Model-based Evaluation of Recall-based Interaction Techniques

Julien Gori
julien.gori@sorbonne-universite.fr
Sorbonne Université, CNRS, ISIR
Paris, France

Bruno Fruchard
bruno.fruchard@inria.fr
Univ. Lille, Inria, CNRS, Centrale Lille,
UMR 9189 CRIStAL
Lille, France

Gilles Bailly
gilles.bailly@sorbonne-universite.fr
Sorbonne Université, CNRS, ISIR
Paris, France

ABSTRACT

This article tackles two challenges of the empirical evaluation of interaction techniques that rely on user memory, such as hotkeys, here coined Recall-based interaction techniques (RBITs): (1) the lack of guidance to design the associated study protocols, and (2) the difficulty of comparing evaluations performed with different protocols. To address these challenges, we propose a model-based evaluation of RBITs. This approach relies on a computational model of human memory to (1) predict the informativeness of a particular protocol through the variance of the estimated parameters (Fisher Information) (2) compare RBITs recall performance based on the inferred parameters rather than behavioral statistics, which has the advantage of being independent of the study protocol. We also release a Python library implementing our approach to aid researchers in producing more robust and meaningful comparisons of RBITs.

CCS CONCEPTS

• **Human-centered computing → HCI theory, concepts and models**; **Empirical studies in HCI**; *Gestural input*.

KEYWORDS

memory, interaction technique, model-based evaluation, maximum likelihood, experimental design

ACM Reference Format:
Julien Gori, Bruno Fruchard, and Gilles Bailly. 2024. Model-based Evaluation of Recall-based Interaction Techniques. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3613904.3642637

1 INTRODUCTION

Recall-based interaction techniques (RBITs) are interaction techniques that rely on user memory, like keyboard and gesture shortcuts *e.g.*, Marking-menus [26], Octopocus [5], MarkPad [16] *etc.* RBITs are generally faster than interaction techniques relying on *recognition* such as menus or toolbars. Moreover, they do not use screen space, letting users focus on their primary task. Learning RBITs takes time and effort, which limits their acceptability by users as well as their overall efficiency. Comparing RBITs on the basis of their memorization is therefore important.

Such comparisons are usually conducted through a recall-based protocol, which may combine several training and test phases. The measure of performance that serves as the basis for comparison is the recall percentage during test phases, which measures the ability of the participants to recall the command-action mapping when no assistance is provided. However, comparing RBITs with these recall-based protocols is more difficult than it appears: (1) the recall percentage depends not only on the RBITs themselves but also on the training *schedule*, *i.e.*, when and how many times commands were presented to the participant [31]. This makes interpreting recall rates difficult across schedules. Another, more subtle risk, which we demonstrate for the first time in this work, is that observing a significant difference in recall percentage between two RBITs may be attributed to differences in interaction time with the technique, rather than actual differences in recall performance (see the "distortion problem" in subsection 3.2). (2) constructing schedules in the first place is difficult, and many experimenters create schedules by "trial and error", *i.e.*, running pilot studies until recall percentage are considered to lie in a "correct" range. Intuitively, some schedules are better than others; schedules which systematically lead to very high (*e.g.*, due to too many repetitions during the training phase) or very low recall scores (*e.g.*, due to a too short training phase and/or with too many commands to learn) are not useful, since they do not show differences between recall rates, but it remains an open question in HCI research to determine which schedules are best at estimating RBIT performances.

Our primary contribution is to suggest model-based evaluation as a method to evaluate RBITs that does not rely on the actual schedule. A model-based evaluation assumes that the observed recall data can be described by a memory model and its "true" parameters, these parameters serving as the basis to summarize and compare RBIT performance. The key advantage of performing the comparison in the parameter space is that the latter are *independent* of the schedules. In particular, we show in section 3 how the model-based evaluation addresses two problems (including the aforementioned distortion problem) that affect the traditional approach based on comparing recall rates.

The set of true parameters describing an RBIT can *not* be observed directly; rather, they have to be inferred from observed recall data given a model of human memory. In this work, we use the exponential forgetting (EF) model as model of human memory and the method of Maximum Likelihood (ML) as inference method. We validate our ML implementation of the EF model with extra scrutiny on correctly estimating the uncertainty about the parameter values to ensure meaningful comparisons in the parameter space.

Our second contribution is to provide a method to determine the most efficient schedule when designing an RBIT experiment *i.e.*, the one which minimizes the uncertainty about the "true" parameter values of the RBITs. Our method is based on maximizing the Fisher information provided by the schedule, which quantifies the cumulative information provided by each trial of the experiment. We namely show via several simulations that Fisher information is an appropriate value function to rank schedules, both for comparisons in the space of model parameters or using recall rates.

Our third contribution is `pyrbit`, a Python library that implements all methods presented in this work; researchers can namely use it to assess the informativeness of an arbitrary schedule they would have designed, and can leverage visualizations and various utilities for simulations.

All code used for this work is publicly available, either from the library directly, or on a GitHub repository[1].

2 BACKGROUND

In this section we lay the basis of the classical and model-based evaluations.

2.1 Paired-Associated Learning and RBITs

Paired-associate learning (PAL) is a classic memory paradigm that is used to understand how people encode and retrieve newly formed associations by having them remember pairs of items; in the RBIT context a pair is a command associated with its trigger which may be gestures [2, 17, 18], shortcuts [19, 21, 43] *etc.* PAL has received significant interest from psychologists [34]; It is well known that two major factors in PAL performance are delay (how much time passes between stimuli presentation), and repetition (how many times stimuli are repeated) [31]. Therefore, the *schedule i.e.*, the sequence that specifies which pair to present at what time, has a major impact on the results whether measured in differences in observed recall rates, savings [31] *etc.*

In a typical study, participants have to learn about 10 to 20 arbitrary pairs, (8 in [18, 21], 12 in [2], 16 in [18] or up to 48 in [35]). Experiments may alternate between learning phases (or blocks), where pairs are presented to the participant, and testing phases, where the command is queried to the participant which should respond with the right trigger. In some experiments, the correct trigger is recalled to the participant at the end of each trial where they failed to provide it, but this is not always the case. PAL experiments may generally last several days [31], but in HCI experiments are usually conducted over one or two days. For example, Fruchard *et al.* [17] alternate learning and test blocks (one after the other) over two days and the correct trigger is never recalled to the participant when they do not remember it; in contrast, Perrault *et al.* [35] conduct a one-day experiment with an initial block used to map items to commands, followed by four test blocks, where the correct trigger is always recalled during the test blocks. Experimenters will then compute the percentage of recall for each testing block, which is the score used for comparing RBITs (potentially together with other indicators such as mean execution time).

Not much guidance exists to construct these schedules, and as a result the variability in protocols is large. From our experience, many experiments on RBITs have been designed by copying existing studies and adjusting them after a pilot study, which may include several trials and errors. A model-based evaluation solves these issues, since it levels out the effect of the schedule.

[1]https://github.com/jgori-ouistiti/chi24-rbit

2.2 The Exponential Forgetting (EF) model

The exponential forgetting (EF) model has been described and used before in HCI [33] and follows from Ebbinghaus' forgetting curve [31]. According to this model, the probability p for a subject to correctly recall a command is given by

$$p = \exp\left(-\alpha(1-\beta)^k \Delta t\right), \quad \alpha \in \mathbb{R}^+, \beta \in [0,1], \tag{1}$$

where k is the number of times the command was previously seen by the subject (repetition number) and Δ_t is the time that has elapsed since the last time that command was seen by the subject (delay). α and β are two "memory" parameters; α (positive) is the initial forgetting rate: the lower α, the slower the forgetting and the better the recall. This rate gets reduced by $(1-\beta)$ (between 0 and 1) with each repetition which further slows the rate of forgetting, and thus the higher the β, the better the recall. While β is unitless, the unit for α depends on the time basis. In this work, we use standard units *i.e.*, Δ_t is expressed in seconds and α is expressed in s^{-1}.

There exist many other models that have been used to describe recall performance in PAL, such as various flavours of ACT-R models [1, 34], the predictive performance equation [45], various power models [31] *etc.* All these models are valid alternative candidates for a model-based approach (subsection 6.1 illustrates how the method generalizes to an ACT-R model), but we chose EF primarily because, having only two parameters, it is possible to illustrate uncertainties with two-dimensional confidence ellipses, contrary to models with more parameters.

2.3 Using a human memory model to model the interaction between a user and a device

It might seem surprising to use a human memory (or any other cognitive) model to describe the interaction between a user and an RBIT. The memory model used here should be viewed as a mathematical *description of observed behavior* rather than one of how the human brain functions: the model simply captures the important effects of delay and repetition when a user interacts with an RBIT. Perhaps, the RBIT's design makes it so that the presented commands are more easily forgotten, and this will be captured by a larger α; the model is thus used to estimate effect sizes of repetition and delay.[2]

The well-known Fitts' law evaluations [20, 40] work under a similar assumption. Fitts' law is also a model-based evaluation: movement time is aggregated into two parameters (a and b), following a motor control model due to Fitts [15]. Yet, Fitts' model has been used countless times to evaluate devices, including some cases of indirect comparisons [40], with the understanding that Fitts' law in that case characterizes the entire perceptual motor loop, and that a and b measure the size of the effect of index of difficulty on movement time [20].

[2]Models used for evaluation (*i.e.*, estimation of effect sizes) do not necessarily have to score high on prediction (*i.e.*, prevision of outcomes) to be useful, but instead must prove to have good identifiability of parameters [39]. Hence, simple models with few parameters are generally preferable.

3 MODEL-BASED EVALUATION AND MOTIVATIONS

A model-based evaluation assumes that observed data can be described by a parametric model, and its "true" parameters. These parameters can be inferred from the observed data, and serve as the basis to summarize and compare performance — in our case that of RBITs. The advantage of performing the comparison in the parameter space is that the "true" parameter values are *independent* of the schedules. In this section, we provide two simulated RBIT comparisons, that illustrate weaknesses of comparisons based on recall rates and advantages of the model-based evaluation. The implementation and validation of the model-based evaluation is delayed to the next section.

3.1 How model-based Evaluations Facilitate Indirect Comparisons

Scenario. We simulate two user studies: *Simulated Experiment 1*, comparing RBIT A(1) with RBIT C(1) and *Simulated Experiment 2* comparing RBIT B(2) with the same RBIT C(2). The two study protocols are identical and consist of a two-day experiment with the same number of learning and testing blocks as in [17]. They differ only in the pauses in the schedule between blocks, as well as trial durations. The exact simulation parameters are detailed in Appendix B. The simulated recall rates for both experiments are displayed in the left and middle panels of Figure 1.

It is visually clear that in Experiment 1, A(1) outperforms C(1) and that in Experiment 2, B(2) outperforms C(2) — computations of statistical significance are not needed. However it is less clear

(1) What the relative performance (recall-wise) of A(1) and B(2) is. This illustrates that indirect comparisons may be difficult to make. Indirect comparisons are important because they are the basis for meta-analyses (quantitative or not), which allows researchers to reliably aggregate existing research outcomes into a consistent whole.
(2) How close the evaluations C(1) and C(2) actually are recall-wise. This is an important question for determining the internal validity of an experiment; in particular, C might be a baseline in which case one would want to judge whether that baseline was reasonably implemented.[3]

Solution provided by the model-based approach: In the model-based approach, the comparison between C(1), C(2), A(1) and B(2) is made in the parameter space of the EF model, see the right panel of Figure 1, where the estimated parameter values and associated 95% confidence ellipses (CE) are shown. It is clear that:

(1) A(1) is statistically significantly different (ssd, $p < 0.05$) from B(2), C(1) and C(2), since the CEs do not overlap;
(2) B(2) is ssd from C(1) and C(2); since the CEs do not overlap
(3) C(1) is not ssd from C(2); since the CEs have major overlap.
(4) Further, we can rank the RBITs, as A ouperforms B which outperforms C, given that they have similar values of estimated $\hat{\beta}$ but that estimated $\hat{\alpha}_A \simeq 10^{-1.8} < \hat{\alpha}_B \simeq 10^{-1.7} < \hat{\alpha}_C \simeq 10^{-1.6}$.
(5) We can quantify by how much A outperforms B and C. With the EF model, for any k and Δ_t, the probability of recall for RBIT A is $p_A = \exp(-\hat{\alpha}_A(1-\hat{\beta})^k\Delta_t)$ and for RBIT B $p_B = \exp(-\hat{\alpha}_B(1-\hat{\beta})^k\Delta_t)$, which implies $p_A = p_B^{\frac{\hat{\alpha}_A}{\hat{\alpha}_B}}$ (this assumes equal values of $\hat{\beta}$). With a log difference of 0.1 as in our experiment, this gives $p_A = p_B^{10^{-0.1}} \simeq p_B^{0.8}$; For example if $p_B = 0.5$, then $p_A = 0.5^{0.8} \simeq 0.57$, if evaluated on the same schedule, on average. Note that RBITs A and B could also be compared through predictive simulations based on $\hat{\alpha}_A$ and $\hat{\alpha}_B$.

[3]There is little incentive for an author to come up with a strong baseline — quite the contrary, actually. For example, the parameters of a baseline could not be finetuned well enough. [12] provides a discussion about weak baselines in a different field but which remains relevant here. Thus, having the ability to evaluate the performance of the same baseline in two different studies is important.

3.2 How Model-based Evaluations Safeguards from Different Execution Times

In this scenario, we pretend a single user study compared RBIT A with RBIT B, which, unbeknownst to the researcher, are strictly identical when it comes to recall *i.e.*, data is generated by the EF model with the same parameters for both RBITs. While the studies share exactly the same protocol, the execution time for each RBIT differ: users operate B faster than A. This is quite a common issue with RBITs, especially in studies that compare a baseline that is easy to operate, with a more advanced technique which might need some adjusting before reaching efficient execution times. For example, Appert and Zhai [2], and Fruchard *et al.* [17] report differences in execution time of about 20% in their compared RBITs.

The simulated recall rates for this experiment are displayed in the left panel of Figure 2. We see that in the three recall blocks, the conditions A and B are ssd (respectively $p < 1e^{-9}$, $p < 1e^{-5}$ and $p < 3e^{-3}$), and the effect size is sizeable (respectively an absolute difference of about 20, 10 and 5%). Here, the experimenter would likely consider that B outperforms A recall-wise.

Solution provided by the model-based approach: In the model-based approach, the comparison between A and B is again made in the parameter space of the model considered for the evaluation, see the right panel of Figure 2, where the estimated parameter values and associated 95% confidence ellipses are given. The considerable overlap of both confidence ellipses tells us that the model-based approach finds no significant difference, and estimates a very small effect size, in line with the scenario. The mechanism at cause here is that different execution times will lead to different delays in between repeated items. Yet, delay is a very important predictor for recall probability, and perhaps unexpectedly, it can lead to great differences in block recall rates as these delays accumulate, especially in early blocks. We coin this effect a *distortion*, since the recall-based comparison is distorted due to differences in the execution times of the RBITs. A point could be made that execution time is inherent to a technique. However, usually HCI studies on RBITs independently assess execution time, so it feels preferable to separate the two effects of recall and execution time, especially considering that often the execution time for new techniques are transiently higher than for well practiced baselines [48].

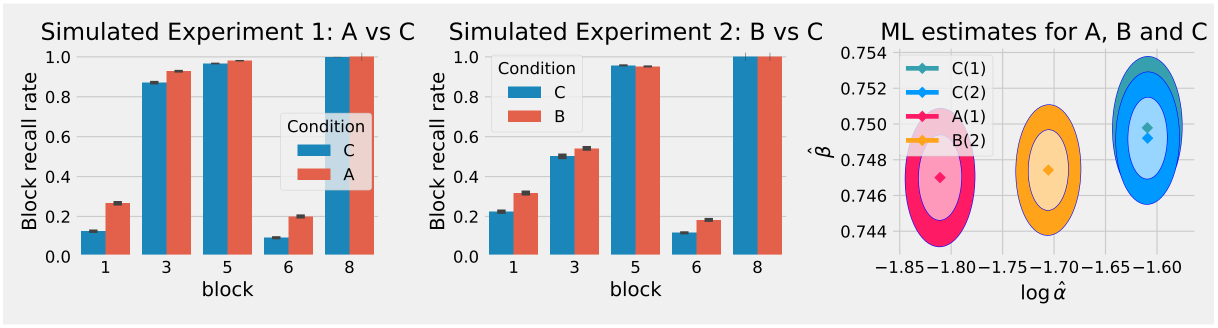

Figure 1: The difficulty of making indirect comparisons when working with recall rates for studies with different schedules disappears when working with the model-based evaluation. Left panel: A simulated experiment comparing two RBITs A and C. A visibly outperforms C. Middle panel: A second simulated experiment comparing two RBITs B and C. B visibly outperforms C, yet we can't say whether A outperforms B and whether C in both experiments have the same performance: indirect comparisons are difficult to conduct. Right panel: ML estimates for A, B, and C in both experiments. We see that A outperforms B which outperforms C. C for both experiments have the same performance. Indirect comparisons are much simpler. Parameters of the simulation are detailed subsection B.1.

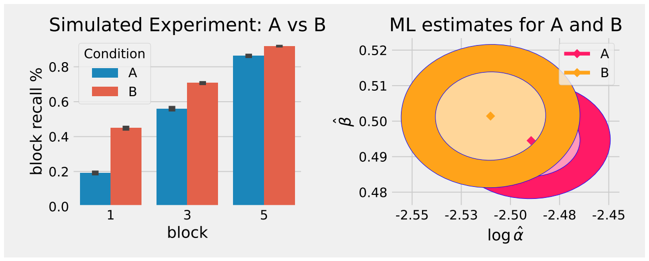

Figure 2: The distortion that results from different execution times in the evaluation using recall rates does not occur in the model-based evaluation. Left: block recall rates. The two RBITs A and B have different recall rates; that difference is statistically significant. Right: The ML estimates for the same two RBITs. The ML confidence ellipses have a big overlap, indicating, that in reality, there is no difference between the two RBITs. Parameters of the simulation are detailed subsection B.2.

4 HOW TO ESTIMATE RBIT PARAMETERS: THEORY, IMPLEMENTATION, VALIDATION

We have just illustrated how model-based evaluations in theory facilitate meaningful direct and indirect comparisons. However, the reliability of these comparisons in the parameter space depends in practice on how well one can estimate the parameters of the EF model and their associated uncertainty. We now describe the method of the maximum likelihood (ML) that we used for estimation of EF's parameters, as well as our implementation and validation of it.

4.1 Parameter Estimation from Recall Sequences using the Method of Maximum Likelihood (ML)

The method of Maximum Likelihood (ML) is a classical tool, whose aim is to determine the model's parameter values that make the observed outcome the most likely.[4] More formally, let $\omega_1, \omega_2, \ldots, \omega_n$ be a sequence of observed recalls for a given command and RBIT. Then the probability of that sequence, given the model $\mathcal{M}$, its parameters $\theta = (\alpha, \beta)$, and the history h_i of pair presentation times and participant responses up to index $(i-1)$ is called the likelihood of the sequence and is given by $l(\theta) = \prod_i p(\omega_i|\mathcal{M}, \theta, h_i)$. The ML estimate of θ is the parameter $\hat{\theta}$ that maximizes the likelihood of the recall data: $\hat{\theta} = \arg\max_\theta \prod_i p(\omega_i|\mathcal{M}, \theta, h_i)$. The method of ML thus boils down to expressing the (log)[5]likelihood of the recall sequence and maximizing it, *e.g.*, using an off-the-shelf numerical optimization algorithm.

4.2 Quantifying the uncertainty of the ML estimate

The method of ML is leveraged to produce a *point estimate*, for example $\hat{\theta} = (\hat{\alpha}, \hat{\beta}) = (0.01, 0.4)$. However, without an associated uncertainty, meaningful comparisons are hard to make. Fortunately, ML also provides the appropriate theory to express the uncertainty of that point estimate. Indeed, under mild conditions of regularity of the likelihood function, the ML estimator is asymptotically (for large enough sample size) normal *i.e.*, the estimated values are normally distributed ($\mathcal{N}$) centered on the true value θ with covariance matrix

[4]We chose ML because of its asymptotic properties (unbiased, consistent, normal) and general applicability. It is the "best" estimator in the sense that it asymptotically reaches the Cramer-Rao bound: for a large sample size, there exists no other frequentist estimation method that is more precise *i.e.*, has tighter confidence intervals. For a basic introduction, we direct the reader to a tutorial on ML for psychological sciences [32].
[5]It is often easier to work with the log-likelihood which transforms the product in the likelihood into a sum. Because log is a strictly increasing function, it is equivalent to maximize the likelihood or its logarithm.

$\mathcal{J}(\hat{\theta})^{-1}$:

$$\hat{\theta} \sim \mathcal{N}(\theta, \mathcal{J}(\hat{\theta})^{-1}). \tag{2}$$

The inverse of the covariance matrix $\mathcal{J}(\hat{\theta})$ is called the observed information, and it is defined as minus the Hessian ($\nabla\nabla^T$) of the likelihood, evaluated at $\hat{\theta}$:

$$\mathcal{J}(\hat{\theta}) = -\nabla\nabla^T l(\theta)_{|\theta=\hat{\theta}}. \tag{3}$$

Closed form asymptotic expressions for confidence intervals (CI) and ellipses (CE) can be derived from the normal distribution. For example, a $(1-\eta)\%$ CI for θ_i, the i^{th} component of θ, is given by $[\hat{\theta}_i \pm \Phi^{-1}(1-\eta/2)\sqrt{(\mathcal{J}(\hat{\theta})^{-1}{}_{ii})}]$, where $\Phi(x)$ is the standard normal distribution's cumulative density function.[6] This example CI shows how to interpret observed information: the "larger" it is, the more narrow the CIs and thus the more "information" it provides about the parameter. We will return to the observed information in section 5 where it serves as the basis by which we evaluate schedules.

The value of α in the EF model is typically between $1e^{-5}$ and $1e^{-1}$ for PAL tasks, which is better expressed on a logarithmic scale. We applied a log-transform on α as explained in the supplementary material, and computed CIs using the so-called delta method. As we show subsection 4.4, this log-transform leads to a better coverage of the computed CIs, and correspondingly in this work we report parameters in the $(\log_{10}\alpha, \beta)$ space.

4.3 Implementation of the ML Estimator

We numerically computed the log-likelihood function for an arbitrary sequence of recalls and associated history. To maximize it, we used the L-BFGS-B algorithm as implemented by the SciPy [44] `optimize` library. CIs and CEs were computed by plugging in the observed information Equation 3 in the CI formula resulting from the normal approximation Equation 2; the delta method was used to compute these in the $(\log\alpha, \beta)$ space. The (log) likelihood of the model, as well as the derivations of the gradients and Hessians needed to express the observed information, and thus computing the CIs and CEs are provided in the supplementary materials. Calculations were verified with symbolic calculus, using SymPy [29].[7]

4.4 How well do the estimated parameters reflect the true value of parameters?

In HCI, and following recommendations from cognitive science [47], the validation of estimation methods (ML included) is sometimes known as parameter recovery [36], where the goal is to show through a simulation study that the estimated parameters correlate strongly with the true value of the parameters used for the simulation. These correlations are given in Figure 3, which confirms high Pearson correlations (ρ) for both parameters in the ranges of interest. However, correlation alone is a measure that is too coarse to validate estimated parameters intended for comparisons between RBITs [8]. To evaluate the results of the ML estimates more finely, we also assessed its bias and standard deviation, and coverage of the CIs, for the pair of true values ($\alpha = 0.01, \beta = 0.4$) for increasing sample sizes (N). To do so, and for $R = 1000$ iterations, we simulated recall data with an EF memory model, estimated the parameters, after which we computed the aforementioned statistics. Because of the memory model, increasing the sample size is not as straightforward as it seems: if one naively selects a schedule of size N, most likely the recalls will only be `False` at the start and become all `True` after some time and both will bring no information, and hence will almost not increase the precision of the estimates as discussed before. Hence, for each sample size N and for each simulation, we randomly drew N pairs (k, Δ_t) from a uniform distribution that were fed to the EF model. This schedule only makes sense for simulations, see section 5.

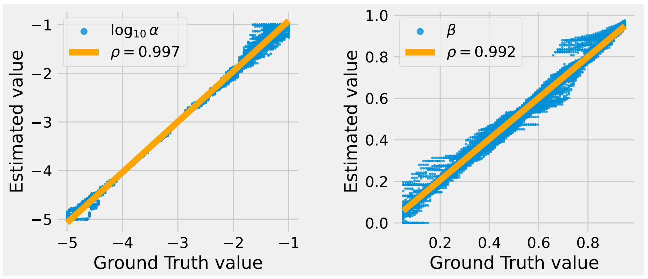

Figure 3: Pearson correlation (ρ) between true and estimated values for 100×100 pairs of fully crossed and uniformly spaced values for $\log_{10}\alpha \in [-5,-1]$ and $\beta \in [.05,.95]$. Left: $\log_{10}\alpha$, Right: β. These parameters were estimated for a fixed, reasonable schedule. The schedule becomes less suited for values of α away from its average, as the estimation becomes less precise. Note that the correlation was tested on ground truth values which include "extreme" model values *e.g.*, very low values of α paired with high values of β that lead to very high recall probabilities, and conversely, high values of α with low values of β that lead to very low recall probabilities.

Bias and standard deviation. To estimate bias (mean error to the true value) and standard deviation (spread around the mean estimated value) of the ML estimates, we computed the average difference to the true value as well as the standard deviation of the estimated parameters. Figure 4 shows that the bias and standard deviation decrease steadily in the left and middle panels. This is expected, since the ML estimate is asymptotically unbiased and consistent. The quasi linear decrease of the bias is due to the asymptotic bias $b(\theta)$ of the ML estimate being proportional to $1/n$ as a first order approximation [13], and thus $\log b(\theta)$ is approximately proportional to $-\log(n)$. These results ensure that our implemented ML method converges to the true values; Additionally they can be

[6]The general formula for confidence ellipsoids is a multivariate generalization of this formula, see *e.g.* [23].

[7]These calculations can be inspected in the source code of the library.

[8]For example, let's imagine a comparison between two RBITs, with estimates $\hat{\beta}_1 = 0.8$ and $\hat{\beta}_2 = 0.85$. Now, compare the assertion "for sample size $N = 500$, the computed estimates correlate with the true values with $\rho = 0.95$" with the other assertion "for sample size $N = 500$, for a truth value of $\beta = 0.8$, the estimator has a bias of 0.01, and the 95% confidence interval, which is known to have adequate coverage for this parameter is $[\hat{\beta} - 0.02, \hat{\beta} + 0.02]$". In the latter we can conclude that the two RBITs are significantly different (at the 0.05 level), whereas in the former we can say little about the comparison.

exploited to determine the sample size needed to get a certain reliability on the estimates (which achieves the same goal as a power analysis).

Coverage of the Confidence Intervals. The quality of CIs and CEs is crucial to assess if observed differences between RBITs in parameter space are meaningful; for example CIs that are too narrow may mean standard errors are underestimated which in turns result in inflated type I errors. We computed CIs directly from the normal approximation Equation 2 and validated them by estimating their coverage. The coverage is the probability that the CI will include the true value of the parameter of interest, thus theoretically the coverage of a 95% CI is 0.95. The CIs depend on the Hessian of the likelihood via the observed information Equation 3, which we computed in three different ways:

(1) the *theoretical* Hessian as derived in the supplementary materials (Coverage H);
(2) the Hessian that is *numerically evaluated* with the BFGS algorithm[9], hereafter called Coverage BFGS;
(3) the log-α Hessian, as described in subsection 4.2 and the supplementary materials (Coverage log-H).

We found that somewhat unexpectedly, the coverage of the CI with numerically computed Hessian (Coverage-BFGS) is not good, and more importantly, does not seem to improve with increasing N. From a practical perspective, this highlights benefits of using analytic formulas for the Hessian, which perform much better, as both analytical CIs end up very close to the nominal coverage of 95%. The CI computed from the log-transformed Hessian also appears to outperform the regular one (coverage H), especially for small sample size. We suspect this is because the α parameter spans several orders of magnitude in linear scale. These results explains why we evaluate RBITs in the $(\log \alpha, \beta)$ space.[10]

To summarize, this simulation study validates our implementation of the ML method applied to the EF model by focusing on several metrics and observing how they improve with sample size (N). It also highlights the benefits of using an analytically computed Hessian, and confirms the asymptotically normal assumption leads to reasonable CIs, even for sample sizes as small as $N = 100$.

4.5 Case Study: Re-analyzing data from [17]

We now illustrate how the proposed method can be applied to a realistic dataset; We used data from the experiment by Fruchard *et al.* [17], which compares two RBITs: Marking Menu and BodyLoci. The study summary and some operational details are delayed to Appendix A; in short, the difficulties are in recreating the timing information and repetitions needed to apply the EF model which were nor properly recorded. Then, based on the repetition, delay and recall information reconstructed from [17], we fitted the EF model for each participant. The result is given in the two top panels of Figure 5. We also aggregated this data in two ways to get a global evaluation of the two RBITs, displayed in the two bottom panels of Figure 5:

(1) We computed a smooth probability distribution on the parameter space via a kernel density estimation (KDE) computed on the parameters of all participants for each technique. This is represented by the blue gradient in the two bottom panels of Figure 5.
(2) We computed the parameter estimates of the two techniques by aggregating all participant data as if they were from one participant only, which represents the parameters of an "average memory model" of the population and RBIT. The associated CEs are displayed on top of the KDE in the two bottom panels of Figure 5, and a direct comparisons between the two CEs is available in Figure 16.

The results are in line with the findings in the original study, and find no differences between the two techniques (see Figure 16). This suggests that the significant difference obtained for the first recall block in the original study [17] is a result of the distortion due to the different execution times as explained in subsection 3.2. The analysis also highlights the very high variability between participants, even though these participants were chosen from a homogeneous pool.

5 MODEL-BASED COMPARISON OF SCHEDULE DESIGNS

Not all participant responses are equally "informative" in a colloquial sense: for example once an association is well learned, participants will systematically recall the right command. Conversely, the first participant responses may all be incorrect before memory is sufficiently formed. The schedule thus plays an important role in the identification of the model's parameters, and choosing an adapted schedule will result in shorter experiments and/or more powerful ones (in the sense of reducing type II errors). However, there are many constraints when it comes to building a schedule: selecting the number of pairs to be learned, determining how many times pairs are repeated per block and whether their distribution should be uniform or Zipfian [27**?**], what the durations of pauses are between blocks, on how many days to conduct the experiment *etc.*

Model-based evaluation can be used to assist the experimenter in the design of the schedule. More precisely, in this section we show that we can analytically compare two schedules in their capacity to discriminate the recall performance of two RBITs *before* conducting the empirical study. Ultimately, by comparing possible schedules of interest, or via constrained optimization [6], a researcher can identify the schedule that best discriminates two RBITs conditional on their constraints.

5.1 Measuring the informativeness of a schedule

Consider two potential schedules on which to compare the RBITs. We say one schedule is better than the other if the estimated CEs are more distinct (narrower and more distant). Since CEs are directly constructed from the observed information (see subsection 4.2), we

[9]The Hessian can be evaluated by finite difference equations, but it is also numerically evaluated by so-called Newton algorithms such as BFGS, which can be used in optimizing the likelihood. For example, the ML module of statsmodels [38] uses BFGS to simultaneously optimize the likelihood and estimate the inverse of the Hessian.

[10]In addition, the CIs computed from the Hessian directly will sometimes include negative values for α, which by definition should be impossible. This problem does not occur with the log transformed Hessian, which is another reason for using it.

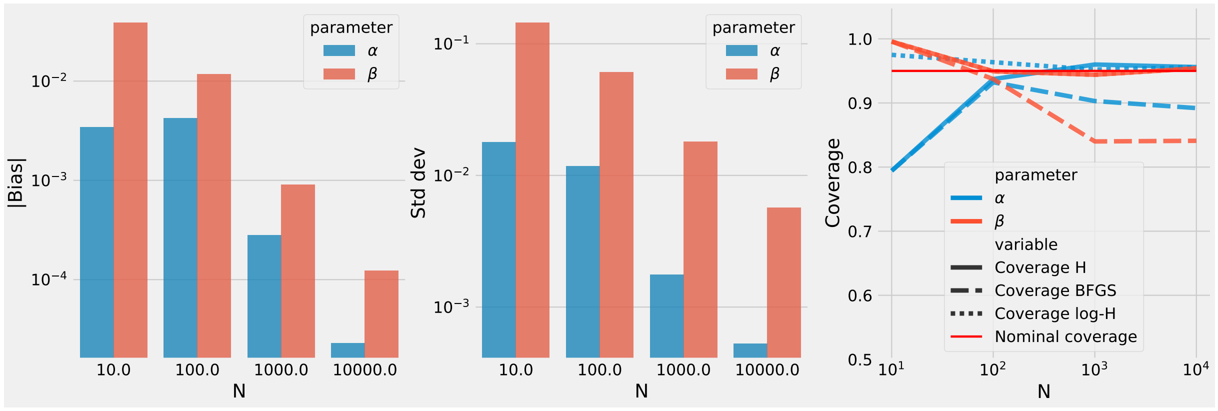

Figure 4: ML estimation for increasing sample size N. Left and middle panels: bias and standard deviation of the estimates in log scale. Right panel: coverage of the CIs constructed with the analytical Hessian (Coverage H), the log transformed analytical Hessian (Coverage log-H) and the numerical Hessian (Coverage BFGS). Simulation parameters detailed subsection B.3

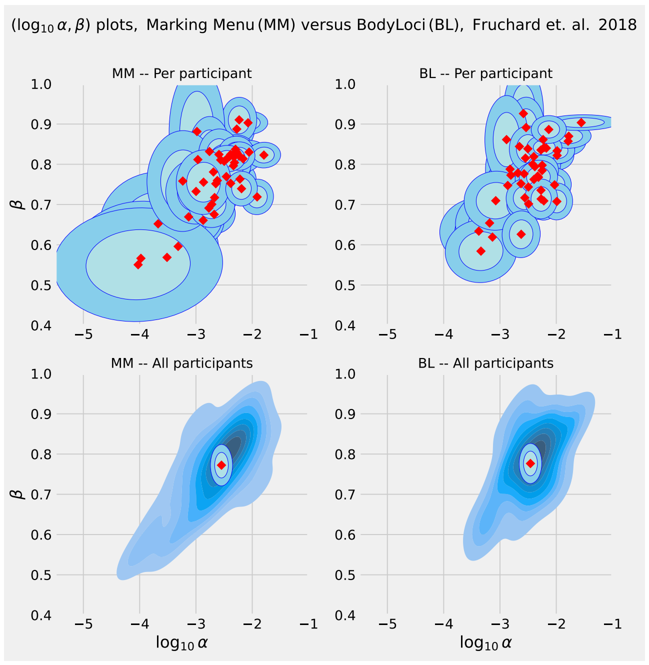

Figure 5: BodyLoci versus marking menus in the parameter space of the EF model. The top panels give the estimated parameters with their associated confidence ellipses for each participant for marking menus (left) and BodyLoci (right). Below: a kernel density estimation (blue gradients) computed on the parameters of all participants. Superimposed is the estimate of the parameters of the EF model when all participants are assumed to have the same parameters.

suggest using the latter to rank schedules: larger observed information means narrower CEs and thus, a better schedule.

Observed information (see Equation 3) quantifies the information provided by one recall outcome about the unknown parameters of the model [11][11]. To get a measure that characterizes the schedule, we need to average information from multiple outcomes obtained with the same schedule. Thus, to evaluate a given schedule with N trials, we simulated R recall experiments with this schedule. The observed information associated with the nth trial and the rth simulation is $J_{n,r}$. We then define the following three aggregate information metrics (see Figure 6 for an illustration):

- the sequence observed information $\mathcal{J}_{:n,r}$, which is the observed information computed for the sequence of recalls until n at repetition r. This information is just the sum of the individual observed informations $\mathcal{J}_{:n,r} = \sum_{i=1}^{n} J_{i,r}$ for each trial i and is the one used to construct the confidence intervals in section 4.
- the Fisher[12] information $< \mathcal{J} >_n$, defined as the average observed information for a particular trial n: $< \mathcal{J} >_n = \frac{1}{R}\sum_{r=1}^{R} J_{n,r}$. This information tells us how much information one particular trial of the schedule provides on average *i.e.*, identifies which trials are informative and which aren't.
- the sequence Fisher information $< \mathcal{J} >_{:n}$, the average observed information for the sequence until the n-th trial. It equals the sum of the n Fisher information, or, equivalently, the average sequence observed information over all repetitions: $< \mathcal{J} >_{:n} = \sum_{i=1}^{n} < \mathcal{J} >_i = \frac{1}{R}\sum_{r=1}^{R} J_{:n,r}$. This information estimates how information is accumulated by the schedule; it is thus the one we suggest for ranking schedules.

[11]For an intuition as to why this may be true: One can approximate the likelihood function via Taylor's theorem around the maximum: up to the second order we have $l(\hat{\theta} + h) = l(\hat{\theta}) + \nabla l(\hat{\theta})h + 1/2h^T\nabla\nabla^T l(\hat{\theta})h + o(\|h\|^2)$. By definition of $\hat{\theta}$ as value that maximizes $l(\theta)$, the gradient term vanishes $\nabla l(\hat{\theta})h \simeq 0$, which shows the likelihood to be a parabola whose flatness is given by the Hessian of the likelihood, and the flatter the likelihood, the lesser the certainty we may have about that estimate. We refer the reader to textbooks on statistics for more details.

[12]Technically, Fisher information [14] is the expected value of the observed information, see [28] for a tutorial on Fisher information for the psychological sciences. Both Fisher and observed information are used interchangeably in the statistical literature depending on which one is the easiest to compute [7], with conflicting results about which type of information actually leads to the most precise estimates [7, 11]. Because we compute averages of observed information, we call these Fisher information, even though the correspondence is only valid asymptotically.

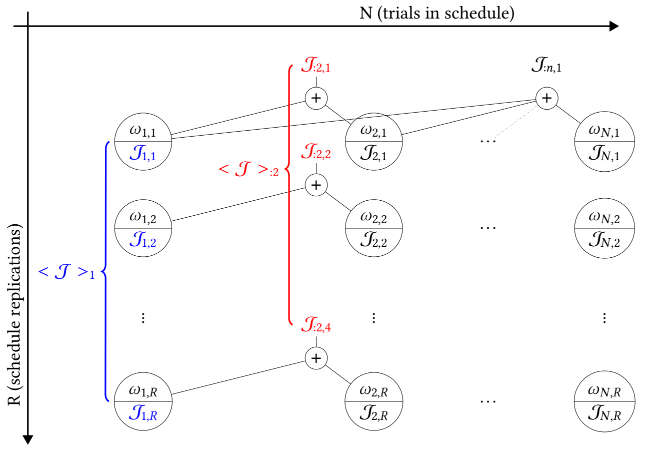
N (trials in schedule)
R (schedule replications)
$\omega_{1,1}$ $\mathcal{J}_{1,1}$
$\omega_{2,1}$ $\mathcal{J}_{2,1}$
$\omega_{N,1}$ $\mathcal{J}_{N,1}$
$\mathcal{J}_{:2,1}$
$\mathcal{J}_{:n,1}$
$\mathcal{J}_{:2,2}$
$\mathcal{J}_{:2,4}$
$< \mathcal{J} >_{:2}$
$< \mathcal{J} >_1$
$\omega_{1,R}$ $\mathcal{J}_{1,R}$
$\omega_{N,R}$ $\mathcal{J}_{N,R}$

Figure 6: How we estimate the informativeness of schedules. We evaluate a schedule with N trials R times. For each trial n in schedule r, we observe the outcome $\omega_{n,r}$ and compute the observed information $\mathcal{J}_{n,r}$. By averaging over all replications, we obtain $< \mathcal{J} >_n$ the Fisher information for trial n. The observed information for the n first trials par a particular repetition r is the sequence observed information $\mathcal{J}_{:n,r}$, and its average over all replications is the sequence Fisher information $< \mathcal{J} >_{:n}$

5.2 A Measure of information: $\mathcal{I}$

The three information measures defined just before are matrices *i.e.*, a multi-dimensional object, which makes them inconvenient to rank schedules. To get a scalar measure of the information of a sample, we need two things: 1) Derive a scalar measure from a matrix, and 2) make sure that this measure is positive. As scalar measure of information $\mathcal{I}$, we computed the square root of the determinant of $\mathcal{J}(\hat{\theta})$, where $\mathcal{J}(\hat{\theta})$ is any one of the three metrics defined before

$$\mathcal{I} = \sqrt{\det(\mathcal{J}(\hat{\theta}))}. \tag{4}$$

The square root of the determinant of a covariance matrix is called a generalized variance, and was previously used and motivated in HCI in [20, Section 4.3]. The generalized variance of the observed information matrix has also been used before as an objective in optimal experiment designs [10].

Sometimes, the inverse of the observed information matrix is not definite semi-positive[13]. To solve that problem, we use an algorithm by Higham [22] that outputs $\widetilde{\mathcal{J}}(\hat{\theta})$, the closest positive semi-definite matrix to $\mathcal{J}(\hat{\theta})$, and we use this information matrix rather than the original one in $\mathcal{I}$. In the end, we get

$$\mathcal{I} = \sqrt{\det(\widetilde{\mathcal{J}}(\hat{\theta}))}. \tag{5}$$

We illustrate the Fisher information (blue), its binned average together with its 95% bootstrap computed CI (red) and the sequence Fisher information (orange) for a schedule where k and Δ_t are uniformly sampled, which we call a uniform iid schedule, see Figure 7. This schedule can not be performed with real participants (it assumes the memory can be "loaded" with an arbitrary history), but provides for an interesting upper bound. Indeed, because recalls are identically distributed and independent, all trials should in theory be equally informative, which is indeed the case in Figure 7. As a result, the sequence Fisher information is linearly increasing.[14]

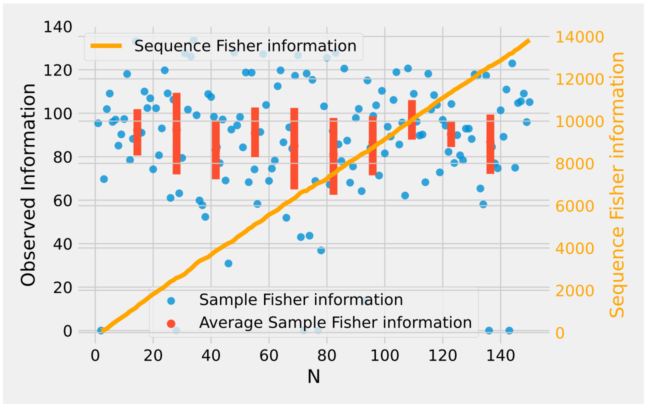

Figure 7: Fisher information (blue) computed over all 1000 repetitions, for an iid schedule for each of the $N = 150$ trials. Its binned average and 95% bootstrap computed CI are displayed in red bars. The sequence Fisher information is displayed in orange. Simulation parameters detailed subsection B.3.

5.3 What makes a schedule informative?

The uniform iid schedule used above, while important to illustrate the metrics, is impossible to enforce in practice because the experimenter cannot impose an arbitrary sequence of k's (repetitions in the EF model): the number of repetitions has to be incremented by one after each presentation of the corresponding command. We now explore a few realistic schedules, to get more insight into the informativeness of schedules.

Linear Schedule. A linear schedule is one where the delays between trials are equal. The information for a linear schedule is displayed in Figure 8. Information is null at the start, increases between $N = 10$ and $N = 25$, and then levels off. Without surprise, before $N = 10$, recall is null, and after $N = 25$, recall is perfect.

A constant recall probability Schedule. The sequence Fisher information in the linear schedule (and our initial intuition) suggests that a schedule where the recall probability stays away from 0 and 1 will be more informative than otherwise. Such a schedule is easy to build: we simply have to increase the delay with the repetition number. For example, we can ensure a constant recall probability p by adjusting the delay as $\Delta_t = \frac{-\log p}{\alpha(1-\beta)^k}$. We evaluated this idea

[13]As the observed information matrix is the inverse of a covariance matrix, it should by definition be positive semi definite — the multiple dimensions equivalent of saying that the variance should be positive. However, for low sample sizes, observed information can be unreliable and actually lead to what essentially are negative variance estimates.

[14]This information graph plotted together with the recall events and the bias and standard deviation of the estimator can be found in the supplementary materials. This is the case for all such information plots. Together, they show bias and standard deviation get reduced with the sequence Fisher information.

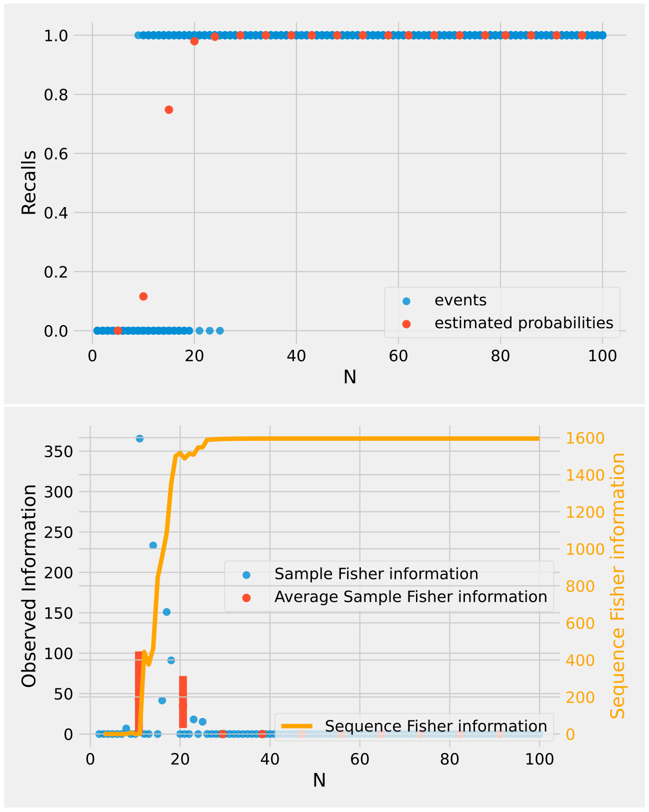

Figure 8: Top panel: Recall events and estimated recall probabilities as the schedule progresses. Bottom panel: Associated Fisher and sequence Fisher informations. Simulation parameters are detailed subsection B.3.

on the uniform iid schedule for several constant recall probabilities and for $\alpha = 10^{-2}$ and $\alpha = 10^{-3}$. Figure 9 displays the maximum of the sequence Fisher information; surprisingly to us, it is more informative to keep the recall rate relatively low *e.g.*, about $p = 0.2$ for $\alpha = 1e^{-2}$ and $p = 0.1$ for $\alpha = 1e^{-3}$. Indeed, recall rates are typically much higher in RBIT studies.

A typical schedule. We evaluated a typical schedule from the RBIT literature, very similar to the one used in [17] see the top panel Figure 10. The schedule was executed for multiple commands, where each command was presented ten times in total over two days, with various pauses between blocks. This schedule has the same linear increase in sequence Fisher information, as did the iid schedule, which shows it is rather effective at accumulating information. The bottom panel of Figure 10 shows the evaluation of an equivalent schedule, with the same number of presentations but where the time between trials is computed to maintain a recall probability $p = 0.2$. We observe the same linear trend for sequence Fisher information, but on average each trial is about three times more informative. As a result, the typical schedule evaluated here, which spanned two days, is three times less informative than the $p = 0.2$ adjusted schedule, which spans less than 3 hours. More details about this simulation are given in Appendix B.

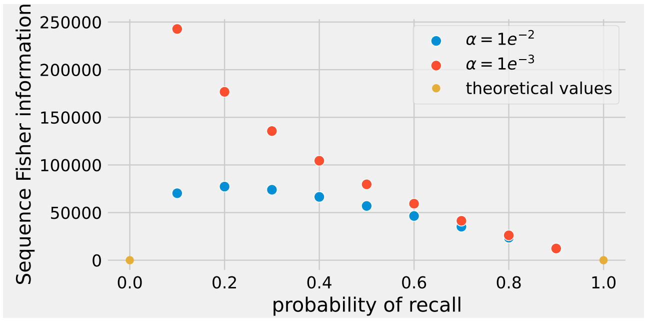

Figure 9: Sequence Fisher information plotted against the expected recall probability for an iid schedule. Simulated with $\beta = .4$, for $\alpha = 1e^{-2}$ (blue) and $\alpha = 1e^{-3}$ (orange). Simulation parameters detailed subsection B.3.

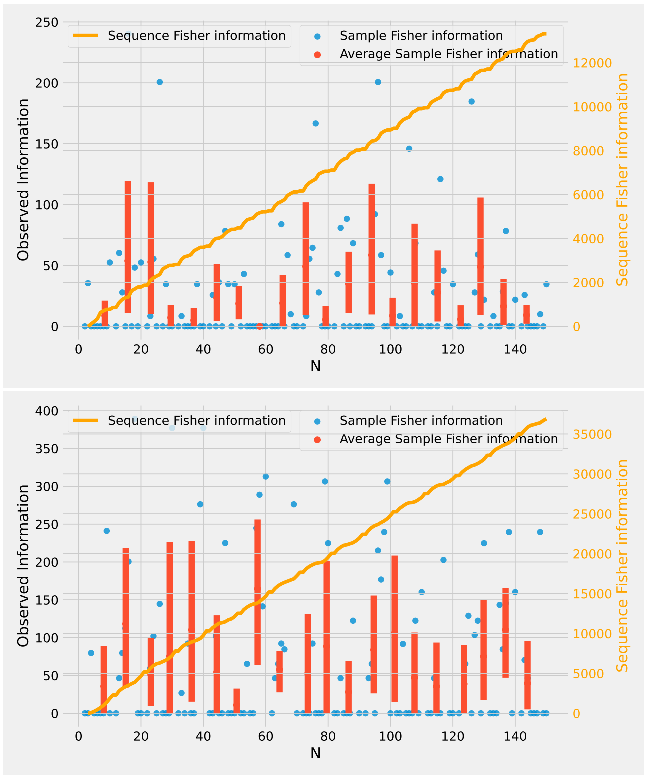

Figure 10: Sample and sequence Fisher information from a typical two-day schedule from the RBIT literature (top panel) with pauses in between blocks, and from a schedule where the duration between blocks is determined to maintain an expected recall probability of $p = 0.2$ (bottom panel). Notice the difference in y-axes, the schedule in the right panel is three times more informative. Simulation parameters for both panels detailed subsection B.5.

5.4 Information is independent of the schedule type

We computed information for various schedules; one could wonder whether an information of X for one schedule type is equivalent to X for another. In theory, the answer is yes: information as defined in Equation 5 maps directly to the size of CIs and CEs. We verified this by plotting the standard deviation of the estimates against the information of the schedule for all schedules that we used in this work (including results not communicated in this paper), see Figure 11. As expected, for a given assumed true value of θ, the more information increases, the better its estimate.

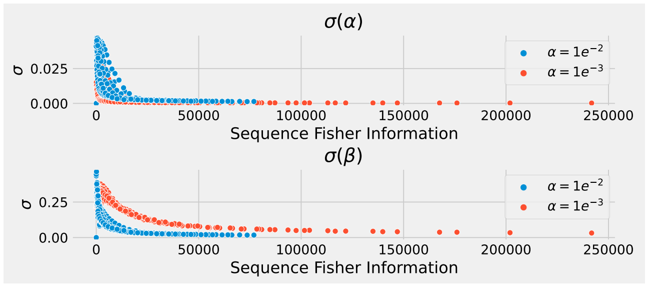

Figure 11: Standard deviation of α ($\sigma(\alpha)$, top panel) and β ($\sigma(\beta)$, bottom panel) as a function of the sequence Fisher information. The data is the aggregate of all the simulations conducted in this work.

These results support that information as defined in this section is a good objective function to maximize or compare schedules.

5.5 Does the typical method based on computing recall rates also benefit from more informative schedules?

Finally, one may also wonder whether a better schedule for the model-based estimation translates to a better schedule for the typical method based on computing recall rates and assessing their significance with p-values. To test this, we simulated a comparative experiment with two RBITS: one with ($\alpha = 10^{-1.9}$, $\beta = .4$), the other with ($\alpha = 10^{-2.1}$, $\beta = .4$). From Figure 9, the most informative condition for ($\alpha = 10^{-2}$, $\beta = .4$) is around a constant probability of recall of $p = 0.2$.

We thus created schedules that maintain a constant recall probability in increments of 0.1, to see whether the $p = 0.2$ condition would again lead to better discriminability between the two RBITs. For each of these schedules, we simulated an experiment, and computed recall rates. We then computed the significance of the difference in recall rates between the two RBITs, which gave us several p-values, one per block. To aggregate these p-values into one, we used two multiple comparisons methods, the Fisher and Stouffer methods [9][15]. This was repeated 100 times. The average of the combined p-values over all repetitions, plotted against the constant recall probability used to design the schedules are displayed in Figure 12. In line with Figure 9, the region of recall probability around 0.2 that maximized the informativeness of the schedule also maximizes the discriminability of the two RBITs when measured in terms of combined p-values, be it with Fisher or Stouffer p-values.

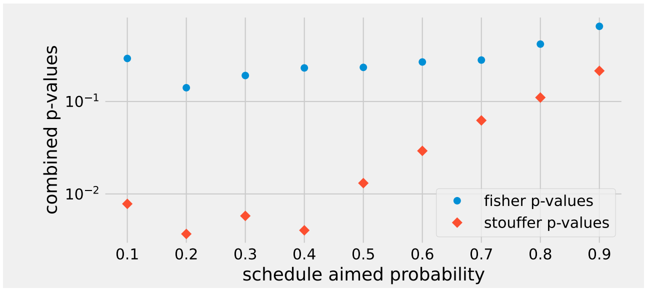

Figure 12: Combined p-values for the difference in mean recalls for schedules that aim for a given recall probability, for two methods of combining p-values: Fisher's method (in blue) and Stouffer's method (in orange). Simulation parameters detailed subsection B.6.

In summary, an informative schedule is thus one where the delay is adjusted with repetition number, keeping the recall probability low. While it has been recognized by some experimenters that integrating longer delays as the experiment progresses is beneficial *e.g.*, having experiments extend more than one day [17], we provide here a quantitative relationship that allows more precise design of schedules. Additionally, and in contrast with existing designs, computing these delays to maintain a rather low probability of recall for participants is recommended. It is important to note that schedules are ranked for a given memory model and parameter values. This implies that an estimate of these values are known prior to the experiment, which can be achieved by means of a pilot study, or by using existing data from the literature. Finally, because the goal is usually to compare two different RBITs, one has to decide which value for α and β to consider (should the parameters of the first or the second RBIT be used?). Here, a practical advice is to consider the average of parameter values.[16]

6 DISCUSSION

In this work, we introduced model-based evaluations for experiment design and comparisons of RBITs based on fitting an exponential forgetting (EF) model on recall data using the method of ML. We validated several aspects of our approach. We now discuss how the approach can be generalized, as well as some of its implications for HCI.

[15]These two methods are suited to combine independent p-values. However, since the same RBITs have been evaluated in each block by the same participants, the p-values are dependent. More sophisticated methods exist for combining multiple dependent p-values, see [9] for more details, but are out of the scope of this paper. So, while the exact p-values might be incorrect, we assume that the relative difference between them is meaningful.

[16]If the two RBITs are similar in performance then the schedule will be close to optimal. In the case when two RBITs have very different performances, it remains a question what the best schedule might be, a reasonable strategy would be to compare the two schedules designed for the two different pairs of parameters and then decide — however, if differences are remarkable, any reasonable schedule will likely pick up the differences anyway.

6.1 Considering alternative models

As with any model-based evaluation, the result of our work depends on how good of a description the EF model is for how people learn an RBIT. That being said, the general method as well as the measures of information remain valid for any other model. For example, we validated the ML approach of section 4 with a 3-parameter ACT-R model [34]. The model reads

$$p = \left(1 + \exp\left(\frac{\tau - m}{s}\right)\right)^{-1} \text{ with } m = \log\left(\sum_{i=1}^{n} \Delta t_i^{-d}\right) \qquad (6)$$

where the activation, m , is a power function of the times Δt_i since the command i was presented, and the exponent d stands for delay, and s and τ are parameters that add extra flexibility to the model. In comparison with the EF model where recall is just a function of when the last command was presented, the ACT-R model accounts for all past command presentations via m. Concretely, ACT-R predicts different behavior when items are repeated *e.g.*, as in the training schedule used in [37]. While the ACT-R model is a bit more complex, it is still possible to derive the needed Hessians analytically (the full derivations can be found in the supplementary materials). The validation of the ML method for ACT-R, as well as the coverage of the CIs is given Figure 13, with similar conclusions to the EF model.

Yet other models can be accounted for. For example, to isolate the impact of a given experimental factor on memory and RBIT evaluation, one may look to modify the model by including some mechanism that accounts for this factor. That factor could be *e.g.*, stimulus exposure duration, or a grouping factor to distinguish between several groups. But what if as a result the model becomes rather complex? As long as it can be described by an explicit equation, one can use symbolic calculus[17] to derive an expression for Hessians if they become too difficult to compute by hand. If that too is impossible, it remains possible to compute numerical estimates of the Hessian, such as the one used to compute the BFGS coverage in subsection 4.4, although their coverage may have to be carefully assessed. Finally, for simulator-like models, where the likelihood is not always possible to express, one can still use likelihood-free methods to infer model parameters (*e.g.*, with ABC, see next subsection). Note that the comparisons and design of schedules could be performed for multiple models, potentially providing more robust conclusions.

6.2 What to do when full recall information is not available?

The proposed method for model-based evaluations requires information that is not always easy to get from existing datasets. While the number of repetitions (k) is usually implicit from the description of the study design, this is not the case for the timing information. Timestamps needed to compute delays are easy to get in theory but are rarely reported in traditional RBITs evaluations, as a limited corpus of data that we collected from colleagues showed. In the example of section 3, we were forced into reconstructing the timing data. One immediate recommendation for authors is thus to gather and publish precise timing information together with recall data.

[17]In fact, the calculations of the Hessians in our library *pyrbit* were verified with symbolic calculus for both models.

With more missing information (like recall data), it remains possible to infer the memory model's parameters based *only* on the recall rates *e.g.*, by leveraging Approximate Bayesian Computation (ABC) [42], a method previously introduced in HCI [24]. ABC is an inference method which uses a simulator to generate outcomes which are then aggregated to summary statistics; these are then compared to a reference set of summary statistics (*i.e.*, the ones reported in a paper). The estimated parameters are the ones which lead to the smallest discrepancy between summary and reference statistics.

To illustrate ABC, we used PyMC4's [46] Sequential Monte Carlo version of ABC (ABC-SMC), and implemented a custom simulator. The simulator uses the EF memory model with the schedule and estimated parameters of BodyLoci [17] ($\alpha = 10^{-2.5}$ and $\beta = .77$, see subsection 4.5 and Figure 16) to evaluate recall rates from an entire experiment with multiple items and participants. The resulting forest plot for the estimation of parameters with ABC is displayed in Figure 14. One sees that while the posterior means are close to the true parameters, the 95% Highest Density Interval (HDI) is very wide. This is unsurprising considering we have a lot less information available than with the full recalls. We also used weakly informative priors (uniform distribution on $[-6, -.5]$ for $\log_{10}\alpha$ and on $[0.01, 0.99]$ for β), and did not fine-tune the procedure extensively, which may explain some of the width of the HDI.

6.3 The distortion due to execution time

In subsection 3.2, we showed that differences in execution time create differences in the evaluation based on recall rates, but not in the model-based evaluation. It is worth discussing what the "ideal" behavior of any evaluation is in that case: should it be sensitive to different execution times or not? We do not provide a definite answer to that question, as we believe it depends on the exact context and goal of the experiment. We do give several arguments to consider before deciding whether the execution time introduces an unwarranted distortion or not:

- Many studies intend to assess execution time separately from recall rates, as they will often display separate barplots for execution time and recall rates. Including execution time in recall rates goes against that intention.
- The execution time is related to the experimental protocol, in particular whether the intertrial time is controlled or not. It is also related to the particular implementation of the RBIT. For example in marking menus, the delay that is chosen to make the novice mode appear is arbitrary and may vary between implementations. Do we want to take this into account into the recall rates?
- Once the parameters of recall have been properly estimated, one can perform various simulations with various execution times to investigate their effect. Having an estimation that is free from execution time thus offers more flexibility from the perspective of modeling/simulation.
- Execution time is intimately linked with skill — usually the more skilled the participant the lower the execution time. This introduces a bias towards techniques that are well known / where skills transfer fast. This bias may be precisely what the experimenter intends to measure, but may also be unwanted.

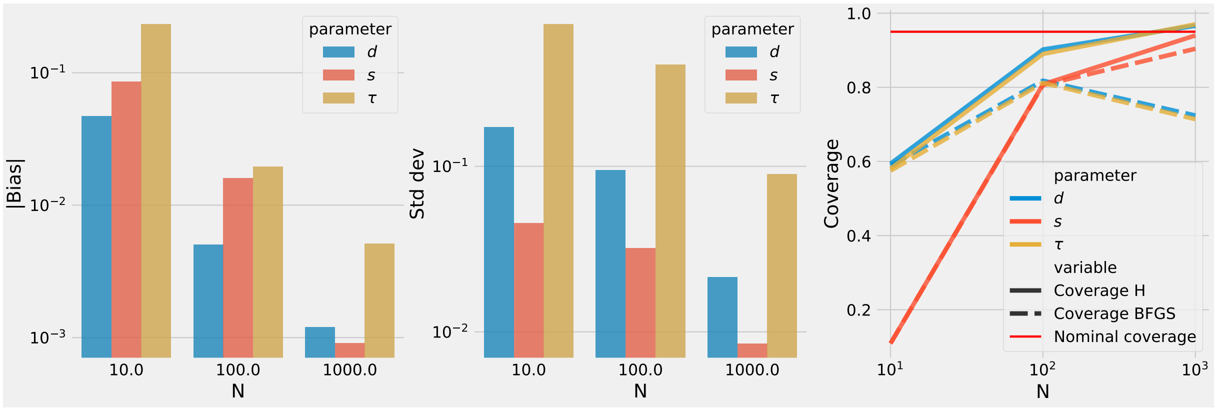

Figure 13: ML estimation for increasing sample size for ACT-R. Left and middle panels: bias and standard deviation of the estimates in log scale. Right panel: coverage of the CIs constructed with the analytical Hessian (Coverage H) and the numerical Hessian (Coverage BFGS).

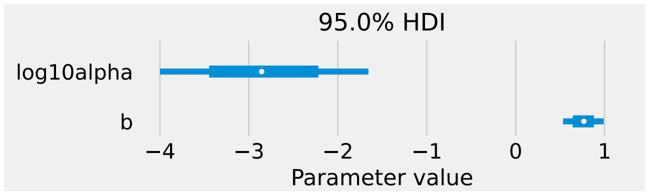

Figure 14: Forest plot for estimation of $\log_{10} \alpha$ and β (b) using ABC.

Researchers should carefully consider these arguments before deciding whether or not to adopt a model-based evaluation and/or control for potential differences in intertrial time.

6.4 On model-based and model-free evaluations

While this paper argues for a model-based evaluation, we do not reject model-free evaluations like those based on recall rates: both methods are complementary, just like parametric and non-parametric statistical tests are [3]. In particular, model-free evaluations, which operate by computing summary statistics on observed data do not rely on the assumption that a particular model well describes recall data. However, provided one can formulate an adequate model, the model-based approach provides several benefits shown in this paper, as well as the ability to predict, and thus to plan *e.g.*, if the model-based evaluation is part of an intelligent agent [33]. Just thinking about memory models might already be helpful to design experiments and analyze their outcomes, as a particular model implicitly specifies which are the important factors to consider. For example the role of delay in EF is primordial; recognizing this could have sparked an empirical investigation on the effect of different execution times like the one in subsection 3.2.

7 PYTHON LIBRARY: PYRBIT

To help researchers engage with this work, we release a Python library called Pyrbit that provides utilities and methods for inference and simulation of RBITs, as well as tools to estimate the informativeness of an arbitrary schedule. It currently has about 2000 lines of code. The library namely features ML and ABC submodules for inference, ACT-R and EF submodules that implement the models as well as the Hessian calculations. These two submodules also propose custom visualisations that help the end-user assess how well the models fit the data. Pyrbit also has submodules to help designing experiments; an *information* module to evaluate the informativeness of a schedule, and the *design* submodule that can be used to run a power analysis.

The library is available on PyPI, Python's package index, and can be installed using `pip install pyrbit`. The library is also documented at https://jgori-ouistiti.github.io/pyrbit/ with more details and examples codes for the following use cases:

- Estimating and plotting the parameters of memory models (EF and ACT-R, see subsection 6.1) from a sequence of recalls and the associated CI/CEs, using ML or ABC (see subsection 6.2). This can be used directly to compare RBITs.
- Evaluating the informativeness of any arbitrary schedule, measured by sequence Fisher information, together with bias and standard deviation information of the estimator. This can be used to rank schedules and/or select an appropriate sample size.
- Comparisons of RBITs based on recall rates, including power analyses for statistical tests aggregating recall rates for two RBITs. This can be used as a complimentary analysis to rank schedules according to the more typical design criterion based on recall rates and/or select an appropriate sample size.

8 FUTURE WORK

We foresee several directions for future work.

8.1 How similar is PAL to learning in the RBIT context?

The PAL paradigm abstracts away most of the complexities of RBITs. However, one may question how much the original problem has been simplified, and how much of the substance of RBITs has been lost. There are several differences between PAL and RBITs. In PAL, very little time is left to the participant to form recall strategies. For example, Ebbinghaus presented new items at the high rate of 1 pair every 0.4 seconds in his seminal study [31]. In comparison, the rate of pair presentations is much lower in typical HCI RBIT studies and may involve different phenomenon. For example, for marking menus [26], the training phase consists in participants selecting a command in the menu using the novice mode, which is both time consuming (a couple of seconds) and sollicits motor memory. Another example is provided by Fruchard *et al.* [17], who list strategies used by participants to enhance their recall: one participant makes associations between items *e.g.*, "the eagle is flying above the monkey", because the items are not entirely nonsensical. Some RBIT studies also involve decision-making *e.g.*, participants get to choose between two interaction techniques, or the choice of the modality during the training phase and sometimes during the recall phase, such as in [2], where the participant may decide between keyboard and stroke shortcuts during some of the test phases.

Without more work, it is hard to determine how much standard memory models developed for PAL are suited to model RBITs, but section 3, subsection 4.5 and Appendix A show that even a very basic model such as EF provides decent explanatory power. An important future work is to gain a better understanding of this aspect. This includes more precise models of how users interact with RBITs (*e.g.*, [4]). We remain optimistic because, as shown in Figure 10, there seems to be a lot of room for improvement in current schedules.

8.2 Empirical validation of optimal designs

This work presented the theory of model-based evaluations applied to the EF model, and how it could be leveraged to design schedules to compare RBITs. Both contributions were validated through several simulations. Model-based evaluation was applied to previously published empirical data, however, we did not conduct an empirical evaluation of the schedule design method. The primary reason is that simulations enable running a multitude of experiments that would otherwise require many different participants, in order to reliably evaluate different schedules [30]. In particular, the ground truth in an empirical study can only be estimated by repeating the study several times, which is very costly. In the future, we plan to re-run existing RBIT studies with the original and our proposed schedules to assess the possible benefits of our proposed method.

8.3 A meta-analysis of RBITs

Based on model-based evaluations, we intend on conducting a large scale meta-analysis of RBITs. Meta-analyses are statistical analyses that aggregate the results of multiple scientific studies — for example, Fisher's method of combining independent p-values presented in subsection 5.5 is a technique that can be used for a meta-analysis. The model-based evaluation would allow aggregating results from RBITs evaluated with different schedules. The use of meta-analyses in HCI is advocated by Kay *et al.* [25], who considers that the "poor state of knowledge accrual in HCI " can be improved by conducting meta-analyses. Given that for many studies, in particular older ones, it may be hard to obtain the data required for the ML analysis, we will likely combine ML and ABC methods. The parameters identified in each study may be influenced by experimental factors not considered in the model, such as whether visual feedback was provided upon selection. This difficulty, shared by the classic recall-percentage method, could be addressed in the model-based evaluation by using memory models that may account for these factors as discussed in subsection 6.1.

8.4 Improving small sample behavior of the estimator

Estimating online the parameters of a memory model used to represent a user interacting with a device is useful to design adaptive or intelligent user interfaces [8]. For example, memory models can be exploited in intelligent tutoring systems; see [33] for an approach that uses the EF model and its inferred parameters to adopt the best teaching strategy, and references therein for similar works using different models. The main place for improvement of the proposed inference method lies in the small sample regime; indeed ML being asymptotically optimal (it reaches the Cramer-Rao bound, see subsection 4.2), there is little room for improvement for large sample sizes. Bayesian inference would also constitute an alternative for the small sample regime (Bayesian inference is asymptotically equivalent to ML). The bias of the ML estimates, and potential solutions to reduce it are discussed by Firth [13], and could likely be exploited.

ACKNOWLEDGMENTS

This research was funded, in whole or in part, by l'Agence Nationale de la Recherche (ANR), project ANR-22-CE33-0006-01. For the purpose of open access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

REFERENCES

[1] John R Anderson, Daniel Bothell, Michael D Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. 2004. An integrated theory of the mind. *Psychological review* 111, 4 (2004), 1036.

[2] Caroline Appert and Shumin Zhai. 2009. Using Strokes as Command Shortcuts: Cognitive Benefits and Toolkit Support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 2289–2298. https://doi.org/10.1145/1518701.1519052

[3] Erkie Asmare and Andualem Begashaw. 2018. Review on parametric and non-parametric methods of efficiency analysis. *Biostatistics and Bioinformatics* 2, 2 (2018), 1–7.

[4] Gilles Bailly, Mehdi Khamassi, and Benoît Girard. 2022. Computational model of the transition from novice to expert interaction techniques. *ACM Transactions on Computer-Human Interaction* (2022).

[5] Olivier Bau and Wendy E Mackay. 2008. OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. 37–46.

[6] Dimitri P Bertsekas. 2014. *Constrained optimization and Lagrange multiplier methods*. Academic press.

[7] Xumeng Cao and James C Spall. 2012. Relative performance of expected and observed Fisher information in covariance estimation for maximum likelihood estimates. In *2012 American Control Conference (ACC)*. IEEE, 1871–1876.

[8] Allan Dafoe, Edward Hughes, Yoram Bachrach, Tantum Collins, Kevin R McKee, Joel Z Leibo, Kate Larson, and Thore Graepel. 2020. Open problems in cooperative AI. *arXiv preprint arXiv:2012.08630* (2020).

[9] Hongying Dai, J Steven Leeder, and Yuehua Cui. 2014. A modified generalized Fisher method for combining probabilities from dependent tests. *Frontiers in genetics* 5 (2014), 32.

[10] P Fernandes de Aguiar, B Bourguignon, MS Khots, DL Massart, and R Phan-Than-Luu. 1995. D-optimal designs. *Chemometrics and intelligent laboratory systems* 30, 2 (1995), 199–210.

[11] Bradley Efron and David V Hinkley. 1978. Assessing the accuracy of the maximum likelihood estimator: Observed versus expected Fisher information. *Biometrika* 65, 3 (1978), 457–483.

[12] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM conference on recommender systems*. 101–109.

[13] David Firth. 1993. Bias reduction of maximum likelihood estimates. *Biometrika* 80, 1 (1993), 27–38.

[14] Ronald A Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 222, 594-604 (1922), 309–368.

[15] Paul M Fitts. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology* 47, 6 (1954), 381.

[16] Bruno Fruchard, Eric Lecolinet, and Olivier Chapuis. 2017. MarkPad: Augmenting touchpads for command selection. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5630–5642.

[17] Bruno Fruchard, Eric Lecolinet, and Olivier Chapuis. 2018. Impact of semantic aids on command memorization for on-body interaction and directional gestures. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces*. 1–9.

[18] Bruno Fruchard, Eric Lecolinet, and Olivier Chapuis. 2020. Side-crossing menus: enabling large sets of gestures for small surfaces. *Proceedings of the ACM on Human-Computer Interaction* 4, ISS (2020), 1–19.

[19] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar Button Icons to Communicate Keyboard Shortcuts. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4715–4726. https://doi.org/10.1145/3025453.3025595

[20] Julien Gori and Quentin Bellut. 2023. Positional Variance Profiles (PVPs): A New Take on the Speed-Accuracy Trade-off. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–16.

[21] Carl Gutwin, Andy Cockburn, Joey Scarr, Sylvain Malacria, and Scott C Olson. 2014. Faster command selection on tablets with FastTap. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2617–2626.

[22] Nicholas J Higham. 1988. Computing a nearest symmetric positive semidefinite matrix. *Linear algebra and its applications* 103 (1988), 103–118.

[23] Richard Arnold Johnson, Dean W Wichern, et al. 2002. Applied multivariate statistical analysis. (2002).

[24] Antti Kangasrääsiö, Kumaripaba Athukorala, Andrew Howes, Jukka Corander, Samuel Kaski, and Antti Oulasvirta. 2017. Inferring cognitive models from data using approximate Bayesian computation. In *Proceedings of the 2017 CHI conference on human factors in computing systems*. 1295–1306.

[25] Matthew Kay, Gregory L Nelson, and Eric B Hekler. 2016. Researcher-centered design of statistics: Why Bayesian statistics better fit the culture and incentives of HCI. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 4521–4532.

[26] Gordon Kurtenbach and William Buxton. 1994. User learning and performance with marking menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 258–264.

[27] Wanyu Liu, Gilles Bailly, and Andrew Howes. 2017. Effects of frequency distribution on linear menu performance. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1307–1312.

[28] Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. 2017. A tutorial on Fisher information. *Journal of Mathematical Psychology* 80 (2017), 40–55.

[29] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. 2017. SymPy: symbolic computing in Python. *PeerJ Computer Science* 3 (Jan. 2017), e103. https://doi.org/10.7717/peerj-cs.103

[30] Roderick Murray-Smith, Antti Oulasvirta, Andrew Howes, Jörg Müller, Aleksi Ikkala, Miroslav Bachinski, Arthur Fleig, Florian Fischer, and Markus Klar. 2022. What simulation can do for HCI research. *Interactions* 29, 6 (2022), 48–53.

[31] Jaap MJ Murre and Joeri Dros. 2015. Replication and analysis of Ebbinghaus' forgetting curve. *PloS one* 10, 7 (2015), e0120644.

[32] In Jae Myung. 2003. Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology* 47, 1 (2003), 90–100.

[33] Aurélien Nioche, Pierre-Alexandre Murena, Carlos de la Torre-Ortiz, and Antti Oulasvirta. 2021. Improving artificial teachers by considering how people learn and forget. In *26th International Conference on Intelligent User Interfaces*. 445–453.

[34] Philip I Pavlik and John R Anderson. 2008. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied* 14, 2 (2008), 101.

[35] Simon T. Perrault, Eric Lecolinet, Yoann Pascal Bourse, Shengdong Zhao, and Yves Guiard. 2015. Physical Loci: Leveraging Spatial, Object and Semantic Memory for Command Selection. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 299–308. https://doi.org/10.1145/2702123.2702126

[36] Aini Putkonen, Aurélien Nioche, Ville Tanskanen, Arto Klami, and Antti Oulasvirta. 2022. How Suitable Is Your Naturalistic Dataset for Theory-based User Modeling?. In *Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization*. 179–190.

[37] Quentin Roy, Sylvain Malacria, Yves Guiard, Eric Lecolinet, and James Eagan. 2013. Augmented letters: mnemonic gesture-based shortcuts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2325–2328.

[38] Skipper Seabold and Josef Perktold. 2010. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*.

[39] Galit Shmueli. 2010. To Explain or to Predict? *Statist. Sci.* 25, 3 (2010), 289 – 310. https://doi.org/10.1214/10-STS330

[40] R William Soukoreff and I Scott MacKenzie. 2004. Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. *International journal of human-computer studies* 61, 6 (2004), 751–789.

[41] Evan Starr and Brent Goldfarb. 2020. Binned scatterplots: A simple tool to make research easier and better. *Strategic Management Journal* 41, 12 (2020), 2261–2274.

[42] Mikael Sunnåker, Alberto Giovanni Busetto, Elina Numminen, Jukka Corander, Matthieu Foll, and Christophe Dessimoz. 2013. Approximate bayesian computation. *PLoS computational biology* 9, 1 (2013), e1002803.

[43] Md Sami Uddin, Carl Gutwin, and Benjamin Lafreniere. 2016. HandMark Menus: Rapid command selection and large command sets on multi-touch displays. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 5836–5848.

[44] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[45] Matthew M Walsh, Kevin A Gluck, Glenn Gunzelmann, Tiffany Jastrzembski, and Michael Krusmark. 2018. Evaluating the theoretic adequacy and applied potential of computational models of the spacing effect. *Cognitive science* 42 (2018), 644–691.

[46] Thomas Wiecki, John Salvatier, Ricardo Vieira, Maxim Kochurov, Anand Patil, Michael Osthege, Brandon T. Willard, Bill Engels, Colin Carroll, Osvaldo A Martin, Adrian Seyboldt, Austin Rochford, Luciano Paz, rpgoldman, Kyle Meyer, Peadar Coyle, Marco Edward Gorelli, Oriol Abril-Pla, Ravin Kumar, Junpeng Lao, Virgile Andreani, Taku Yoshioka, George Ho, Thomas Kluyver, Kyle Beauchamp, Alexandre Andorra, Demetri Pananos, Eelke Spaak, Benjamin Edwards, and Eric Ma. 2023. *pymc-devs/pymc: v5.6.0*. https://doi.org/10.5281/zenodo.8113401

[47] Robert C Wilson and Anne GE Collins. 2019. Ten simple rules for the computational modeling of behavioral data. *Elife* 8 (2019), e49547.

[48] Louis E Yelle. 1979. The learning curve: Historical review and comprehensive survey. *Decision sciences* 10, 2 (1979), 302–328.

A DETAILS ON THE CASE STUDY

A.1 Study summary

Studies prior to that of Fruchard *et al.* [17] had shown that spatial memory and semantic aids can help users learn and remember gestural commands. Using the body as a support to combine both dimensions had been proposed, but no formal evaluations had yet been reported. Fruchard *et al.*'s study evaluated on-body interactions, with an RBIT called *BodyLoci* and compared it with mid-air *marking menus*. For the two RBITs, command are mapped with arbitrary locations: either a zone on the body (BodyLoci) or a location in a marking menu. Both RBITs have a novice and an expert mode. The study consisted in interleaved learning (L) and test (T) blocks, with on the first day L1/T1/L2/T2/L3/T3 and on the second day T4/L4/T5. The study found no statistically significant benefit of using on-body interactions in a block recall percentage evaluation for T2/T3/T4/T5 and a significant difference for T1. The average execution times were: 5 seconds for BodyLoci, 4 seconds for marking menus.

We used the complete dataset of the experiment, which does not include precise timing information; we have the duration of a trial but not the associated timestamps, which we have to reconstruct based on information in the paper.

A.2 Reconstructing data

To reconstruct timing information, we added a constant intertrial time to the trial duration. In between blocks, we also added a constant interblock time, except between recall blocks R3 and R4, where we add a full day. With an intertrial time of 8 seconds, and an interblock time of 30 seconds, we get an experiment duration of (27 (marking menus)+31 (BodyLoci) = 58 minutes) for the first day and (11 (marking menus) + 13 (BodyLoci) = 24 minutes), which conforms to the paper's description (about an hour for the first day, 30 minutes for the second day).

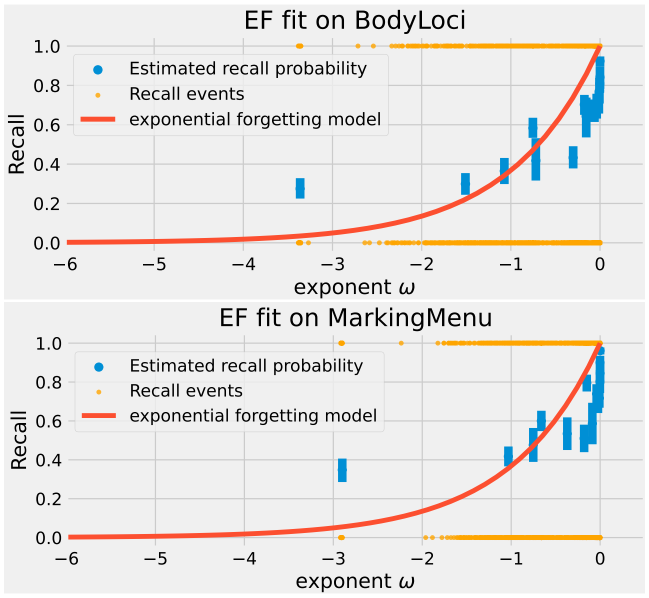

Figure 15: Binned scatterplots of recall events against the exponent for BodyLoci (left) and marking menus (right) for reconstructed data from [17].

To increment repetitions, we follow this logic:

- during the learning phases, participants could use either novice or expert mode, and the right location was always revealed. This means that whether the participant recalled correctly or not, the repetition number (k) is incremented.
- during the testing phase, if the recall is incorrect, k is not incremented because the correct location was not exposed systematically after an incorrect recall. If the recall is correct, k is incremented because finding the correct location reinforces the memory. We can't account for edge cases where *e.g.*, the participant hesitated between two locations, and selected the wrong one (the participant would then likely deduce where the correct location was, which would also refresh their memory).
- We do not use the recalls from the learning phase to fit the memory model. Indeed, during the learning phase the participants were free to choose the expert or novice mode; Some participants may choose to use novice mode even though they are capable of using the expert mode *e.g.*, by habit or by compliance with the idea of the novice mode in the learning phase.

To verify the quality of the reconstructed data, as well as the fit of the EF model to this experimental data, we propose the following visualization. We define the exponent of the exponential forgetting model as the term inside the exponential in Equation 1:

$$\omega = -\alpha(1-\beta)^k \Delta t \quad (\text{exponent } \omega \in \mathbb{R}^-), \tag{7}$$

which gives an exponential relationship between the probability of recall and the exponent:

$$p = \exp(\omega). \tag{8}$$

A binned scatterplot [41] of recall events against the exponent should thus appear to follow an exponential curve. The plot is displayed on Figure 15. Overall, the match between the reconstructed data and the EF model seems reasonable in catching the main effects of delay and repetition, except for the recalls around the exponent value of $\omega = 3$.

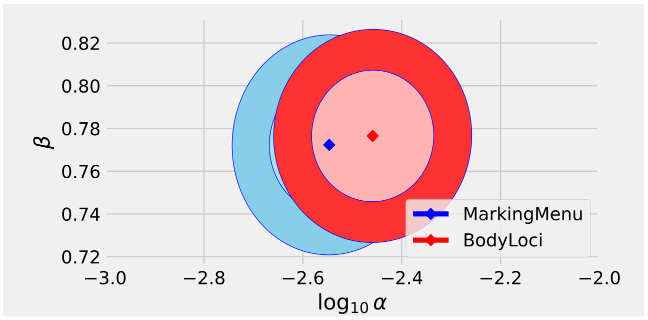

Figure 16: BodyLoci versus marking menus in the parameter space of the EF model. Parameters adjusted from recall data belonging to all participants and for all items. For more context, see the bottom panels of Figure 5.

B SIMULATION PARAMETERS

B.1 Figure 1

Parameters similar to those from [17], see Appendix A. We used more participants to be sure to have statistically significant difference that is obvious to the eye.

- $\beta = .75$
- $\alpha_A = 10^{-1.8}$, $\alpha_B = 10^{-1.7}$, $\alpha_C = 10^{-1.6}$
- interleaved learning and test phases: L1/T1/L2/T2/L3/T3/T4/L4/T5, command mapping always shown during test phases.
- schedules:

  - A vs C: interblock durations: [10, 10, 10, 200, 200, 100000, 200, 200] seconds (100000 seconds $\simeq$ 28 hours), intertrial time: 5 seconds.
  - B vs C: interblock durations: [0, 1000, 400, 400, 400, 86400, 1000, 400] seconds (86400 seconds = 24 hours), intertrial time: 4 seconds.
- 15 commands
- 100 participants

B.2 Figure 2

A one day experiment with three test phases. A shorter experiment better shows the distortion.

- $\beta = .5$
- $\alpha = 10^{-2.5}$
- interleaved learning and test phases: L1/T1/L2/T2/L3/T3, command mapping always shown during test phases.
- interblock durations: [10, 200, 200, 200, 200] seconds
- intertrial time A: 10 seconds
- intertrial time B: 5 seconds
- 50 commands
- 24 participants

B.3 Figure 4, Figure 7 and Figure 9

- $\beta = .4$
- $\alpha = 10^{-2}$ (and $\alpha = 10^{-3}$ for Figure 9)
- k drawn uniformly in $\{0, 1, \ldots, 10\}$
- Δ_t drawn uniformly in $[0, 5000]$ seconds
- 1000 repetitions
- schedule length: $N \in \{10, 100, 1000, 10000\}$ for Figure 4, $N = 150$ for Figure 7, $N = 250$ for Figure 9.

B.4 Figure 8

- $\beta = .4$
- $\alpha = 10^{-2}$
- $\Delta_t = 17280$ seconds (about 5 hours, for a schedule that lasted 20 days)
- schedule length: $N = 100$.
- one command, one participant.
- 1000 repetitions

B.5 Figure 10

- $\beta = .4$
- $\alpha = 10^{-2}$
- interleaved learning and test phases: L1/T1/L2/T2/L3/T3/L4/T4/L5/T5, command mapping always shown during test phases.
- Time instant at which block starts. Left panel:[0, 200, 400, 600, 800, 2000, 2200, 2400, 86200, 86400]. Right panel: [97, 161, 268, 447, 745, 1242, 2070, 3450, 5749, 9582]
- 1000 Repetitions

B.6 Figure 12

- $\beta = .4$
- $\alpha_A = 10^{-2.1}$, $\alpha_B = 10^{-1.9}$
- Schedule optimized for $\alpha = 10^{-2}$ (same parameters as the right panel of Figure 10)
- 100 Repetitions
- 1 items, 48 participants

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009