velocity (in deg/s), acceleration (in deg/s), and sample-to-sample (relative) angle (in rad). Colors indicate classification by gazeHMM and symbols by human coder MN. Note that the human coder classified all fixations from gazeHMM as smooth pursuits and delayed the separation between the saccade and PSOs.

Table A2

Response parameter estimates for example data from Andersson et al. (2017) (ref:tab-example-resp-note) Con = concentration parameter of the von-Mises distribution.

Shape (vel)	Scale (vel)	Shape (acc)	Scale (acc)	Mean	Con
3.23	0.03	2.88	0.02	-	-
3.08	0.75	2.12	0.23	-0.02	12.47
1.31	0.38	2.27	0.13	5.02	0.11
5.00	0.04	3.66	0.03	-0.02	1.06

Note. (ref:tab-example-resp-note)

Appendix B

Technical Details

The algorithm is divided in three parts: Preprocessing, classification, and postprocessing.

Input: x-coordinate, y-coordinate, timestamp, unit of coordinates (px vs. deg), screen resolution, screen dimensions, distance to screen, sampling rate, blink indicator, order of SG filter, length of SG filter, number of states in the HMM, start values for the HMM, downsampling factor, random response start values, seed for random response start values, control options for fitting the HMM, minimum saccade duration.

Preprocessing

(1) The algorithm checks whether samples are valid: Samples for which the x- or y-coordinates are NA, infinity, or (0,0) are marked (internally) as invalid.

(2) Samples that match the blink indicator are labeled as blinks.

(3) If the unit is px, the x- and y-coordinates are transformed into va with:

   i. The coordinates are centered ([0,0] is center)

   ii. $\theta_{deg} = atan(\frac{x \cdot res_x}{d_{screen} \cdot dim_x})\frac{180}{\pi}$

(4) Samples that are outside the screen are marked as invalid.

(5) Velocities and accelerations for the x- and y-coordiantes are calculated using an SG filter (3rd order, length = 5).

(6) Combined velocitiy and acceleration are calculated as: $v = f\sqrt{v_x^2 + v_y^2}$ and $a = f\sqrt{a_x^2 + a_y^2}$, with f as the sampling rate. Note that acceleration is only multiplied by the sampling rate once. The second multiplication is done during postprocessing.

(7) Velocity and acceleration of value zero are nudged to be 0.01.

(8) Samples with velocity above 1000 deg/s or acceleration above 100 deg/s are marked as invalid.

(9) The sample-to-sample angle is calculated as:

$$\theta_t = atan2(\frac{y_{t+1} - y_t}{x_{t+1} - x_t}) - atan2(\frac{y_t - y_{t-1}}{x_t - x_{t-1}}).$$

Angles are normalized to $[0, 2\pi]$ by adding 2π to negative values.

(10) Samples that have an angle with value NA are marked as invalid.

(11) The velocity, acceleration, and angle of invalid samples are set to NA.

Output: Data frame with x-coordinate, y-coordinate, timestamp, velocity, acceleration, angle, and initial label variable with blinks.

Classifcation

Input: Data frame from preprocessing and HMM related general input variables.

(12) Sets default start values if none are provided.

(13) Downsamples velocity and acceleration.

(14) Fits a HMM with velocity, acceleration, and angle as dependent variables.
Velocity and acceleration are modeled by a two-parameter gamma distribution (shape and scale). Angle is modeled by a uniform distribution for the first component and von-Mises distributions (mean and kappa) for further components. Initial state probabilities are optimized on the identity scale and transition probabilities on the multinomial logit scale. All response parameters except for the mean parameters of von-Mises distributions are optimized on the log-scale.

Output: Fitted model.

Postprocessing

Input: Data frame from preprocessing and posterior state sequence of the fitted model.

(15) Relabels posterior states of samples with recursive algorithm:

  i. If a fixation or smooth pursuit sample has no previous or following sample with the same label (single sample event), it is relabeled as the label from the previous sample.
  ii. If a PSO sample is followed by a saccade sample or following a non-saccade sample, it is relabeled as the last different label.
  iii. Saccade events (batch of consecutive saccade samples) below the minimum saccade duration (in s) are relabeled as the label from the previous event.
  iv. These steps are applied recursively until all conditions are met.

(16) For each event (batch of subsequent valid samples belonging to the same label), the following metrics are computed:

  i. Fixations:
    a. Duration: Max. timestamp - min. timestamp
    b. Position: Trimmed mean (20%) of x- and y-coordinates
  ii. Saccades, PSOs, and smooth pursuits:
    a. Duration: Max. timestamp - min. timestamp
    b. Start: First x- and y- coordinates
    c. End: Last x- and y- coordinates
    d. Amplitude: Euclidian distance from start to end
    e. Max. velocity
    f. Avg. velocity
    g. Max. acceleration $\times f$
    h. Avg. acceleration $\times f$
    i. Direction: $\theta = atan2(\frac{end_y - start_y}{end_x - start_x})$

General output: List with a data frame containing the labeled samples, a list with data frames for the different events, and the fitted model.