Modeling visual working memory with the MemToolbox

The authors[1]
[1] Department of Psychology, Harvard University, Cambridge, MA 02138

Abstract

The MemToolbox is a collection of MATLAB functions for modeling visual working memory. In support of its goal to provide a full suite of data analysis tools, the toolbox includes implementations of popular models of visual working memory, real and simulated data sets, Bayesian and maximum likelihood estimation procedures for fitting models to data, visualizations of data and fit, validation tools, model comparison metrics, and experiment scripts. The MemToolbox is released under a BSD license and is freely available at memtoolbox.org.

Keywords: visual working memory, mixture models, model comparison

Introduction

Working memory is a storage system that actively holds information in mind and allows for its manipulation, providing a workspace for thought (Baddeley, 1986). Its strikingly limited capacity has inspired a slew of research aimed at characterizing those limits in terms of both the spatiotemporal properties of the stimulus and the age, gender, intelligence, sleepiness, and mental health of the individual.

A handful of experimental paradigms are popular in the study of working memory. These include the delayed match-to-sample task used in studies of animal cognition (Blough, 1959) and the span tasks used in studies of verbal working memory (Daneman & Carpenter, 1980). In the case of working memory for visual information, research has primarily relied on the partial report task and the change detection task (Fig. 1). In a partial report task, the participant is shown a set of letters, shapes, or colorful dots, and then after a brief delay is asked to report the properties of one or a subset of

the items (Sperling, 1960; Wilken & Ma, 2004; Zhang & Luck, 2008). In a change detection task, the participant is shown a pair of displays, one after the other, and is then asked a question that requires comparing them, e.g., whether they match (Luck & Vogel, 1997; Phillips, 1974; Pashler, 1988).

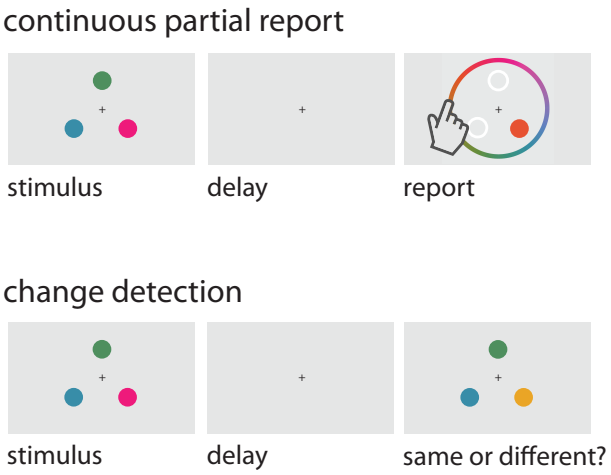

FIGURE 1: (a) A continuous report task. Observers see the stimulus display, and then at test are asked to report the exact color of a single item. (b) A change detection task. Observers see the stimulus display, then after a blank must indicate whether the test display is identical to the stimulus display or whether a single item has changed color.

Performance on these visual tasks is used to draw conclusions about working memory. For example, using data from a change detection task, Luck & Vogel (1997) showed that the same number of features as conjunctions of features can be stored in working memory, and from this inferred that the storage format in working memory is integrated objects, not individual features.

In recent years, a number of formal models have been proposed that link performance in change detection and partial report tasks to the architecture and capacity of the working memory system. These include the item-limit model (Pashler, 1988), the slot model (Luck & Vogel, 1997; Cowan, 2001), the slots+averaging model (Zhang & Luck, 2008), the slots+resources model (Awh et al., 2007), the continuous resource model (Wilken & Ma, 2004), the resources+swaps model (Bays et al., 2009), the ensemble statistics+items model (Brady & Alvarez, 2011), and the variable-precision model (van den

Berg et al., 2012). These models specify the underlying structure of visual memory and a model of how observers perform the task. They can they be used to examine how well the proposed memory representations explain the data of observers.

For example, to fit a model using probabilistic methods, a likelihood function is defined that describes the model's predictions for each possible setting of its parameters. Formally, if we have a model M with free parameters θ, the model's likelihood function must describe a probability distribution $P(D|\theta)$ over possible datasets D. Having formalized such a likelihood function, estimation procedures can be used to find the best parameters for the model and examine its fit. For example, maximum likelihood algorithms (Dempster et al., 1977; Lagarias et al., 1998) may be used to find the combination of parameter values that maximize that model's likelihood function for a given set of data. This procedure is typically performed separately for each participant and experimental condition, resulting in a set of estimates that are then compared using traditional statistical tests (e.g., Zhang & Luck, 2008). Using these values, claims can then be made about the architecture or capacity of memory (e.g., Bays & Husain, 2008; Zhang & Luck, 2008).

The MemToolbox

We created the MemToolbox, a collection of MATLAB routines for modeling visual working memory. The toolbox provides everything needed to perform the analyses routinely used in visual working memory, including model implementations, maximum likelihood routines, and data validation checks. In addition, it provides tools that offer a deeper look into the data and the fit of the model to the data. In the following sections, we highlight some of the toolbox's core functionality and some of the potential improvements it offers to the current standard workflow for modeling working memory.

The standard approach

The MemToolbox uses two MATLAB structs to organize information: a data structure, describing the data to be fit (including information about the paradigm); and a model structure, describing the model and its likelihood function. For example, in a continuous report paradigm, the data struct must contain a field called `errors` that indicates, for each trial, how far an observers' response was from the correct response (in degrees). This data structure can optionally contain other information about each trial, for

example, distractors , which gives the color of the other items in the display; or n, which indicates the number of stimuli present on the trial. Fitting a set of data with a model is then as simple as calling the built-in MemFit() function. For example:

```
>> model = StandardMixtureModel();
>> data.errors = [−89, 29, −2, 6, −16, 65, 43, −12, 10, 0, 178, −42];
>> fit = MemFit(data, model)
```

This will return the best fit parameters for this observers' data and credible intervals for these parameters, allowing for standard analysis techniques to be used. Thus, with very little effort the MemToolbox can be used to simplify (and speed-up) existing workflows by allowing for straightforward fitting of nearly all of the standard models used in the visual working memory literature. In support of this goal, the toolbox includes what we refer to as the StandardMixtureModel (that of Zhang & Luck, 2008), the SwapModel of Bays et al. (2009), several variations of a VariablePrecisionModel, and many more models.

The Bayesian approach

While the MemToolbox supports the standard workflow, the toolbox defaults to a Bayesian workflow that encourages full exploration of the data. Thus, rather than simply fitting the maximum likelihood parameters, the toolbox strongly encourages a more thorough examination of model fit. Thus, the default fitting function, MemFit, constructs a full posterior distribution over parameters rather than simply returning the maximum likelihood parameters.

Inference in any model requires combining the likelihood function with a set of prior beliefs about the value of each of the model's parameters. This prior, $P(\theta)$, expresses what parameter values are reasonable ones, and can be as straightforward as limits on the upper and lower bounds of the parameters (for example, treating the guess rate as limited to between 0 and 1). Once a set of prior beliefs is specified, the beliefs must then be updated (according to Bayes' rule) to take into account the experimental data. After observing data D, our beliefs about the parameters are given by:

$$P(\theta|D) \propto P(D|\theta) \cdot P(\theta)$$

In other words, the posterior probability distribution ($P(\theta|D)$) is given by the likelihood of the data given the parameters, combined with the prior

probability of the parameters. This posterior indicates the range of parameter values that should be considered reasonable after seeing the data. If we treat all priors as entirely flat, then the most probable value for the parameters θ is the maximum likelihood estimate: the setting of the parameters that maximizes the likelihood function, $P(D|\theta)$. Indeed, for the purposes of exploratory data analysis, it is common to use a noninformative or weakly informative prior that spreads the probability thinly over a swath of plausible parameter values. Following this tradition, the toolbox uses the Jeffreys prior, a class of noninformative priors that is invariant under reparameterization of the model (Jeffreys, 1946; Jaynes, 1968). These priors have almost no influence on the final inferences, but serve to limit the range of parameters to those that are meaningful.

Estimating the entire posterior distribution is more difficult than simply finding the maximum likelihood estimate. For some models it is possible to derive closed-form expressions for the posterior distribution, but for most models this is intractable and so sampling-based algorithms must be used to approximate it. One such algorithm, the Metropolis-Hastings variant of Markov Chain Monte Carlo (MCMC), is applicable to a wide range of models and is thus the one used by MemToolbox. MCMC works by choosing an initial set of model parameters, and then, for several thousands iterations, proposing small moves to these parameter values and either accepting or rejecting these moves based on how likely the new parameter values are in both the prior and the likelihood function. In this way, it constructs a random walk that visits parameter settings with frequency proportional to their probability under the posterior (Metropolis et al., 1953; Hastings, 1970). This allows the estimation of the full posterior of the model in a reasonable amount of time, but is theoretically equivalent to the more straightforward (but much slower) technique of evaluating the model's likelihood and prior at every possible set of the parameters (implemented in the GridSearch function).

With the posterior distribution in hand, there are a number of ways to analyze and visualize the results. First, the maximum of the posterior distribution (the maximum a posteriori or MAP estimate) can be used as a point-estimate that is analogous to the maximum likelihood estimate, differing only in the MAP's consideration of the prior. However, visualizing the full posterior distribution also provides information about how well the data constrains the parameter values and whether there are trade-offs between parameters (Fig. 2).

For example, Figure 2 shows a posterior for data analyzed with the standard mixture model of Zhang & Luck (2008). The plots on the diagonal show the range of reasonable values for each parameter (the marginals of

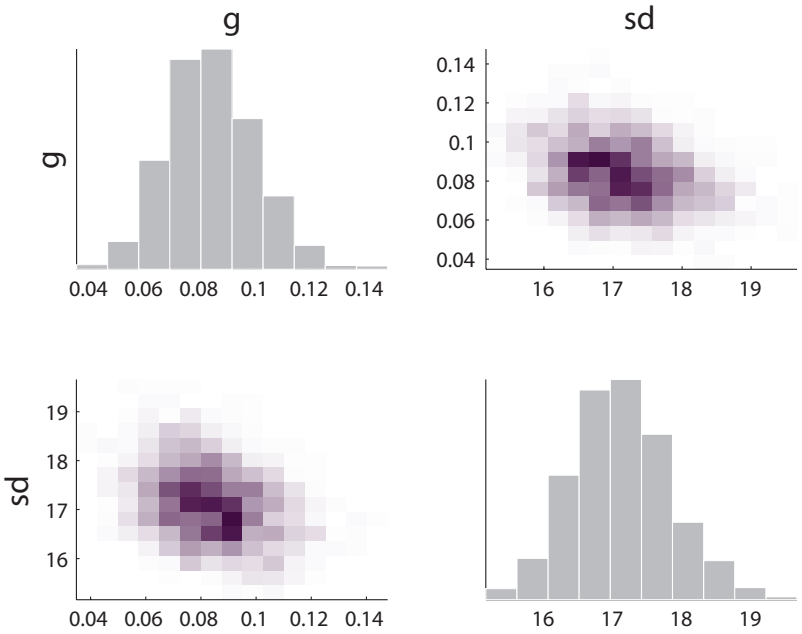

FIGURE 2: An example full posterior of the standard mixture model of Zhang & Luck (2008), where "g" is the guess rate and "sd" is the standard deviation of observers' report for remembered items.

the posterior); the correlations between parameters are shown on the off-diagonals. Note that in the standard mixture model, there is a slight negative correlation between the standard deviation parameter and the guess rate parameter: data can either be seen as having a slightly higher guess rate and a slightly lower standard deviation, or a slightly lower guess rate and higher standard deviation (as in Brady et al., 2011). As a result of this, neither parameter can independently be estimated as tightly as the combination of the two parameters can be – the "hill" in the posterior is tilted, such that the range of reasonable parameters is wider if your cross it horizontally (in g) or vertically (in sd), than if you cross it diagonally. Examining the full posterior visualizes this trade-off, which is hidden when using maximum likelihood fits.

Posterior predictive checks

Another technique applied by the MemToolbox is the automatic use of posterior predictive checks. Sometimes a whole class of models performs poorly, such that there are no parameter settings that will produce a good fit. In this case, maximum likelihood and maximum a posteriori estimates are misleading: they dutifully pick out the best, even if the best is quite bad. A good practice is to check the quality of the fit, examining which aspects of the data it captures, and which aspects of the data it fails to capture (Gelman et al., 1996, 2004). This can be accomplished through a posterior predictive check, which simulates new data from the posterior fit of the model, and then compares the histograms of the actual and simulated data (Fig. 3). Posterior predictive checks are a default function of MemFit.

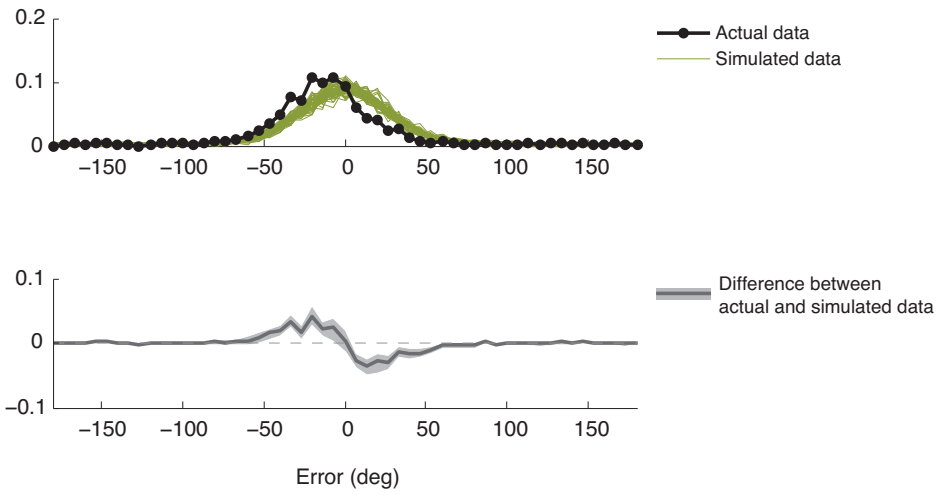

FIGURE 3: (a) Simulated data from the posterior of the model, with the actual data on top. The mismatch between the two indicates a poor fit. (b) The difference between the simulated and real data, with credible intervals. Any spot where the credible intervals do not include zero is an indication that the model does not accurately fit the data.

A model that can accurately fit the data in a posterior predictive check does not necessarily provide a good fit. For example, the model may fit the averaged data but fail to fit observers' data from individual displays, perhaps because of reports of incorrect items (Bays et al., 2009) or because of the use of grouping or ensemble statistics (Brady & Tenenbaum, 2010). In addition, a good fit does not necessarily indicate a good model: an extremely flexible model that can mimic any data always provides a good fit, but this provides

no evidence in favor of that model (Roberts & Pashler, 2000). However, models that systematically deviate in a posterior predictive check nearly always need improvement.

Hierarchical modeling

The standard technique for fitting multiple subjects involves fitting ... []

prevent extremely unlikely values in sensible ways – for example, using maximum likelihood estimates, observers with high guess rates are sometimes estimated to have guess rates of near zero but standard deviations of 3000 degrees. This problem is avoided by fitting subjects in a single model.

Model comparison

Often there is a question about which model best describes the data. Answering it requires considering both the resemblance between the model and data and also the flexibility of the model. Flexible models can fit many data sets, and so a good fit provides only weak evidence of a good match between model and data. In contrast, finding the same fit between an inflexible model and the data provides stronger evidence of a good match. To account for this, many approaches penalize more flexible models; these include the Akaike Information Criterion with correction for finite data (AIC_C ; Akaike, 1974; Burnham & Anderson, 2004), the Bayesian Information Criterion (BIC; Schwarz, 1978), the Deviance Information Criterion (DIC, Spiegelhalter et al., 2002), and the Bayes factor (Kass & Raftery, 1995). Implementations of these model comparison techniques are included in the toolbox. Specifically, performing model comparison in MemToolbox simply requires passing multiple models to the MemFit function:

```
>> model1 = StandardMixtureModel();
>> model2 = SwapModel();
>> modelComparison = MemFit(data, {model1, model2})
```

This will output all model comparison metrics as well as give descriptions of the metrics.

In general, we recommend doing model comparison by choosing a metric that is the least generous to your preferred model. Furthermore, we recommend computing the metric for each subject independently and using t-tests across subjects to make inferences about the best fitting models. Thus, if you believe the best model is the simpler model, you may wish to compute AIC_C values for each subject, and then do a t-test comparing the AIC_C values

for the simpler model against those of the more complex model. On the other hand, if you wish to claim the more complex model is a better fit, it makes a stronger argument if you do so by comparing BIC values or Bayes Factors, which more heavily penalize complex models. Importantly, by using t-tests over these values, you take into account subject variance, and are thus able to generalize to the population as a whole. By contrast, computing a single AIC_C or BIC value across all subjects does not allow generalization to the population, as it ignores subject variance (one subject that is fit much better by a particular model can drive the entire AIC_C or BIC value). This kind of *fixed effects* analysis can thus seriously overstate the true significance of results (**?**).

Availability

The MemToolbox is available on the web at **memtoolbox.org**. Installing the toolbox is as simple as running the Setup.m file included in the toolbox to add the folder to the MATLAB's default path. The distribution includes a tutorial that reviews all of the functionality, as well as demos and source code. It is released under a BSD license, allowing free use for research or teaching.

Conclusion

We created the MemToolbox for modeling visual working memory. This introduction gave a high-level overview of its approach and core features. To learn to use the toolbox, we recommend the tutorial, available in the supplementary materials or at **memtoolbox.org**.

Acknowledgments

This research was supported by the color green and the number 4. The authors thank ...

Commercial relationships: none.
Corresponding author:
Email:
Address: 33 Kirkland Street, Cambridge, MA, 02138.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Trans. Automatic Control*, AC-19, 716–723.

Awh, E., Barton, B., & Vogel, E. (2007). Visual working memory represents a fixed number of items regardless of complexity. *Psychological Science*, 18(7), 622.

Baddeley, A. D. (1986). *Working Memory*. Oxford University Press.

Bays, P. M., Catalao, R. F. G., & Husain, M. (2009). The precision of visual working memory is set by allocation of a shared resource. *Journal of Vision*, 9(10), 1–27.

Bays, P. M. & Husain, M. (2008). Dynamic shifts of limited working memory resources in human vision. *Science*, 321(5890), 851–854.

Blough, D. S. (1959). Delayed matching in the pigeon. *Journal of the Experimental Analysis of Behavior*, 2, 151–160.

Brady, T. & Alvarez, G. (2011). Hierarchical encoding in visual working memory: ensemble statistics bias memory for individual items. *Psychological Science*, 22(3), 384.

Brady, T., Fougnie, D., & Alvarez, G. (2011). Comparisons between different measures of working memory capacity must be made with estimates that are derived from independent data [response to anderson et al.]. *Journal of Neuroscience*, Oct. 14th.

Brady, T. & Tenenbaum, J. (2010). Encoding higher-order structure in visual working memory: A probabilistic model. In *Proceedings of the 32nd Annual Conference of the Cognitive Science Society* (pp. 411–416).

Burnham, K. P. & Anderson, D. R. (2004). Multimodel inference. *Sociological Methods & Research*, 33(2), 261–304.

Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24(01), 87–114.

Daneman, M. & Carpenter, P. A. (1980). Individual differences in working memory and reading. *Journal of Verbal Learning & Verbal Behavior*, 19(4), 450–466.

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)*, 39(1), 1–38.

Gelman, A., Carlin, J., Stern, H., & Rubin, D. (2004). *Bayesian data analysis*. CRC press.

Gelman, A., Meng, X., & Stern, H. (1996). Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, 6, 733–759.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1), 97–109.

Jaynes, E. T. (1968). Prior probabilities. *IEEE Transactions on Systems Science and Cybernetics*, 4, 227–241.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London, Series A, Mathematical and Physical Sciences*, 186(1007), 453–461.

Kass, R. E. & Raftery, A. E. (1995). Bayes factors. *Journal of the American Statistical Association*, 90(430), 773–795.

Lagarias, J. C., Reeds, J. A., Wright, M. H., & Wright, P. E. (1998). Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1), 112–147.

Luck, S. J. & Vogel, E. K. (1997). The capacity of visual working memory for features and conjunctions. *Nature*, 390, 279–281.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6), 1087–1092.

Pashler, H. (1988). Familiarity and the detection of change in visual displays. *Perception & Psychophysics*, 44, 369–378.

Phillips, W. (1974). On the distinction between sensory storage and short-term visual memory. *Attention, Perception, & Psychophysics*, 16, 283–290.

Roberts, S. & Pashler, H. (2000). How persuasive is a good fit? a comment on theory testing. *Psychological review*, 107(2), 358.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6, 461–464.

Sperling, G. (1960). The information available in brief visual presentations. *Psychological Monographs*, 74.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., & der Linde, A. V. (2002). Bayesian measures of model complexity and fit (with discussion). *Journal of the Royal Statistical Society, Series B*, 64(4), 583–616.

van den Berg, R., Shin, H., Chou, W.-C., George, R., & Ma, W. J. (2012). Variability in encoding precision accounts for visual short-term memory limitations. *Proceedings of the National Academy of Sciences*, 109, 8780–8785.

Wilken, P. & Ma, W. J. (2004). A detection theory account of change detection. *Journal of Vision*, 4(12), 1120–1135.

Zhang, W. & Luck, S. J. (2008). Discrete fixed-resolution representations in visual working memory. *Nature*, 453, 233–235.