# An R package for State-Trace Analysis

Melissa Prince, Guy Hawkins,
Jonathon Love & Andrew Heathcote

The School of Psychology,
The University of Newcastle, Australia

Address for Correspondence

Melissa Prince,
The School of Psychology,
Psychology Building, The University of Newcastle,
University Avenue, Callaghan, 2308, Australia
Email: Melissa.Prince@newcastle.edu.au
Phone: 61-2-49216959
Fax: 61-2-49216906

Abstract: 122 words

Body: 6844 words

References: 654 words

Tables: 1

Figures: 2

Figure Captions: 202 words

# Abstract

State-trace analysis (Bamber, 1979) is a graphical analysis that can determine whether one or more than one latent variable mediates an apparent dissociation between the effects of two experimental manipulations. State-trace analysis makes only ordinal assumptions, and so is not confounded by range effects that plague alternative methods, especially when performance is measured on a bounded scale, such as accuracy. We describe and illustrate the application of a freely available GUI driven package, **StateTrace**, for the R language. **StateTrace** automates many aspects of a state-trace analysis of accuracy and other binary response data, including customizable graphics and the efficient management of computationally intensive Bayesian methods for quantifying evidence about the outcomes of a state-trace experiment developed by Prince, Brown and Heathcote (2011).

One of the most fundamental questions asked in experimental psychology and the neurosciences concerns latent dimensionality: does a single latent (i.e., not directly observable) variable or dimension mediate the relationship between two or more experimental factors? For example, is recognition memory mediated by one (*memory strength*) or two (*familiarity* and *recollection*) processes (Dunn, 2004, 2008)? Traditionally, researchers seek particular patterns of data, called *dissociations,* in order to answer this question (Shallice, 1988) and so to infer the existence of functionally independent neural (e.g., Teuber, 1955) or cognitive (e.g., Glanzer & Cunitz, 1986) systems. A single dissociation occurs when an experimental factor selectively affects performance under one condition, or in one task or group, or by one measure, but not another. Stronger evidentiary value is often associated with a double dissociation, which occurs when a second factor selectively influences the second condition/task/group/measure but not the first (see Dunn, 2003, for a fuller treatment). Typically, a one-dimensional or one-system account is rejected when a dissociation is confirmed by a significant interaction test.

However, there have been many demonstrations that interaction tests of dissociations can only reject a one-dimensional explanation based on strong assumptions (e.g., Bogartz, 1976; Busemeyer & Jones, 1983; Dunn, 2003; Dunn & Kirsner, 1988; Henson, 2006; Loftus, 1996; Poldrack, 2006). Loftus (1978) described one cause of this problem that commonly plagues bounded response measures, such as accuracy calculated from binary data. When the function mapping the latent variable to the bounded response scale is nonlinear, an observed interaction (or equally, the failure to observe an interaction) might be scale dependent. For example, floor and ceiling effects can mean that neither the presence nor absence of an interaction is diagnostic of dimensionality. Even when an experiment is calibrated to avoid extreme performance confounding still cannot be ruled out without making further debateable assumptions that are difficult, if not impossible, to directly test (see Prince, Brown & Heathcote, 2011, for details).

*State-trace analysis* (Bamber, 1979), which is also known as *dimensional analysis* (Loftus, Oberg & Dillon, 2004), avoids these and the other problems that plague traditional dissociation methods (see Newell & Dunn, 2008, for a concise and non-technical treatment). It does so by making only the weak and arguably plausible assumption that latent variables have a *monotonic* effect on performance; that is, that the direction of the latent variable's effect does not change with its magnitude. Latent dimensionality can then be assessed in a way that is easily understood and reported, using a *state-trace plot*. This approach is as general and flexible as dissociation analysis; the plot can be made by graphing one type of dependent variable against another (a *dependent-variable state-trace analysis*) or by graphing the same dependent measure taken under different conditions or from different tasks or groups. In all cases, if the plot is monotonic (i.e., does not change from increasing to decreasing or vice versa) performance is mediated by the same latent variable; otherwise more than one latent variable must be in play.

State-trace analysis has been applied to a diverse range of topics in areas, ranging from basic research in perception, attention, short-term and long-term memory, categorization, problem solving and meta-cognition to applications in aging, legal, clinical, educational, human factors and developmental psychology (Prince et al., 2011, provide an exhaustive summary and taxonomy). Unfortunately, state-trace analysis is unfamiliar to many researchers who could benefit from its use. Prince et al. attempted to make it more accessible by providing a general methodology for designing and fine-tuning state-trace experiments, as well as statistics suitable for the type of bounded data where state-trace analysis is often needed. Their Bayesian approach guides the fine-tuning of experimental designs as well as inference about monotonicity, and hence latent dimensionality.

Prince et al. (2011) argued that their Bayesian methods are particularly suited to state-trace analysis for several reasons. First, for accuracy data they require only a minimal additional assumption (that the binary data are binomially distributed), so detract little from the relatively assumption-free nature of state-trace analysis. Second, hypotheses about different dimensionalities vary tremendously in their ability to fit data by chance. Bayesian methods take this into account, and so do not inappropriately favour more flexible models. Finally, and in contrast to null hypothesis testing, Bayesian analysis can quantify evidence in favour of a simpler "null" (e.g., a one-dimensional) model, as well as evidence against it. State-trace analysis has been described as an example of an “equivalence” method that treats the discovery of simplicity and difference as equally important (Loftus, 2002) and so ideally requires a correspondingly even-handed statistical method.

In this paper we describe the scope and capabilities of **StateTrace**, a package written for the freely available R language (R Development Core Team, 2011), that implements Prince et al.’s (2011) methods using computationally intensive posterior sampling to perform Bayesian estimation and model selection. R is freely available for Windows, Mac OS X and Linux, and can be downloaded from http://www.r-project.org/. **StateTrace** is designed for users with no experience implementing Bayesian analyses. Functions can be accessed through a GUI, so it is also suitable for users with relatively minimal experience of R. The GUI functionality is provided by Hoffman and Laird's (2009) **fgui** package. We provide a detailed step-by-step guide on how to perform an analysis of two example data sets in the “vignette” document included in the **StateTrace** package. This vignette was developed based on our experience deploying the package with research students and can be accessed within R by typing `vignette(topic = "StateTrace", package = "StateTrace")` on the R command line after installing the package by typing `install.packages("StateTrace").`

For experienced users who can implement the relatively straightforward sampling methods it employs, **StateTrace** is convenient because it provides facilities that manage potentially large demands on computer time and memory. With these facilities analysis of appropriately designed experiments is practical on commonly available personal computers. **StateTrace** also provides convenient tabular and graphical methods for examining and summarising results, including customizable state-trace plots, that address both the individual and group levels of analysis. In this paper we provide an overview of the capabilities and limitations of **StateTrace**. In the next section we discuss the types of experimental designs and data types that it accommodates, and in the following section we describe its statistical and graphical capabilities.

# Design and Data

**StateTrace** can analyse three-factor (2x2xN) repeated-measures experiments that yield a binary dependent measure, including measures calculated either with or without reference to a measured baseline. Accuracy quantified by the difference between a hit rate and false alarm rate are examples of the former type of measure. Proportion correct is an example of the latter type of measure. As state-trace analysis is not affected by monotonic transformation, equivalent results are produced for related measures, such as *d*' from signal detection theory. The example data sets presented in this paper are taken from recognition memory experiments and address both types, as measured by testing single items that were either studied or not ("yes-no" testing), or by two-alternative forced choice (2AFC) testing.

Prince et al. (2011) label the three factors as state, dimension and trace factors. In this section we explore the nature of each type of factor and the types of designs that **StateTrace** can and cannot analyse. A global limitation is that all three factors must be of the repeated measures type, as Prince et al.'s (2011) methods analyse each participant's data separately, and then combines the individual results to address the group level.

## *State and Dimension Factors*

The state factor has two levels that constitute the axes of the state-trace plot. In our example data the two levels are based on recognition memory decisions made about pictures of houses vs. pictures of faces. In the case of a dependent-variable state trace analysis these levels correspond to different binary dependent measures. For example, they might be the outcome of a recognition decision and a classification of the decision as being made with high vs. low confidence.

The dimension factor has levels corresponding to different experimental manipulations. In our examples, the manipulation corresponds to whether the study and/or test presentations used upright or inverted images. The dimension factor can be thought of as interacting with the state factor in a way that changes the dimensionality of the processes underlying performance. For example, upright faces are thought to be able to be encoded both in terms of their constituent features and in a more holistic or relational way, whereas only the former encoding is available for inverted faces (Maurer, LeGrand & Mondloch, 2002; Valentine, 1988; Yin, 1969).

The state and dimension factors correspond to the two factors examined in a dissociation analysis. For a dependent-variable state-trace analysis assignment of the two factors to state and dimension roles is clear, but otherwise these roles can be exchangeable. An exception concerns measures assessed against a baseline. For such measures the levels of the dimension factor must have a common baseline for state-trace plot monotonicity to be diagnostic of dimensionality. As this restriction does not apply to the state factor assignment can be made accordingly (as was the case with our baseline example) or a measurement method that does not include a baseline must be used (e.g., 2AFC). One of our examples has no baseline (a “B0” design) the other has a separate baseline for each level of the state factor (a “B2” design). In the latter case the baseline conditions correspond to test house or face images that were not studied.

**StateTrace** assumes the dimension factor has two levels based on practical motivations related to the size of (i.e., number of cells in) a design. First, the computational method used by **StateTrace** for the Bayesian analysis rapidly becomes prohibitive as size increases. Second, for large designs it is difficult to run enough trials per cell to obtain estimates that are sufficiently precise to support individual participant analysis. Hence, using a two-level dimension factor is prudent, perhaps after piloting to select appropriate levels, as this is typically sufficient to test dimensionality.

### *Trace Factor*

The third experimental manipulation, the trace factor, has no analogue in dissociation analysis. In Prince et al.’s (2011) method the effect of the trace factor on dimensionality is not of interest. Indeed, the trace factor is chosen specifically because past research indicates it is unlikely to affect dimensionality, so that dimensionality evidence can be unambiguously interpreted in terms of the state and dimension factor effects. In our examples the trace factor was the time spent studying each item. Given past research indicates that increasing study time increases accuracy for houses and faces both when they are upright and when they are inverted, study time should have a monotonic effect in the state-trace plot. For the same reasons as apply to the dimension factor, the trace factor cannot have different baselines for different levels.

The trace factor's role is to ensure that state-trace analysis is diagnostic of dimensionality. A state-trace plot can be non-diagnostic for two reasons. When state and dimension factors have only two levels and there is no trace manipulation the state-trace plot has only two points, and so is always monotonic. If the plot has more than two points it can still be non-diagnostic if the dimension factor effect is so large that results do not overlap on either axis of the plot. The remedy is to induce overlap using the trace factor. In our examples, greater accuracy for the upright condition is counteracted by a longer study time for the inverted condition, inducing overlap.

**StateTrace** can accommodate any number of trace-factor levels. Prince et al. (2011) recommend three to four levels with evenly spaced effects. Too many levels can cause the sort of design-size related problems discussed previously. Too few levels, and unevenly spaced effects, risk the state-trace plot being non-diagnostic due to an unlucky configuration of estimated points even when overlap is achieved. Choosing an appropriate number and spacing of trace factor levels can require some calibration through pilot testing. When doing so it is important to recognize that it may be most efficient to use different trace-factor levels within each level of the dimension factor (i.e., the design need not be fully factorial). For example, we used generally longer study times in the inverted than upright condition in order to counteract generally greater accuracy for upright than inverted items. A trace factor can sometimes be created by the post-hoc construction of a factor according to criteria set to obtain the most diagnostic outcome. For example, Heathcote, Freeman, Etherington, Tonkin and Bora (2009) constructed a trace factor in post-hoc in a recognition memory experiment by dividing test trials on study-test interval.

### *Input Data Formats*

**StateTrace** reads in text data files made up of equal numbers of entries on each line. Data files may contain *trial data*, with a column indicating the response on each trial, or *summarized* *data*, with columns indicating the number of correct responses and number of trials for each condition. **StateTrace** accepts either a single file with all participants' data or individual participant files. Columns can be delimited in a variety of common ways, and relevant columns can be selected from a larger set either by name or number. The relevant columns indicate the response (number of trials for summarized data), and the state, dimension and trace levels, for each row.

**StateTrace** accepts data files formatted as coming from designs either with no baseline (B0) or with a different baseline condition for each state level (B2). As baseline conditions cannot differ over dimension and trace levels a B2 design is automatically detected through the presence of blank or 'NA' entries in these columns. Designs where accuracy in all conditions is measured relative to a single baseline can be treated as coming from either a B0 or B2 design. In the former case the baseline data are omitted, whereas in the latter case they are are included twice. The outcome of state-trace analysis is the same in both cases, but the B2 format may be preferred as accuracy results can be displayed by differences between non-baseline and baseline results.

## Statistics and Graphics

Inference about dimensionality based on visual inspection of state-trace plots (e.g., Figure 1) can sometimes be misleading due to measurement noise. This is particularly the case for individual participant data (e.g., Figures 1a and 1b) where levels of measurement noise can be high. However, individual analysis is required to make strong inferences based on state-trace analysis because neither the monotonicity nor non-monotonicity of state-trace plots is necessarily preserved when they are averaged over participants (Prince et al., 2011). Hence, a quantitative approach to state-trace analysis provides an important complement to visually assessing a state-trace plot.

**StateTrace** obtains evidence about questions relevant both to refining an experimental design and diagnosing dimensionality by selecting among four mutually exclusive models. In defining these models it is convenient to refer to 'data traces', lines joining data points from the same level of the dimension factor (e.g., the solid and dashed lines in Figures 1a and 1b).

1) *Non-trace model:* The trace factor does not always have a monotonic effect (i.e., one or

more data traces are non-monotonic) and so any non-monotonicity in the state-trace plot cannot be unambiguously attributed to the interaction between the state and dimension factors.

2) *No-overlap model:* Data traces do not overlap on either axis, so that even though the state-trace plot is monotonic no conclusions can be made about dimensionality.
3) *Uni-dimensional model:* A single latent variable mediates performance; that is, the state-trace plot is monotonic and provides a valid basis for inference about dimensionality.
4) *Multi-dimensional model:* More than one latent variable mediates performance; that is, the state-trace plot is non-monotonic.

Each model specifies a set of order restrictions on the points in a state-trace plot. Evidence for each model is quantified using a Bayes factor (BF; Kass & Raftery, 1995). The BF quantifies the change in relative beliefs (odds) about two models caused by observing the data. That is, it is the change from prior odds (odds before seeing the data) to posterior odds (odds after seeing the data). For example, if two models, M1 (the numerator model) and M2 (the denominator model), are initially considered equally likely, BF = 10 implies M1 is ten times more likely than M2 after observing the data.

Bayes factors can be thought of as measures of relative goodness-of-fit that compensate for differences in model flexibility. Our denominator (M2) is always an *encompassing* model under which all orders are equally likely. Hence, BF > 1 favours the less flexible order-restricted (numerator) model over the encompassing model, which by definition fits any data perfectly. If it is assumed that one model amongst a set of models generated the data (i.e., is the *true* model) Bayes factors for the set of models can be combined to calculate posterior model probabilities, *p*, which quantify relative evidence. High probabilities provide evidence favouring a model and low probabilities against. If the true-model assumption is not made, a posterior probability still provides a number that quantifies relative evidence on an easy to interpret zero-to-one scale, even though it cannot be interpreted as a probability. Additionally, when the trace factor is selected based on strong prior evidence that it has a monotonic effect it makes sense to exclude the non-trace model when computing posterior model probabilities.

**StateTrace** can output both Bayes factors, to quantify absolute evidence about whether the data provide sufficient support for clear conclusions about each model, and posterior model probabilities (calculated either including or excluding the non-trace model) to quantify relative evidence. We suggest evidence for the numerator model can be considered weak for a BF < 3, positive for a BF between 3 and 20, strong for a BF between 20 and 100, and very strong for a BF > 100. Table 1 provides similar conventions for posterior model probabilities. The two sets of conventions are closely related because $p$ = BF / (1 + BF), given BF = 1 by definition for the encompassing model. Like all conventions, these suggestions should not be used uncritically; Bayes factors and posterior model probabilities have a natural scale making their interpretation straightforward.

### *Computing Bayes Factors*

Prince et al. (2011) used posterior sampling methods proposed by Klugkist, Kato and Hoijtink (2005) and Klugkist, Laudy and Hoijtink (2005) to compute Bayes factors for the four models. Although conceptually straightforward (i.e., they simply count the frequency with which different orders occur in a set of simulated samples), this method is computationally expensive. In order to make it practical, **StateTrace** uses two types of sampling:

1) *Encompassing sampling*: Monte-Carlo (MC) sampling, used to obtain samples from a model that makes no order constraints, and
2) *Trace model sampling*: Markov-Chain Monte-Carlo (MCMC) sampling, used to obtain samples under the trace model constraint.

In principle only MC methods are needed, but the yield of samples relevant to models (2)-(4) is often so low that it would be far too inefficient. Although slower per sample, the MCMC method always yields samples from models (2)-(4) after an initial “burn-in” period, and so is better in practice.

For trace model sampling, initial (burn-in) samples are discarded because it can take some time for the MCMC process to converge to the target distribution (see Gilks, Richardson, & Spiegelhalter, 1996). Our experience is that convergence is very fast and that at most 100 initial samples need be discarded. However, this may not be the case in all applications, so **StateTrace** provides facilities to check. **StateTrace** uses Plummer, Best, Cowles and Vine's (2006) **coda** package to automatically calculate one check, Gelman’s “R-hat” statistic (values close to one indicate convergence), and also allows MCMC samples to be exported in a format suitable for further checks provided by **coda**.

**StateTrace** manages the sampling process, enabling estimates to be automatically refined to a specified level of accuracy, and for the time spent sampling to be limited to convenient periods (e.g., overnight runs). Statistics to support simultaneous selection among all four models (an *exhaustive strategy*) or only models (2)-(4) (a *trace-true strategy*) are calculated automatically and raw counts can also be accessed to support other approaches, such as sequential model selection (see Prince et al., 2011).

## Overview of Functions

Typing `guista()` at the R command line invokes the main **StateTrace** GUI, which provides access to the main **StateTrace** functions: **stFirst**, **stSample**, **stSummary**, **stProbplot**, **stBootav**, **stPlot** and **staManage**. Alternatively, the GUIs for each separate function can be called by typing “gui” followed by the function name and parentheses (e.g., `guistFirst()`). In general, command line users can remove "gui" from the start of the function and enter argument values within the parentheses. Details on available arguments can be obtained by consulting the function’s help, called by typing a ? prior to the function name.

The **stFirst** function performs an initial analysis, first reading in data for one or more participants from one or more text files, and then making a quick preliminary assessment of the results based on a limited number of posterior samples. It then creates an object of class *sta* in the R environment, which is named by the user. The *sta* object encapsulates the data and the numerical results based on sampling. Once **stFirst** is complete the *sta* object can be saved from the R environment to a file in a compressed format and restored in a later R session using the R `save` and `load` functions respectively.

Three **StateTrace** functions allow the contents of an *sta* object to be displayed. The **stSummary** and **stProbplot** functions provide, respectively, tabular and graphical summaries of the Bayesian analysis. The **stPlot** function makes state-trace plots (e.g., Figure 1). All three functions work, to some degree, with an *sta* object created by **stFirst**. However, for more accurate results from the Bayesian analysis, and to access the full range of state-trace plot options, two other functions may have to be run: **stSample** and **stBootav**. These functions perform time-consuming computations whose results are stored in the *sta* object. The **stSample** function collects enough extra samples to reach a specified level of accuracy in the Bayesian analysis. Some of these extra samples can also be stored in the *sta* object so that a line representing the trace (e.g., Figure 1c) or monotonic (Figure 1d) model that is best supported by the data can be added to a state-trace plot. However, to enable these lines to be added to state-trace plots averaged over participants, the **stBootav** function must also be run.

The main GUI also provides access to the **staManage** function, which allows users to manage and export posterior samples stored in an *sta* object. An *sta* object is an R list that can be directly accessed by users, but **staManage**, and the other functions, are designed so this should not be necessary. Samples are stored in an *sta* object to enable efficient generation of graphical summaries of uncertainty in estimation (i.e., credible regions, the Bayesian analogue of confidence intervals, e.g., the ellipses in Figure 1). However, this can sometimes cause an *sta* object to become so large that it is slow to load and save. The **staManage** can be used to remove samples after the graphics have been generated. The **staManage** function also allows users to export a list containing posterior samples with one entry for each participant. Each entry in the list contains samples in a format (the *mcmc.list* class) for which **coda** provides many easy to use analysis methods (e.g., **plot** and **summary** functions).

### *First Steps*

Before **stFirst** initiates sampling, checks are run to ensure the selected data files are compatible with **StateTrace** (e.g., state and dimension factors each have only two levels) and the multiple files are compatible with each other (i.e., all have either a B0 or a B2 design). It then obtains 100,000 MC samples for each independent set of design cells (four sets for B2 designs and two for B0 designs) and determines the proportions that follow the order specified by the trace factor. Estimates of 95% credible intervals for each proportion estimate are obtained, and where the precision of the interval is less than 0.0005, it marks sampling as complete. Otherwise it estimates the time required to get enough samples to narrow the interval sufficiently, based on the time required for the initial 100,000. These time estimates are only approximate and will vary, especially if sampling is completed on a different computer that is faster or slower.

Next an order constrained Gibbs sampler (Gelfland, Smith & Lee, 1992) is used to draw two sequences of 5,000 MCMC samples (“chains”) from the trace model for each participant. The proportions of samples following the no-overlap, uni-dimensional and multi-dimensional model orders are tabulated, and the corresponding 95% credible intervals calculated. Sampling for a participant is marked as complete if all intervals are less than 0.005. Otherwise the additional time required to complete is estimated. A smaller interval criterion is used for encompassing than trace sampling as encompassing proportion estimates have greater potential to reduce precision overall because they multiply the trace proportions in the calculation of Bayes factors. Once the two chains are complete, **stFirst** reports whether the MCMC process has worked properly (i.e., has “converged”) using Gelman’s multivariate R-hat statistic.

In the final stage of computation **stFirst** draws 10,000 bootstrap average samples and uses the two-dimensional density estimator provided by Wand and Ripley’s (2009) **KernSmooth** package (with its default parameters) to calculate the posterior modes (measures of central tendency) and 68% credible regions (i.e., the analogue of a standard error) around the modes. These calculations are used to display an average state-trace plot, which provides the user with an immediate view of results averaged over participants, as well as state-trace plots for each individual participant. Additionally, the corresponding posterior mode estimates are output in a tabular form to the R console. The **stFirst** function can be called repeatedly to add additional participants to the *sta* object. A warning will be issued if duplicate data sources are mistakenly specified; however, this data will still be added to the object without replacing the old entries.

### *Refining Estimates*

Further sampling may be required if the analysis did not reach completion during the initial pass or the user wishes to alter the credible interval precision criteria from those used by **stFirst**. This is achieved using **stSample**, which allows fine-grained control over the defaults used by **stFirst**. The only required input is the name of an existing *sta* object. Because obtaining enough samples to fulfil stricter criteria can be time consuming **stSample** has a ‘refresh’ mode, which allows the predicted time to completion to be calculated for different criteria. This refresh mode is fast to run as no actual sampling is done, unless none has been done yet, in which case a single pass is made. As the time information will vary depending on the computer used, if the most recent pass (e.g., obtained by running **stFirst**) was on a different computer it is useful to turn off the refresh mode but leave the maximum run time at zero; this will cause a single pass to be run and to update the timing information for the new computer.

Once a sampling plan is determined the refresh mode can be turned off, a suitable maximum run time entered and sampling initiated. Sampling is completed in a series of passes, and users may choose to sample only from the encompassing model, trace model, or both. Sampling terminates when precision criteria are satisfied, so **stSample** may complete before the maximum time elapses and one type of sampling may complete before another. The *verbose* argument (with values 0, 1, or 2) controls information printed to the R console during sampling: 0 is silent, 1 prints the estimated total time remaining after each run for all participants, and 2 adds timings per participant.

The number of samples for each pass of each type of sampling is chosen to satisfy a trade-off. Using a small number per pass inherits a cost in housekeeping between passes and initial *burn-*

*in* samples on each trace-model run are lost. A large number uses more RAM and can result in more samples being taken than required to achieve the required precision. A larger value is advisable for encompassing than trace model sampling, as encompassing sampling is usually an order of magnitude faster. In our applications we have found the defaults work well and that little is gained in particular cases by altering them. Similarly, we have found the default criteria (credible interval type, e.g., 95%, and precision) strike an appropriate balance between computation time and the accuracy of Bayes factor and posterior model probability estimates.

A second reason for running **stSample** is to collect samples that enable visualization of each model; that is, samples that follow the order(s) dictated by a model. We have found the default value of 10,000 encompassing samples collected by **stFirst** is sufficient for accurate visualization of central tendencies and credible regions, although large regions (which require estimation of distribution tails) can require more. The 10,000 trace samples collected by **stFirst** are also usually more than sufficient given they are only used to estimate central tendency. The **stSample** function also collects a particular type of trace sample, monotonic model samples, which may be relatively rare, especially when the data are far from monotonic. Monotonic samples are used to plot the central tendencies of the uni-dimensional or no-overlap models, with the latter type of sample often being extremely rare unless the data are strongly non-overlapping. Given this, by default **stSample** keeps all monotonic samples.

Storing large sets of samples for each participant can greatly increase the size of *sta* objects. The default values (assuming stored monotonic samples are not allowed to grow too large) are rarely problematic. The **staManage** function can be used to reduce the number of stored samples where problems arise, including the ability to keep only samples for the “best” (i.e., most frequently occurring, and hence most probable) monotonic order rather than all samples with monotonic orders. The **staManage** function can also be used to join multiple *sta* objects; for example, it can be computationally efficient to divide a very large set of participants and then run the sampling for sub-groups of participants on separate machines, after which the *sta* objects can be combined and the group aggregate results examined.

### *Extracting Results*

The **stSummary** function produces tabular model-selection results to the R console. It also provides information about the status of sampling for an *sta* object (i.e., whether it is complete or if not, how much more computation time is required according to the criteria stored in the object). Results can be output as Bayes factors or posterior model probabilities based on either exhaustive or *trace-true* strategies. By default, **stSummary** reports results summarized over participants based on group Bayes factors, which are the product of each participant's Bayes factors and assumes each participant contributes independent evidence (Prince et al, 2011). However, in some cases the group results can be inappropriately influenced by outlying individual participant results (e.g., most participants are uni-dimensional but a few strongly multi-dimensional participants dominate the group results). We recommend that users also output and examine individual participant results in order to check this possibility. This can be done, and outlier participants can then be excluded from the calculation of group results, using **stSummary** options. Users may also output a large range of additional results, including the prior probabilities for each model (calculated analytically, see Prince et al., 2011), the total number of encompassing and trace model samples and counts of the number of times the orders specified by each model were sampled.

The **stProbplot** GUI allows the distribution over participants of posterior model probabilities to be inspected graphically. Outlying participants can be excluded, and the annotation within the plots customised. Additionally, the posterior probability for the group (based on the group Bayes factor) can be displayed numerically in the title and as a line on the plot. As shown in the output for the 2AFC example data in Figure 2, the **stProbplot** plot includes a panel for each model. Participants in each panel are sorted by their results, allowing those with extreme values to be easily identified. Figure 2 was made using defaults, which produce appropriate annotations in most cases, and by choosing the option to use letter plot symbols in order to easily identify each participant's results.

Inspection of Figure 2 suggests that participant ‘J’ provides an outlying result in favour of the non-trace model. However, follow-up analysis excluding participant ‘J’ revealed little influence on the group posterior model probability (*gp* in the panel titles). Overall these results show positive or greater evidence for the trace model and for data-trace overlap (i.e., low probabilities for the non-trace and no-overlap models). Evidence is weaker and individual variability greater in relation to the dimensionality results, but the group evidence supports a multi-dimensional outcome. The **stProbplot** plots can also reveal a potential mixture of uni-dimensional and multi-dimensional sub-groups, but that is not indicated in Figure 2.

Figure 1 shows examples of state-trace plots produced by **stPlot**. The **stPlot** function can represent accuracy data by the mode, mean or median measures of central tendency applied to samples from the encompassing model. As the encompassing model makes no order assumptions these measures (e.g., large symbols in Figure 1) provide a model-free estimate of the observed data. The default choice used to create Figure 1 (the mode) produces estimates that are usually equivalent to the familiar maximum-likelihood estimator (e.g., $n/N$ for the 2AFC hit rate, where $n$ is the number of correct responses in $N$ trials).

The other central tendency measures usually produce similar results, at least for reasonable sample sizes not subject to floor or ceiling effects. For example, for the hit rate and uniform prior used by **StateTrace**, the mean of a large sample from the encompassing model is equivalent to $(n+1)/(N+1)$. For other accuracy measures such simple formulae are not available. This is also the case for any accuracy measures for any of the order-restricted models. Hence, estimates based on samples have the advantage of providing an easily applied and general approach.

The **stPlot** function uses the same type of approach to display the degree of uncertainty in central tendency estimates, by drawing contours around regions containing a specified percentage of the posterior encompassing-model samples. Estimating Bayesian credible regions in this way works with all accuracy measures in a way that takes account of any floor and ceiling effects, which can be very influential when contours are near bounds in an accuracy measure. The regions, and modes, are estimated using the **bkde2D** function in Wand and Ripley’s (2009) **KernSmooth** package, which is included by default with R. Users can choose the percentage contained by the regions and the degree of smoothing, as a multiple of the maximum over data points of the values provided by the **dpik** function: this **KernSmooth** function, and **bkde2D**, are called with default values. The default multiplier of five used by **stPlot** was chosen to produce very smooth contours even for large regions, which can otherwise be irregular because they require estimation of distribution tails; users are encouraged to experiment with the multiplier as appropriate for their application.

Figure 1a and 1b plots results for participant ‘J’, who had the strongest evidence for a violation of the trace model in Figure 2, and participant ‘R’, who had the strongest evidence for the multi-dimensional model; both state-trace plots are clearly consistent with the model-selection analyses. Individual participant data are typically quite noisy, so for clarity the credible regions in these plots contain only 50% of the posterior samples. Both data sets display strong data-trace overlap, consistent with the results for the no-overlap model in Figure 2.

Figure 1c is a state-trace plot of the average over all participants in the 2AFC example data. Reflecting the reduction in uncertainty associated with an average, the credible regions are much smaller in this case, even though they contain the default value of 68% of the posterior samples. The lines in Figure 1c join the modes of the average of samples from the trace model. The points joined by the lines are different from the large symbols, which are estimated based on encompassing model samples, as the encompassing model admits samples that violate the trace model. However, the difference is not large, reflecting the fact that for most participants the trace model provides an excellent description of this data.

Figure 1d plots average results for the yes-no example using the signal-detection theory *d*’ measure of accuracy. The lines in Figure 1d join the modes of the most commonly occurring order[1]

[1] It is important to note that the best order may differ between participants. Before interpreting the best (most frequently occurring) monotonic model in the average, such as is plotted in Figure 1d, it is advisable to use **stSummary** to examine the degree of variability in the best orders over participants, as strong individual differences may mean that taking an average is not sensible.

for monotonic MCMC samples (the *best* monotonic model); that is, MCMC samples from either the uni-dimensional or no-overlap models. Because this data is well described by a one-dimensional model the difference between the best monotonic model and encompassing model modes is relatively small.

Figures 1c and 1d were created after first running **stSample** to the default criterion then **stBootav** to average the trace and monotonic samples stored by **stSample**. The **stBootav** function allows users to choose to calculate bootstrap averages (based on the stored samples for each participant) from one or more of the encompassing, trace and monotonic models. A set of bootstrap averages is created by repeatedly randomly selecting with replacement one sample from each participant's set of posterior samples for a given model and taking their mean. Each time **stBootav** is invoked it can compute averages for only one type of accuracy.

Monotonic samples may be relatively rare for some data (e.g., the 2AFC data, as it is strongly non-monotonic), and so the averages can be unreliable; to alert users to this possibility **stBootav** reports the number for participants who have less than 100 samples. A participant with no monotonic samples is automatically excluded from the average. The problem of a lack of monotonic samples might be addressed by calling **stSample** again with a stricter criterion, but usually a lack of monotonic samples indicates the monotonic model is not appropriate for the data and so there is no point plotting it.

Once averages are stored in an *sta* object, **stPlot** can make a corresponding average state-trace plot (e.g., Figure 1d). Data points in average plots represent the central tendency of the set of bootstrap averages. Variability among the averages is used to construct credible regions in the same way as for individual participants. These credible regions reflect uncertainty in the estimated average over the particular set of participants in an experiment. The **stBootav** function also allows participants to be selected at random with replacement on each bootstrap repetition; this produces a set of averages with the same central tendency but greater variability that is appropriate when the participants are treated as a sample from a population.

# Limitations and Future Directions

**StateTrace** is limited in a number of ways that we plan to address: it works with only fully repeated measures design, allows only two levels for state and dimension factors, and works with only binary data. First, it is both valid and useful to perform a state-trace analysis involving between-subjects factors (e.g., does the dimensionality of memory differ between amnesiacs and controls?). However, statistical analysis of such state-trace plots requires estimation of a population level model, and so would require a hierarchical extension of Prince et al.'s (2001) approach. Bayesian methods are suited to this extension and we hope to pursue it in future work. Until then we recommend Verhaeghen and Cerella's (2002) multilevel to address such designs, although it requires the assumption of a functional form (e.g., linear) for data traces.

In principle, both state and dimension factors may have more than two levels. We are currently testing a fast approximation for the Bayesian analysis, which removes one of the two reasons for the latter restriction. Hence, it will likely be removed as part of the update incorporating the fast approximation. An extension beyond two state levels requires more fundamental change, although applicable methods have been developed less (Dunn & James, 2003).

For now, where users are interested in the dimensionality underlying the relationship between three state-factor levels (e.g., A, B and C) or more they can use **StateTrace** by taking advantage of the transitivity of pairwise state-trace inference (e.g., test A vs. B and B vs. C, with one-dimensional results in both cases indicating a single latent variable explains variation in all three). A similar approach can be required if more than two dimension-factor levels are required.

Finally, the limitation to binary data can be accommodated by collapsing (e.g., a 1-10 confidence rating could be collapsed to high vs. low), but subject to the usual caveats about loss of information. We plan to explore the extension of Prince et al.'s (2011) Bayesian analysis, and **StateTrace**, to finer-grained measures using multinomial data generating assumption.

# References

Bamber, D. (1979). State-trace analysis: A method of testing simple theories of causation. *Journal of Mathematical Psychology, 19,* 137-181.

Bogartz, R. S. (1976). On the meaning of statistical interactions. *Journal of Experimental Child Psychology, 22,* 178-183.

Busemeyer, J.R. & Jones, L.E. (1983). Analysis of multiplicative combination rules when the causal variables are measured with error. *Psychological Bulletin, 93,* 549-562.

Dunn, J.C. (2003). The elusive dissociation. *Cortex, 39,* 177-197.

Dunn, J.C. (2004). Remember-know: A matter of confidence. *Psychological Review, 111,* 524-542.

Dunn, J.C. (2008). The dimensionality of the remember-know task: A state-trace analysis. *Psychological Review, 115,* 426-446.

Dunn, J. C. & James, R. N. (2003). Signed difference analysis: Theory and application. *Journal of Mathematical Psychology, 47,* 389-416.

Dunn, J.C., & Kirsner, K. (1988). Discovering functionally independent mental processes: The principle of reversed association. *Psychological Review, 95,* 91-101.

Gelfland, A.E., Smith, A.F.M. & Lee, R.-M. (1992). Bayesian analysis of constrained parameter and truncated data problems. *Journal of the American Statistical Association, 87,* 523-532.

Gilks, W. R., Richardson, S., & Spiegelhalter, D. J. (Eds.) (1996). *Markov chain Monte Carlo in practice*. Boca Raton, FL: Chapman & Hall/CRC.

Glanzer, M., & Cunitz, A.R. (1966). Two storage mechanisms in free recall. *Journal of Verbal Learning and Verbal Behavior, 5,* 351-360.

Heathcote, A., Freeman, E., Etherington, J., Tonkin, J. & Bora, B. (2009). A dissociation between similarity effects in episodic face recognition. *Psychonomic Bulletin & Review, 16,* 824-831.

Henson, R. (2006). Forward inference using functional neuroimaging: Dissociations versus associations. *Trends in Cognitive Sciences, 10,* 64-69.

Hoffman, T.J., Laird, N.M. (2009). fgui: A method for automatically creating graphical user interfaces for command-line R packages. *Journal of Statistical Software, 30,* 1-14. URL http://www.jstatsoft.org/v30/i02/.

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

Klugkist, I., Kato, B., & Hoijtink, H. (2005). Bayesian model selection using encompassing priors. *Statistica Neerlandica, 59,* 57-69.

Klugkist, I., Laudy, O., & Hoijtink, H. (2005). Inequality constrained analysis of variance: A Bayesian approach. *Psychological Methods, 10,* 477-493.

Loftus, G.R. (1978). On interpretation of interactions. *Memory & Cognition, 6,* 312-319.

Loftus, G.R. (1996). Psychology will be a much better science when we change the way we analyse data. *Current Directions in Psychological Science, 5,* 161-171.

Loftus, G. R. (2002). Analysis, interpretation, and visual presentation of experimental data. In H. Pashler (Ed.), *Stevens' handbook of experimental psychology* (Vol. 4, pp. 339-390). New York: John Wiley and Sons.

Loftus, G.R., Oberg, M.A., & Dillon, A.M. (2004). Linear theory, dimensional theory, and the face-inversion effect. *Psychological Review, 111,* 835-863.

Maurer, D., LeGrand, R. & Mondloch, C.J. (2002). The many faces of configural processing. *Trends in Cognitive Sciences, 6,* 255-260.

Newell, B.R., & Dunn, J.C. (2008). Dimensions in data: Testing psychological models using state-trace analysis. *Trends in Cognitive Sciences, 12,* 285-290.

Plummer, M., Best, N., Cowles, K., & Vines, K. (2006). CODA: Convergence diagnosis and output analysis for MCMC. *R News, 6,* 7-11.

Poldrack, R.A. (2006). Can cognitive processes be inferred from neuroimaging data? *Trends in Cognitive Science, 10,* 59-63.

Prince, M., Brown, S. & Heathcote, A. (2011). The design and analysis of state-trace experiments. *Psychological Methods.* Advanced online publication.

R Development Core Team. (2011). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Raftery, A.E. (1995). Bayesian model selection in social research. *Sociological Methodology, 25,* 111-163.

Shallice, T. (1988). *From Neuropsychology to Mental Structure*. New York: Cambridge University Press.

Teuber, H-L. (1955). Physiological psychology. *Annual Review of Psychology, 6,* 267-296.

Valentine, T. (1988). Upside-down faces: A review of the effect of inversion upon face recognition. *Journal of British Psychology, 79,* 471-491.

Verhaeghen, P. & Cerella, J. (2002). Aging, executive control, and attention: A review of meta-analyses. *Neuroscience and Biobehavioral Reviews, 26,* 849–857.

Wand, M., & Ripley, B. (2009). KernSmooth: Functions for kernel smoothing for Wand and Jones (1995) “Kernel Smoothing”. R package. http://cran.r-project.org.

Yin, R.K. (1969). Looking at upside-down faces. *Journal of Experimental Psychology, 81,* 141-145.

# Author Notes

Thanks to Cassandra Barrett, Amanda Brown, Christopher Brown, Sara Buller, Anna Charley, Nekia Dalley, Stuart Donaldson, Katie Foster, Jessica Gordon, Alice Grady, Tonelle Handley, Georgia Kelaher, Elaine Yheng Ching Loke, Katherine Moore, Soo Li Quah, Kate Smith, Evelyn Tan and Lucy West for assistance with stimulus preparation and running of participants for the example data sets.

# Tables

Table 1.

*Conventions to aid interpretation of the posterior model probabilities (after Raftery, 1995).*

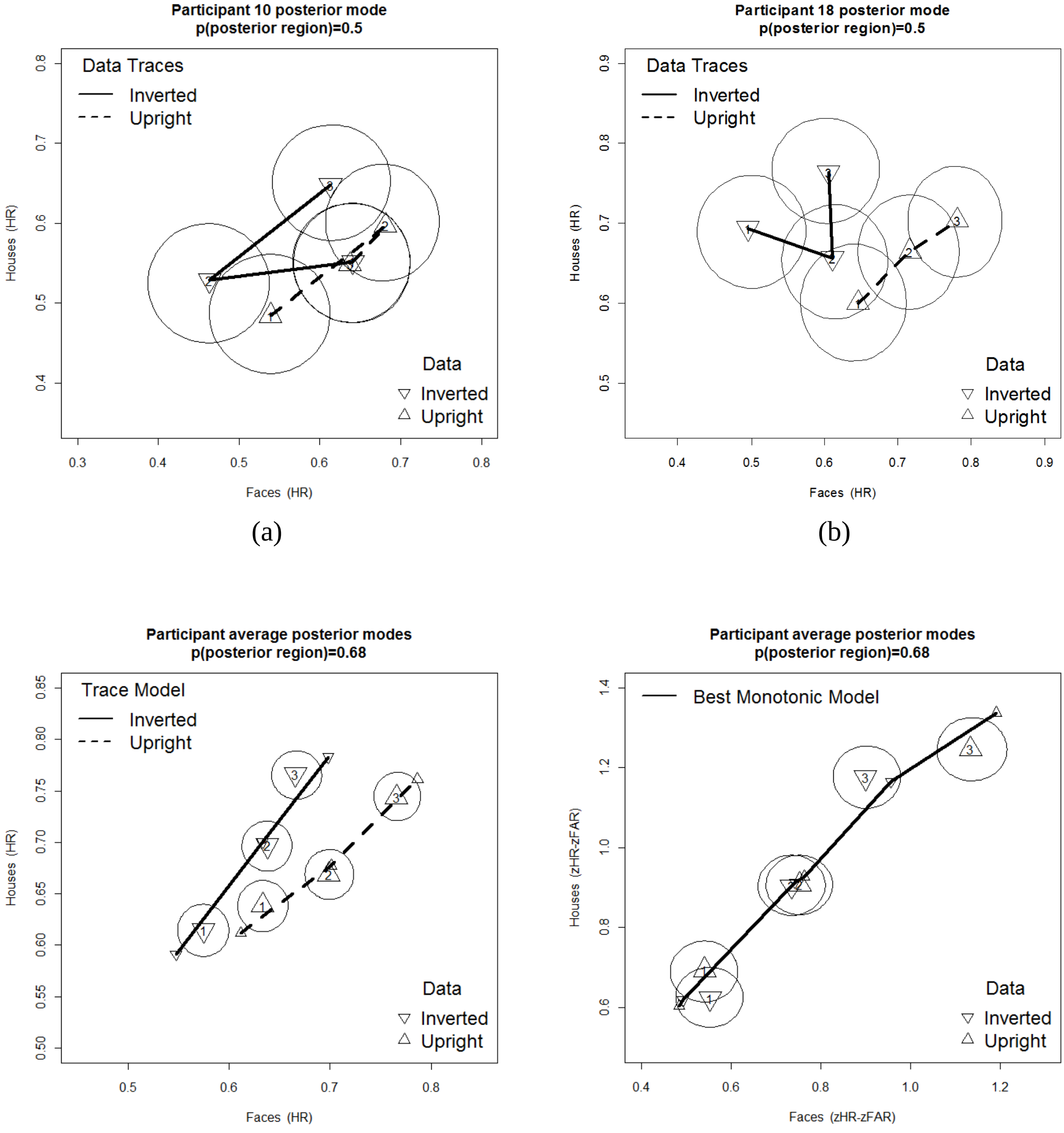
| Favouring model | | Against model |
|---|---|---|
| $p > .99$ | Very strong evidence | $p < .01$ |
| $.95 < p \leq .99$ | Strong evidence | $.01 \leq p < .05$ |
| $.75 < p \leq .95$ | Positive evidence | $.05 \leq p < .25$ |
| | $.25 \leq p \leq .75$ Equivocal evidence | |

# Figure Captions

*Figure 1.* Accuracy state-trace plots showing modes of the posterior estimates from the encompassing model (large symbols) for the 2AFC data set for (a) participant ‘J’, (b) participant ‘R’, and (c) on average, as well as (d) on average for the yes-no data set. Accuracy is indicated by plotting the hit rate (HR) for the 2AFC data and *d’* (i.e., *z*(HR) – *z*(FAR), where FAR is the false alarm rate) for the yes-no example. For (a) and (b) the lines are data traces, joining posterior modes of the encompassing model samples, and ellipses represent the 50% credible regions. For (c) lines with small symbols join posterior modes of the trace model and for (d) they join the highest posterior probability (i.e., most frequently sampled) monotonic model. The ellipses in (c) and (d) represent 68% credible regions.

*Figure 2.* Posterior model probabilities for each participant (denoted by letters) for the 2AFC example data for each of the four diagnostic models (panels). Group posterior model probabilities for each model are indicated in panel titles and plotted as a heavy dashed line. Within each panel, participant results are sorted in ascending order of their probability estimates and faint dashed lines demarcate the categories in Table 1.

# Figures



(a) (b)

(c) (d)

**Figure 1.**

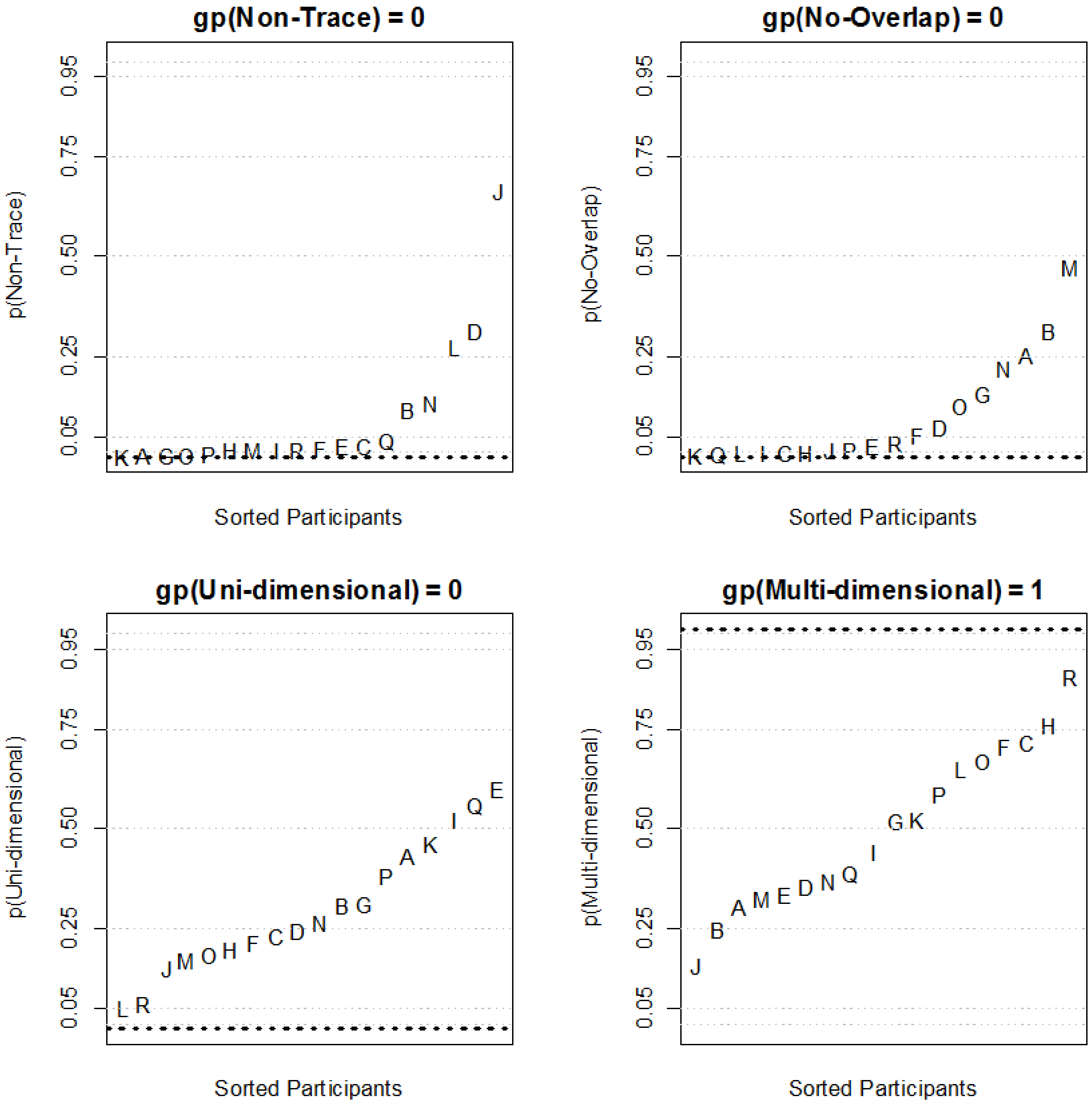

gp(Non-Trace) = 0
p(Non-Trace)
0.05
0.25
0.50
0.75
0.95
K A G O P H M I R F E C Q B N L D J
Sorted Participants
gp(No-Overlap) = 0
p(No-Overlap)
0.05
0.25
0.50
0.75
0.95
K Q L I C H J P E R F D O G N A B M
Sorted Participants
gp(Uni-dimensional) = 0
p(Uni-dimensional)
0.05
0.25
0.50
0.75
0.95
L R J M O H F C D N B G P A K I Q E
Sorted Participants
gp(Multi-dimensional) = 1
p(Multi-dimensional)
0.05
0.25
0.50
0.75
0.95
J B A M E D N Q I G K P L O F C H R
Sorted Participants

**Figure 2.**