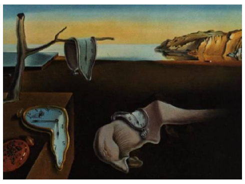

Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

vazul: An R Package for Analysis Blinding

Tamás Nagy
ELTE Eötvös Loránd University

Márton Kovács
Independent researcher

Alexandra Sarafoglou
University of Amsterdam

Abstract

Analysis blinding is a methodological approach designed to protect the confirmatory status of an analysis by concealing crucial test-relevant aspects of the data from the analysts. This article introduces the vazul package for R, which provides a comprehensive suite of tools for implementing masking and scrambling techniques. We discuss the theoretical foundations of analysis blinding, particularly as a complement or alternative to preregistration. The package allows researchers to maintain analytic flexibility while safeguarding against bias. By using vazul, data managers can easily generate blinded datasets that preserve essential distributional properties while obscuring associations or labels that could lead to biased decisions. We provide detailed examples of vector-level, data-frame-level, and row-wise blinding operations, and demonstrate how the package handles complex data structures such as hierarchical or repeated-measures designs.

Keywords: analysis blinding, R package, bias control.

1. Introduction

In data analysis, researchers face the challenge of maintaining objectivity and minimizing bias. A central concern is the considerable analytic flexibility present at various stages of the workflow, including data preprocessing, variable selection, model specification, and statistical testing. This flexibility has been described as the “garden of forking paths” (Gelman and Loken 2013), meaning that possibly hundreds or thousands of plausible, non-redundant analytic strategies can be applied to the same dataset (Steegen, Tuerlinckx, Gelman, and Vanpaemel 2016; Simonsohn, Nelson, and Simmons 2020). Crucially, different analytic paths can yield different results, and analysts may, intentionally or not, gravitate toward those that align with their expectations.

The result can be biased or even spurious findings (Ioannidis 2005b; Simmons, Nelson, and Simonsohn 2011; Gelman and Loken 2016), which have eroded trust in science and contributed to replication failures across medical sciences, psychology, economics, and computer science

(Ioannidis 2005a; Open Science Collaboration 2015; Baker 2016; Nosek, Hardwicke, Moshontz, Allard, Corker, Dreber, Fidler, Hilgard, Kline Struhl, Nuijten *et al.* 2022; Pashler and Wagenmakers 2012; Camerer, Dreber, Forsell, Ho, Huber, Johannesson, Kirchler, Almenberg, Altmejd, Chan *et al.* 2016; Cockburn, Dragicevic, Besançon, and Gutwin 2020).[1] To ensure the reliability of empirical findings, it is therefore essential to prevent the idiosyncrasies of the data from feeding back into the formulation of the hypotheses being tested (HARKing, Kerr 1998) or from prompting researchers to exploit their analytic freedom to accentuate desired outcomes (Simmons *et al.* 2011). In response, researchers have proposed methodologies to limit the influence of bias in data analysis. One such method is analysis blinding, which supplies analysts with an altered version of the real data that conceals biasing features until the analysis plan is finalized.

Analysis blinding, or blind analysis (MacCoun and Perlmutter 2015; MacCoun and Perlmutter 2018), is a methodological approach designed to protect the confirmatory status of an analysis by concealing crucial test-relevant aspects of the data from the analysts—for example, by scrambling dependent variables or masking condition labels. The procedure typically involves two independent parties: a data manager, who has full access to the raw data and is responsible for implementing the blinding steps, and an analyst, who is tasked with developing the analysis plan. After the data manager applies the blinding procedure, the analyst receives the resulting modified dataset: the blinded data.

Working with the blinded data, the analyst can explore and preprocess the dataset and develop an analysis plan without being influenced by whether a particular analytic choice produces the desired result, as any patterns in the blinded data are, by design, only a product of chance. Importantly, although the test-relevant elements are concealed, other meaningful characteristics remain intact, including demographic information, secondary variables, and the distributional properties of both dependent and independent variables. As a result, the analyst retains all information necessary to tailor the analytic approach to the specific, and potentially unexpected, features of the data. Once the analysis plan has been finalized using the blinded data, the analyst is granted access to the raw, unblinded dataset, and the confirmatory analysis can then be carried out strictly according to the pre-established plan.

Analysis blinding ties in with other methodologies to minimize bias in the research cycle, such as single-blind or double-blind experimental designs, where participants and/or experimenters are unaware of certain aspects of the study to prevent bias in data collection (Schulz and Grimes 2002). Here, the analysts are intentionally kept unaware of certain aspects of the data to prevent bias in data analysis, which is why the methodology is also referred to as triple-blind (Schulz and Grimes 2002).

Although many blinding techniques are conceptually straightforward, such as scrambling outcome variables or masking variable names and factor labels, their practical implementation can become technically challenging once real research designs and data structures are considered. For instance, in hierarchical models, researchers may need to scramble the outcome variable to destroy the effect of interest while still preserving information within grouping levels, such as country-level means in cross-cultural datasets (e.g., Sarafoglou, Hoogeveen, and Wagenmakers 2023a). Similarly, in repeated-measures designs, condition labels may need to

[1] But see Gilbert, King, Pettigrew, and Wilson (2016), Muradchanian, Hoekstra, Kiers, and van Ravenzwaaij (2021), Savitz, Wise, Bond, Hatch, Ncube, Wesselink, Willis, Yland, and Rothman (2024), and McShane, Bradlow, Lynch Jr, and Meyer (2024) for critiques of how replication failure is measured and of the design and validity of some replication attempts.

be masked so that each participant's condition structure is preserved while permuting the mapping of conditions across participants to obscure the true experimental effect. These complexities make manual analysis blinding difficult to carry out consistently and correctly. To lower barriers for data managers and facilitate the broader adoption of analysis blinding in research labs, dedicated software tools, such as R packages, are needed to implement these procedures reliably.

To meet this need, the **vazul**[2] package in R (https://CRAN.R-project.org/package=vazul) offers flexible and reproducible tools for implementing analysis blinding. The R package provides functionality for data scrambling as well as data and variable masking. In the remainder of this article, we introduce the methodology in more detail, describe the functionalities of the **vazul** package, and illustrate its application through practical examples. We conclude with a brief discussion and outline future directions.

1.1. Analysis Blinding: Safeguarding Against Bias Without Preregistration

Originally, analysis blinding gained traction in astrophysics in the early 2000s (MacCoun and Perlmutter 2018) as a means of safeguarding analysts against bias. Since then, the methodology has also been advocated in the social and behavioral sciences as an alternative or complement to preregistration (Dutilh, Sarafoglou, and Wagenmakers 2019; Nagy, Hergert, Elsherif, Wallrich, Schmidt, Waltzer, Payne, Gjoneska, Seetahul, Wang, Scharfenberg, Tyson, Yang, Skvortsova, Alarie, Graves, Sotola, Moreau, and Rubínová 2025; MacCoun 2021; MacCoun and Perlmutter 2015; MacCoun and Perlmutter 2018; Aczel, Szaszi, Sarafoglou, Kekecs, Kucharský, Benjamin, Chambers, Fisher, Gelman, Gernsbacher, Ioannidis, Johnson, Jonas, Kousta, Lilienfeld, Lindsay, Morey, Munafò, Newell, Pashler, Shanks, Simons, Wicherts, Albarracin, Anderson, Antonakis, Arkes, Back, Banks, Beevers, Bennett, Bleidorn, Boyer, Cacciari, Carter, Cesario, Clifton, Conroy, Cortese, Cosci, Cowan, Crawford, Crone, Curtin, Engle, Farrell, Fearon, Fichman, Frankenhuis, Freund, Gaskell, Giner-Sorolla, Green, Greene, Harlow, Hoces de la Guardia, Isaacowitz, Kolodner, Lieberman, Logan, Mendes, Moersdorf, Nyhan, Pollack, Sullivan, Vazire, and Wagenmakers 2020).

This interest partly reflects the fact that analysis blinding addresses practical limitations of preregistration when analytic flexibility is required. While preregistration provides a valuable framework for increasing transparency, analysts may nonetheless encounter unexpected features of the data that necessitate deviations from the preregistered plan. Such deviations are common and often unavoidable, and when transparently reported and justified, they can enhance the credibility and validity of research findings (e.g., Van Den Akker, Bakker, Van Assen, Pennington, Verweij, Elsherif, Claesen, Gaillard, Yeung, Frankenberger, Krautter, Cockcroft, Kreuer, Evans, Heppel, Schoch, Korbmacher, Yamada, Albayrak-Aydemir, Alzahawi, Sarafoglou, Sitnikov, Děchtěrenko, Wingen, Grinschgl, Hartmann, Stewart, De Oliveira, Ashcroft-Jones, Baker, and Wicherts 2024; Lakens 2024; Schneider, Rosman, Kelava, and Merk 2022; Song, Markowitz, and Taylor 2022).

At the same time, preregistration offers limited guidance on how analytic decisions should be adapted in response to unforeseen data characteristics, and how such deviations should be

[2] Vazul was an 11th-century Hungarian prince of the Árpád dynasty, a cousin of King Stephen I (c. 975–1038). Around 1031, Vazul was blinded—according to contemporary chronicles-to make him unfit for succession to the throne. In the same spirit, the **vazul** package helps create blinded data that are unfit for p-hacking.

interpreted with respect to confirmatory status. As a result, researchers may face a tension between adhering strictly to a preregistered plan and making principled, data-informed adjustments. Analysis blinding alleviates this tension by allowing analysts to flexibly develop and refine analysis plans using blinded data, before any test-relevant information is revealed.

In contrast to preregistration, analysis blinding allows researchers to maintain flexibility in their analysis plans, as they can explore the data without being constrained by a rigid preregistered protocol. This flexibility is especially important in analyses involving advanced statistical modeling or preprocessing, where not all decisions can be anticipated in advance and researchers must make data-dependent choices. For instance, many psychological constructs, such as well-being, are measured with multi-item scales whose factor structure must be examined before further analysis; whether the items load onto a single factor or multiple factors determines subsequent analytic steps, including whether composite scores can be computed or whether a more complex measurement model is required (see e.g., Hoogeveen, Sarafoglou, van Elk, and Wagenmakers 2023). The flexibility gained by analysis blinding enables researchers to develop analysis strategies that are appropriately tailored to the specific characteristics of the dataset without the need to anticipate and specify all eventualities in advance. Consequently, researchers who analyze their data with analysis blinding are less likely to deviate from their developed analysis plan compared to researchers who preregister their analytic strategy in a text-based format (Sarafoglou *et al.* 2023a).

Moreover, similar in spirit to approaches proposed in code-based preregistration (Peikert, Van Lissa, and Brandmaier 2021; Van Lissa, Brandmaier, Brinkman, Lamprecht, Peikert, Struiksma, and Vreede 2021), analysis blinding naturally produces analysis plans that are specific, precise, and exhaustive, because they are written directly in code rather than described verbally in preregistration templates. As with preregistration, analysis scripts based on the blinded data can be uploaded or attached to preregistration platforms (e.g., the Open Science Framework) to provide a transparent record of the planned analysis before the data are unblinded. In short, analysis blinding can serve as a valuable alternative or additional safeguard to preregistered analysis plans: it effectively protects against bias while providing analysts with the flexibility needed to develop an analysis strategy that is optimally suited to their data. Notably, even when the blinding code itself is shared, the original data cannot be recovered from the blinded dataset without access to the raw data, making analysis blinding compatible with transparent and open workflows.

Analysis blinding can be particularly valuable for secondary data analysis, where the confirmatory power of preregistration is often compromised by prior exposure to the data (Thibault, Kovacs, Hardwicke, Sarafoglou, Ioannidis, and Munafò 2023). In archival datasets, large-scale surveys, or shared repositories, researchers may already possess unavoidable knowledge of variable distributions or previously reported associations. By developing and finalizing analysis plans on a blinded version of the existing data, researchers can maintain analytic flexibility while providing a verifiable safeguard against post-hoc bias. This approach effectively protects the confirmatory status of the study in contexts where traditional preregistration might otherwise be perceived as impractical or insufficient.

2. Overview of the main functions

The package provides functions for two main types of analysis blinding:

1. **Masking**: Replaces original values with anonymous labels, completely hiding the original information.
2. **Scrambling**: Randomizes the order of existing values while preserving all original data content.

2.1. Choosing between masking and scrambling

When choosing between masking and scrambling for analysis blinding, the decision usually turns on what information must be concealed and what structure must be preserved to allow for the development of a suitable analysis pipeline. Masking is most appropriate when the risk lies in the labels of categories rather than their relationships. By replacing group names (e.g., 'Treatment' and 'Control') with arbitrary codes (e.g., 'Group A' and 'Group B'), the group allocation is preserved. This allows analysts to perform group-level operations—such as ANOVA or regression—while remaining blinded to the substantive identity of those groups. Note, that since group-level distinctions remain intact, masking is best suited for workflows where knowing the direction of a difference would not inadvertently reveal the underlying category meaning. Masking also accommodates cases where different variables require different relabeling schemes or prefixes, for example when several independent categorical factors need to remain distinguishable but substantively opaque. This approach is common in clinical trials, behavioral experiments, and preregistered confirmatory analyses where preserving the structure of factors is essential but knowledge of factor identity would bias analytic choices. For instance, an analyst can check for data consistency across four masked study sites (e.g., "Site A" through "Site D") to ensure uniform data quality without knowing the specific geographic locations. This preserves the site-level grouping for variance testing while preventing biased assumptions about data from specific regions.

Masking can be indispensable in situations where the variable names themselves carry substantive meaning. This occurs, for example, in network analysis, where variables become nodes with interpretable labels, or in factor-analytic work, where item names often hint at their latent content (Hoekstra, Huth, Sekulovski, Delhalle, and Sarafoglou 2025). In such settings, simply scrambling values would not prevent analysts from inferring the underlying constructs, whereas masking removes this semantic cue entirely. The `vazul` package supports this use case directly by offering functionality for masking variable names alongside their values.

Scrambling becomes the better choice when the goal is to preserve the marginal distributions of numeric variables while severing their hypothesized associations. It is particularly useful in settings where analysts need access to realistic distributions—variance, skew—to make decisions about transformations, model families, or robustness checks, yet must remain blind to the relationship between predictors and outcomes. By permuting values within variables, scrambling prevents recognition of treatment or experimental effects, correlations, or temporal patterns, making it harder to inadvertently tune an analysis toward the desired result. It is often the preferred option in longitudinal studies, high-dimensional observational datasets where categorical relabeling would either destroy important numeric structure or be easy to reverse-engineer (e.g., in case of three experimental conditions, the researcher may guess which one is which based on the means). Scrambling is generally the more versatile option, because it can be applied across variable types and adapts well to a wide range of study designs and analytic workflows. Its ability to break associations while preserving distributions makes it suitable for almost everything from tightly controlled experiments to complex observational datasets, including high-dimensional or mixed-format data (Sarafoglou and Hoogeveen 2025; Sarafoglou, Hoogeveen, and Wagenmakers 2023b).

Table 1: Comparison of masking and scrambling approaches for analysis blinding.

Aspect	**Masking**	**Scrambling**
Original values	Replaced with arbitrary codes	Preserved but permuted
Data distribution	Category counts preserved with different labels	Marginal distributions fully preserved
Associations with blinded variables	Preserved	Broken (correlations, treatment–outcome links removed)
Risk addressed	Hiding the identity or meaning of categories or variable names	Preventing recognition of true relationships or effects
Best suited for	Categorical variables with many different labels; variables whose names convey substantive meaning	Numeric or categorical variables; mixed-type data
Typical use cases	Clinical trials, behavioral experiments, network analysis, factor analysis	Longitudinal data, observational datasets, robustness checks
When essential	When variable names or category labels carry substantive information	When analysts need realistic numeric structure but must remain blind to associations
Flexibility across research settings	Limited by categorical/semantic constraints	Broadly applicable to most data types and study designs

In the `vazul` package, each approach is available at three levels:

- **Vector level**: `mask_labels()` and `scramble_values()` - operate on single vectors
- **Data frame level**: `mask_variables()` and `scramble_variables()` - operate on columns in a data frame
- **Row-wise level**: `mask_variables_rowwise()` and `scramble_variables_rowwise()` - operate within rows across columns

The design of these three levels reflects the common ways researchers interact with data in R. Vector-level functions are the primary building blocks, designed for simple pipelines where a researcher might want to blind a single outcome or a specific grouping factor. These functions are particularly useful when working with **dplyr**'s `mutate()` or within custom functions where a specific variable needs to be isolated and transformed without affecting the rest of the environment (Wickham, François, Henry, Müller, and Vaughan 2025). By providing these atomic operations, **vazul** ensures that the core blinding logic remains accessible even for non-standard data structures.

Moving to the data frame and row-wise levels, the package addresses more complex research designs. Data frame-level functions allow for bulk operations, which are essential for datasets with dozens of variables requiring consistent masking or independent scrambling. The row-wise operations solve a frequent headache in repeated-measures and longitudinal research. In these designs, the "unit" of analysis is often spread across multiple columns (e.g., independent varibles in factorial designs); the row-wise functions ensure that the relationship within a participant's data is handled correctly—either by scrambling values across those specific time points for each person or by applying a consistent mask that hides the identity of the measurement while preserving the participant-level structure.

All of the functions require a data frame or vector as input and return a modified version with masked or scrambled values. The original data structure, including class attributes and metadata, is carefully preserved to ensure that the blinded data can be passed directly into existing analysis scripts or visualization packages without requiring additional type conversions.

Table 2: Overview of vazul functions for masking and scrambling data

Function	Level	Purpose
`mask_labels()`	Vector	Replace categorical values with anonymous labels
`mask_variables()`	Data frame	Mask multiple columns
`mask_variables_rowwise()`	Row-wise	Masking within rows
`mask_names()`	Variable names	Mask column names
`scramble_values()`	Vector	Randomize value order
`scramble_variables()`	Data frame	Scramble multiple columns
`scramble_variables_rowwise()`	Row-wise	Scramble values within rows

In the following sections, we provide detailed descriptions and examples for each of these functions, illustrating their use cases and practical applications in research workflows. We starts with setting up the environment and loading the included datasets.

```
R> library(vazul)
R> library(dplyr)
R>
R> set.seed(1037)
R>
R> data(marp)
R> data(williams)
```

2.2. Included datasets

The `vazul` package includes two research datasets for demonstration and practice. The `marp` dataset contains cross-national survey data on religiosity (Hoogeveen *et al.* 2023), while the `williams` dataset contains experimental data from a stereotyping study (Sarafoglou and Hoogeveen 2025).

2.3. Masking functions

Masking functions replace categorical values with anonymous labels. This is useful when you want to hide the original information, such as treatment conditions or group assignments, and there are a limited number of unique values. The `mask_labels()` function takes a character or factor type vector and replaces each unique value with a randomly assigned masked label. The function preserves factor structure when the input is a factor. After masking, each unique value receives a unique masked label, the same original value always maps to the same masked label, and the assignment of masked labels to original values is randomized. Missing values are preserved during masking.

```
R> # Create a simple treatment vector
R> treatment <- c("control", "treatment_1", "treatment_2", "control", "treatment_1", "trea
R>
R> # Mask the labels
R> mask_labels(treatment)

[1] "masked_group_01" "masked_group_02" "masked_group_03" "masked_group_01"
[5] "masked_group_02" "masked_group_03"
```

It is possible to customize the prefix used for masked labels:

```
R> mask_labels(treatment, prefix = "group_")

[1] "group_01" "group_02" "group_03" "group_01" "group_02" "group_03"
```

The `mask_variables()` function extends the masking functionality to multiple columns in a data frame simultaneously. It is possible to use tidyelect helpers to select columns. for instance, in the MARP dataset, we want to simultaneously mask labels within the columns "country" and "denomination". To make the example manageable, we first create a smaller example dataset with 10 random rows. Then we demonstrate how to use `mask_variables()` to mask the selected columns. It is possible to use tidyselect helpers (such as `starts_with()`, `any_of()`, etc.) to select columns. In the next example, we apply masking to all character columns in the example dataset.

```
R> marp_example <-
+     marp |>
+     select(subject, country, denomination) |>
+     sample_n(6)
R>
R> marp_example

  subject country              denomination
1    5011   India Christian (Roman Catholic)
2    4015 Germany Christian (Roman Catholic)
3    4271 Germany               Evangelical
4    4591 Germany                      <NA>
5   10225      US Christian (Roman Catholic)
6    9060   Spain                      <NA>
```

```
R> marp_example |>
+     mask_variables(c("country", "denomination"))

  subject          country         denomination
1    5011 country_group_01 denomination_group_01
2    4015 country_group_03 denomination_group_01
3    4271 country_group_03 denomination_group_02
4    4591 country_group_03                 <NA>
5   10225 country_group_04 denomination_group_01
6    9060 country_group_02                 <NA>

R> marp_example |>
+     mask_variables(where(is.character))

  subject          country         denomination
1    5011 country_group_04 denomination_group_01
2    4015 country_group_02 denomination_group_01
3    4271 country_group_02 denomination_group_02
4    4591 country_group_02                 <NA>
5   10225 country_group_03 denomination_group_01
6    9060 country_group_01                 <NA>
```

By default, each column gets its own set of masked labels with the column name as prefix. When `across_variables = TRUE`, all selected columns share the same mapping. This can be useful when the same conditions appear in multiple columns.

```
R> df <- data.frame(pre_condition = c("A", "B", "C", "D"),
+                   post_condition = c("B", "D", "D", "C"),
+                   id = c(1, 2, 3, 4)
+                   )
R> df

  pre_condition post_condition id
1             A              B  1
2             B              D  2
3             C              D  3
4             D              C  4

R> mask_variables(df,
+                 c("pre_condition", "post_condition"),
+                 across_variables = TRUE)

    pre_condition  post_condition id
1 masked_group_02 masked_group_01  1
2 masked_group_01 masked_group_04  2
3 masked_group_03 masked_group_04  3
4 masked_group_04 masked_group_03  4
```

The `mask_variables_rowwise()` function applies consistent masking within each row across multiple columns. This is useful for example in factorial designs or repeated measures designs, where multiple independent variables require concealment. Within each row, the original values are consistently mapped to masked labels, but the mapping is independent across rows. The function can use tidyselect helpers. In the next example, we show how to mask treatment conditions across multiple waves of a longitudinal study.

```
R> df <- data.frame(id = 1:3,
+                   wave_1 = c("control", "treatment_1", "treatment_2"),
+                   wave_2 = c("treatment_1", "treatment_2", "control"),
+                   wave_3 = c("treatment_2", "control", "treatment_1"),
+                   wave_4 = c("treatment_2", "control", "treatment_1")
+                   )
R> df

  id      wave_1      wave_2      wave_3      wave_4
1  1     control treatment_1 treatment_2 treatment_2
2  2 treatment_1 treatment_2     control     control
3  3 treatment_2     control treatment_1 treatment_1

R> mask_variables_rowwise(df, starts_with("wave_"))

  id          wave_1          wave_2          wave_3          wave_4
1  1 masked_group_02 masked_group_01 masked_group_03 masked_group_03
2  2 masked_group_01 masked_group_03 masked_group_02 masked_group_02
3  3 masked_group_03 masked_group_02 masked_group_01 masked_group_01
```

Masking variable names

The `mask_names()` function allows users to rename variables in a dataset to anonymous, generic labels. This is especially useful in analyses where variable names carry substantive meaning (e.g., questionnaire items, network nodes, test-items), and one wants to prevent analysts from recognizing which item is which during preprocessing or exploratory phases.

The `mask_names()` function takes a data frame and one or more variable sets, which can be specified either with tidyselect expressions (e.g., `starts_with("Q")`) or as character vectors of column names. The `prefix` argument sets the base for the masked names. For instance, in the `williams` data we can blind all variables that are related to the impulsivity scale.

```
R> names(williams)

 [1] "subject"        "SexUnres_1"       "SexUnres_2"
 [4] "SexUnres_3"     "SexUnres_4_r"     "SexUnres_5_r"
 [7] "Impuls_1"       "Impuls_2_r"       "Impul_3_r"
[10] "Opport_1"       "Opport_2"         "Opport_3"
[13] "Opport_4"       "Opport_5"         "Opport_6_r"
[16] "InvEdu_1_r"     "InvEdu_2_r"       "InvChild_1"
```

```
[19] "InvChild_2_r"        "age"                 "gender"
[22] "ecology"             "duration_in_seconds" "attention_1"
[25] "attention_2"
```

```
R> williams |>
+     mask_names(starts_with("Impul"), prefix = "masked_") |>
+     names()
```

```
 [1] "subject"             "SexUnres_1"          "SexUnres_2"
 [4] "SexUnres_3"          "SexUnres_4_r"        "SexUnres_5_r"
 [7] "masked_03"           "masked_02"           "masked_01"
[10] "Opport_1"            "Opport_2"            "Opport_3"
[13] "Opport_4"            "Opport_5"            "Opport_6_r"
[16] "InvEdu_1_r"          "InvEdu_2_r"          "InvChild_1"
[19] "InvChild_2_r"        "age"                 "gender"
[22] "ecology"             "duration_in_seconds" "attention_1"
[25] "attention_2"
```

Use multiple `mask_names()` calls to blind different sets of variables independently. Assigning distinct prefixes allows variables to be anonymized while preserving information about their original clusters. For example, an analyst can recognize that a set of variables belongs to the same construct—enabling procedures such as factor analysis or reliability estimation—without knowing which specific items are being evaluated. The example below demonstrates this clustered masking approach in a pipe-based workflow.

```
R> masked_williams <-
+     williams |>
+     mask_names(starts_with("SexUnres"), prefix = "masked_set_A_") |>
+     mask_names(starts_with("Impul"), prefix = "masked_set_B_") |>
+     mask_names(starts_with("Opport"), prefix = "masked_set_C_") |>
+     mask_names(starts_with("InvEdu"), prefix = "masked_set_D_") |>
+     mask_names(starts_with("InvChild"), prefix = "masked_set_E_")
R>
R> names(masked_williams)
```

```
 [1] "subject"             "masked_set_A_05"     "masked_set_A_01"
 [4] "masked_set_A_02"     "masked_set_A_04"     "masked_set_A_03"
 [7] "masked_set_B_02"     "masked_set_B_03"     "masked_set_B_01"
[10] "masked_set_C_01"     "masked_set_C_02"     "masked_set_C_03"
[13] "masked_set_C_04"     "masked_set_C_06"     "masked_set_C_05"
[16] "masked_set_D_02"     "masked_set_D_01"     "masked_set_E_01"
[19] "masked_set_E_02"     "age"                 "gender"
[22] "ecology"             "duration_in_seconds" "attention_1"
[25] "attention_2"
```

Once masked, one can proceed with analyses (e.g. factor analysis, network analysis) on the masked variables, without knowing which original items correspond to which columns. For example, we can run a factor analysis on the masked items:

```
R> masked_williams |>
+   select(starts_with("masked_set")) |>
+   factanal(factors = 3, rotation = "varimax") |>
+   loadings() |>
+   print(cutoff = 0.3, sort = TRUE)
```

```
Loadings:
                Factor1 Factor2 Factor3
masked_set_A_05  0.648           0.504
masked_set_A_01  0.705
masked_set_A_02  0.730           0.340
masked_set_B_02  0.751
masked_set_C_01  0.873
masked_set_C_02  0.833
masked_set_C_03  0.866
masked_set_C_04  0.761
masked_set_C_06  0.863
masked_set_E_01  0.770
masked_set_A_04          0.745   0.306
masked_set_A_03          0.568
masked_set_B_03          0.861
masked_set_B_01          0.740
masked_set_C_05          0.696
masked_set_D_02          0.695
masked_set_D_01          0.841
masked_set_E_02          0.865

               Factor1 Factor2 Factor3
SS loadings      6.314   4.842   0.731
Proportion Var   0.351   0.269   0.041
Cumulative Var   0.351   0.620   0.660
```

Because the column names are anonymized, any decisions about factor inclusion or rotation are made blind to the substantive meaning of items, reducing the risk of interpretive bias.

In short, `mask_names()` supports naming-based blinding by decoupling analysis from semantic content, while preserving the full data structure needed for legitimate exploratory workflows.

2.4. Scrambling functions

Scrambling functions randomize the order of values while preserving all original data content. This approach maintains the data distribution while breaking the connection between observations and their original values. The `scramble_values()` function randomly reorders the elements of a vector. The vector can contain numeric, character, or factor type data. Scrambling is useful when we want to preserve the original values but eliminate any correspondence between observations and their values.

```
R> # Numeric data
R> numbers <- 1:10
R> scramble_values(numbers)
```

```
 [1]  7  1 10  2  3  5  4  8  9  6
```

The `scramble_variables()` function extends the scrambling functionality to data frames. It allows scrambling multiple columns simultaneously, with options for independent scrambling, joint scrambling, and within-group scrambling. Columns can be selected using tidyselect helpers. For instance, in the williams data we could choose to scramble the columns age (a demographic variable) and ecology (the identifier of the experimental condition) as follows:

```
R> williams_example <-
+     williams |>
+     select(subject, age, ecology) |>
+     head()
R>
R> williams_example
```

```
# A tibble: 6 x 3
  subject          age ecology
  <chr>          <dbl> <chr>
1 A3OMP4LXV4MIFD    34 Hopeful
2 A16X5FB3HAFCKN    30 Desperate
3 A1E9D1OT9VJYDZ    40 Desperate
4 A16FPOYD7566WI    35 Hopeful
5 A11NOTVHWST7Y3    26 Desperate
6 A3TDR6MXS6UO5Z    33 Desperate
```

```
R> scramble_variables(williams_example, c("age", "ecology"))
```

```
# A tibble: 6 x 3
  subject          age ecology
  <chr>          <dbl> <chr>
1 A3OMP4LXV4MIFD    35 Hopeful
2 A16X5FB3HAFCKN    33 Desperate
3 A1E9D1OT9VJYDZ    40 Hopeful
4 A16FPOYD7566WI    34 Desperate
5 A11NOTVHWST7Y3    30 Desperate
6 A3TDR6MXS6UO5Z    26 Desperate
```

By default, columns are scrambled independently. Setting `together = TRUE` causes the selected columns to be scrambled jointly, preserving their row-wise associations. Adding this argument to the previous example therefore maintains the relationship between the two scrambled variables.

```
R> scramble_variables(williams_example, c("age", "ecology"), together = TRUE)
```

```
# A tibble: 6 x 3
  subject          age ecology
  <chr>          <dbl> <chr>
1 A30MP4LXV4MIFD    30 Desperate
2 A16X5FB3HAFCKN    35 Hopeful
3 A1E9D10T9VJYDZ    34 Hopeful
4 A16FP0YD7566WI    33 Desperate
5 A11N0TVHWST7Y3    40 Desperate
6 A3TDR6MXS6U05Z    26 Desperate
```

Scrambling can be done in groups using the `.groups` parameter. This ensures that values are only scrambled within their original groups. In the following example, we scramble the columns x and y within each group defined by the group column:

```
R> df <- data.frame(x = 1:6,
+                   y = letters[1:6],
+                   group = c("A", "A", "A", "B", "B", "B")
+                   )
R> df
```

```
  x y group
1 1 a     A
2 2 b     A
3 3 c     A
4 4 d     B
5 5 e     B
6 6 f     B
```

```
R> scramble_variables(df, c("x", "y"), .groups = "group")
```

```
# A tibble: 6 x 3
      x y     group
  <int> <chr> <chr>
1     2 c     A
2     3 b     A
3     1 a     A
4     5 e     B
5     6 f     B
6     4 d     B
```

The `scramble_variables_rowwise()` function scrambles values within each row across specified columns. This is useful for scrambling repeated measures or item responses. Within each row, the values are shuffled among the item columns. You can scramble multiple sets of columns independently. The function can use tidyselect helpers. In the next example, we scramble values across three day variables for each participant in a hypothetical dataset:

```
R> df_wide <- data.frame(
+   day_1 = c(1, 4, 7),
+   day_2 = c(2, 5, 8),
+   day_3 = c(3, 6, 9),
+   score_a = c(10, 40, 70),
+   score_b = c(20, 50, 80),
+   id = 1:3
+ )
R>
R> df_wide

  day_1 day_2 day_3 score_a score_b id
1     1     2     3      10      20  1
2     4     5     6      40      50  2
3     7     8     9      70      80  3

R> scramble_variables_rowwise(df_wide, starts_with("day_"), c("score_a", "score_b"))

  day_1 day_2 day_3 score_a score_b id
1     1    10    20       3       2  1
2    40    50     6       4       5  2
3     7     8    80      70       9  3
```

3. Summary

The introduction of **vazul** fills a notable gap in the R ecosystem, as no dedicated package currently exists to specifically address the diverse requirements of analysis blinding. While some basic blinding tasks—such as simple vector permutations—can be achieved using base R or **tidyverse** functions like sample() or mutate(), these manual approaches quickly become error-prone as study designs grow in complexity. When dealing with multiple independent variables, hierarchical data structures, or the need for consistent masking across several columns, the risk of “breaking” the data structure or accidentally unblinding the analyst increases significantly. By providing a unified and tested framework, **vazul** abstracts these technical intricacies, and allow researchers to focus on the conceptual rigor of their analysis plans rather than the underlying data manipulation logic.

The functions implemented in **vazul** are applicable to a wide range of paradigms and data structures and simplify the blinding process, which in turn facilitates the assignment of data-blinding roles within research teams. Specifically, since data blinding no longer requires implementing custom code the data-manager role need not be filled by an expert in the analytic method or data domain. Instead, junior lab members or external collaborators can manage data blinding, while domain experts develop the analytic strategy.

Beyond standalone use in R scripts, the package’s modular design makes it an ideal candidate for integration into other statistical software platforms. For instance, the inclusion of **vazul** as a back-end for the JASP (JASP Team 2025) graphical interface could provide a point-and-click solution for analysis blinding, further facilitating the adoption of these best practices across the social and behavioral sciences.

4. Acknowledgements

Tamás Nagy was supported by the University Excellence Fund of Eötvös Loránd University, Budapest, Hungary (ELTE), and the János Bolyai research fellowship of the Hungarian Academy of Sciences. Alexandra Sarafoglou was supported by an 2024 Ammodo Science Award.

References

Aczel B, Szaszi B, Sarafoglou A, Kekecs Z, Kucharský v, Benjamin D, Chambers CD, Fisher A, Gelman A, Gernsbacher MA, Ioannidis JP, Johnson E, Jonas K, Kousta S, Lilienfeld SO, Lindsay DS, Morey CC, Munafò M, Newell BR, Pashler H, Shanks DR, Simons DJ, Wicherts JM, Albarracin D, Anderson ND, Antonakis J, Arkes H, Back MD, Banks GC, Beevers C, Bennett AA, Bleidorn W, Boyer TW, Cacciari C, Carter AS, Cesario J, Clifton C, Conroy RM, Cortese M, Cosci F, Cowan N, Crawford J, Crone EA, Curtin J, Engle R, Farrell S, Fearon P, Fichman M, Frankenhuis W, Freund AM, Gaskell MG, Giner-Sorolla R, Green DP, Greene RL, Harlow LL, Hoces de la Guardia F, Isaacowitz D, Kolodner J, Lieberman D, Logan GD, Mendes WB, Moersdorf L, Nyhan B, Pollack J, Sullivan C, Vazire S, Wagenmakers EJ (2020). "A Consensus-Based Transparency Checklist." *Nature Human Behaviour*, **4**, 4–6. doi:10.1038/s41562-019-0772-6.

Baker M (2016). "1,500 scientists lift the lid on reproducibility." *Nature News*, **533**, 452. doi:10.1038/533452a.

Camerer CF, Dreber A, Forsell E, Ho TH, Huber J, Johannesson M, Kirchler M, Almenberg J, Altmejd A, Chan T, *et al.* (2016). "Evaluating replicability of laboratory experiments in economics." *Science*, **351**(6280), 1433–1436. doi:10.1126/science.aaf0918.

Cockburn A, Dragicevic P, Besançon L, Gutwin C (2020). "Threats of a replication crisis in empirical computer science." *Communications of the ACM*, **63**(8), 70–79. doi:10.1145/3360311.

Dutilh G, Sarafoglou A, Wagenmakers EJ (2019). "Flexible Yet Fair: Blinding Analyses in Experimental Psychology." *Synthese*, **198**, S5745–S5772. doi:10.1007/s11229-019-02456-7.

Gelman A, Loken E (2013). "The garden of forking paths: Why multiple comparisons can be a problem, even when there is no "fishing expedition" or "p-hacking" and the research hypothesis was posited ahead of time." *Department of Statistics, Columbia University*, **348**. URL http://www.stat.columbia.edu/~gelman/research/unpublished/p_hacking.pdf.

Gelman A, Loken E (2016). "The statistical crisis in science." In M Pitici (ed.), *The best writing on mathematics*, volume 102, pp. 305–318. Princeton University Press.

Gilbert DT, King G, Pettigrew S, Wilson TD (2016). "Comment on "Estimating the reproducibility of psychological science"." *Science*, **351**(6277), 1037–1037. doi:10.1126/science.aad7243.

Hoekstra RHA, Huth K, Sekulovski N, Delhalle M, Sarafoglou A (2025). "Safeguarding against bias without preregistration: A tutorial on analysis blinding for network analysis." *PsyArXiv*. doi:10.31234/osf.io/gruw5_v1. URL https://osf.io/preprints/psyarxiv/gruw5_v1.

Hoogeveen S, Sarafoglou A, van Elk M, Wagenmakers EJ (2023). "A Many-Analysts Approach to the Relation between Religiosity and Well-Being: Reflection and Conclusion." *Religion, Brain & Behavior*, **13**(3), 356–363. doi:10.1080/2153599X.2022.2070263.

Ioannidis JP (2005a). "Contradicted and initially stronger effects in highly cited clinical research." *JAMA*, **294**(2), 218–228. doi:10.1001/jama.294.2.218.

Ioannidis JP (2005b). "Why most published research findings are false." *PLoS medicine*, **2**(8), e124. doi:10.1371/journal.pmed.1004085.

JASP Team (2025). "JASP (Version 0.95.4)[Computer software]." URL https://jasp-stats.org/.

Kerr N (1998). "HARKing: Hypothesizing After the Results are Known." *Personality and Social Psychology Review*, **2**, 196–217. doi:10.1207/s15327957pspr0203_4.

Lakens D (2024). "When and how to deviate from a preregistration." *Collabra. Psychology*, **10**(1), 117094. ISSN 2474-7394. doi:10.1525/collabra.117094. URL https://online.ucpress.edu/collabra/article-pdf/10/1/117094/839302/collabra_2024_10_1_117094.pdf.

MacCoun R (2021). "Blinding to remove biases in science and society." In R Hertwig, C Engel (eds.), *Deliberate ignorance: Choosing not to know*, pp. 51–64. MIT Press, Cambridge.

MacCoun R, Perlmutter S (2015). "Blind analysis: Hide results to seek the truth." *Nature*, **526**, 187–190. doi:10.1038/526187a.

MacCoun R, Perlmutter S (2018). "Blind Analysis as a Correction for Confirmatory Bias in Physics and in Psychology." In SO Lilienfeld, I Waldman (eds.), *Psychological Science Under Scrutiny: Recent Challenges and Proposed Solutions*, pp. 297–322. John Wiley and Sons. doi:10.1002/9781119095910.ch15.

McShane BB, Bradlow ET, Lynch Jr JG, Meyer RJ (2024). ""Statistical Significance" and Statistical Reporting: Moving Beyond Binary." *Journal of Marketing*, p. 00222429231216910. doi:10.1177/00222429231216910.

Muradchanian J, Hoekstra R, Kiers H, van Ravenzwaaij D (2021). "How best to quantify replication success? A simulation study on the comparison of replication success metrics." *Royal Society Open Science*, **8**, 201697. doi:10.1098/rsos.201697.

Nagy T, Hergert J, Elsherif MM, Wallrich L, Schmidt K, Waltzer T, Payne JW, Gjoneska B, Seetahul Y, Wang YA, Scharfenberg D, Tyson G, Yang YF, Skvortsova A, Alarie S, Graves K, Sotola LK, Moreau D, Rubínová E (2025). "Bestiary of Questionable Research Practices in Psychology." *Advances in Methods and Practices in Psychological Science*, **8**(3), 25152459251348431. ISSN 2515-2459, 2515-2467. doi:10.1177/25152459251348431.

Nosek BA, Hardwicke TE, Moshontz H, Allard A, Corker KS, Dreber A, Fidler F, Hilgard J, Kline Struhl M, Nuijten MB, *et al.* (2022). "Replicability, robustness, and reproducibility in psychological science." *Annual review of psychology*, **73**, 719–748. doi:10.1146/annurev-psych-020821-114157.

Open Science Collaboration (2015). "Estimating the Reproducibility of Psychological Science." *Science*, **349**, aac4716. doi:10.1126/science.aac4716.

Pashler H, Wagenmakers EJ (2012). "Editors' introduction to the special section on replicability in psychological science: A crisis of confidence?" *Perspectives on Psychological Science*, **7**, 528–530. doi:10.1177/1745691612465253.

Peikert A, Van Lissa CJ, Brandmaier AM (2021). "Reproducible research in R: A tutorial on how to do the same thing more than once." *Psych*, **3**(4), 836–867. doi:10.3390/psych3040053.

Sarafoglou A, Hoogeveen S (2025). "Analysis blinding as a potential means to foster a productive collaboration between original authors and replicators." *Collabra. Psychology*, **11**(1), 136869. ISSN 2474-7394. doi:10.1525/collabra.136869. URL https://dx.doi.org/10.1525/collabra.136869.

Sarafoglou A, Hoogeveen S, Wagenmakers EJ (2023a). "Comparing analysis blinding with preregistration in the Many-Analysts Religion Project." *Advances in Methods and Practices in Psychological Science*, **6**(1), 25152459221128319. doi:10.1177/25152459221128319.

Sarafoglou A, Hoogeveen S, Wagenmakers EJ (2023b). "Comparing Analysis Blinding With Preregistration in the Many-Analysts Religion Project." *Advances in Methods and Practices in Psychological Science*, **6**(1), 25152459221128319. ISSN 2515-2459. doi:10.1177/25152459221128319. URL https://doi.org/10.1177/25152459221128319.

Savitz DA, Wise LA, Bond JC, Hatch EE, Ncube CN, Wesselink AK, Willis MD, Yland JJ, Rothman KJ (2024). "Responding to Reviewers and Editors About Statistical Significance Testing." *Annals of Internal Medicine*, **177**, 385–386. doi:10.7326/M23-2430.

Schneider J, Rosman T, Kelava A, Merk S (2022). "Do open-science badges increase trust in scientists among undergraduates, scientists, and the public?" *Psychological Science*, **33**(9), 1588–1604. ISSN 0956-7976,1467-9280. doi:10.1177/09567976221097499. URL http://dx.doi.org/10.1177/09567976221097499.

Schulz KF, Grimes DA (2002). "Blinding in randomised trials: Hiding who got what." *The Lancet*, **359**(9307), 696–700.

Simmons J, Nelson L, Simonsohn U (2011). "False–positive Psychology: Undisclosed Flexibility in Data Collection and Analysis Allows Presenting Anything as Significant." *Psychological Science*, **22**, 1359–1366. doi:10.1177/0956797611417632.

Simonsohn U, Nelson LD, Simmons JP (2020). "Specification curve analysis." *Nature Human Behaviour*, **4**, 1208–1214. doi:10.1038/s41562-020-0912-z.

Song H, Markowitz DM, Taylor SH (2022). "Trusting on the shoulders of open giants? Open science increases trust in science for the public and academics." *The Journal of*

Communication, **72**(4), 497–510. ISSN 0021-9916,1460-2466. doi:10.1093/joc/jqac017. URL https://dx.doi.org/10.1093/joc/jqac017.

Steegen S, Tuerlinckx F, Gelman A, Vanpaemel W (2016). "Increasing transparency through a multiverse analysis." *Perspectives on Psychological Science*, **11**, 702–712. doi:10.1177/1745691616658637.

Thibault RT, Kovacs M, Hardwicke TE, Sarafoglou A, Ioannidis JPA, Munafò MR (2023). "Reducing bias in secondary data analysis via an Explore and Confirm Analysis Workflow (ECAW): a proposal and survey of observational researchers." *Royal Society Open Science*, **10**(10), 230568. ISSN 2054-5703. doi:10.1098/rsos.230568. URL https://dx.doi.org/10.1098/rsos.230568.

Van Den Akker OR, Bakker M, Van Assen MALM, Pennington CR, Verweij L, Elsherif MM, Claesen A, Gaillard SDM, Yeung SK, Frankenberger JL, Krautter K, Cockcroft JP, Kreuer KS, Evans TR, Heppel FM, Schoch SF, Korbmacher M, Yamada Y, Albayrak-Aydemir N, Alzahawi S, Sarafoglou A, Sitnikov MM, Děchtěrenko F, Wingen S, Grinschgl S, Hartmann H, Stewart SLK, De Oliveira CMF, Ashcroft-Jones S, Baker BJ, Wicherts JM (2024). "The potential of preregistration in psychology: Assessing preregistration producibility and preregistration-study consistency." *Psychological Methods*. ISSN 1939-1463, 1082-989X. doi:10.1037/met0000687.

Van Lissa CJ, Brandmaier AM, Brinkman L, Lamprecht AL, Peikert A, Struiksma ME, Vreede BM (2021). "WORCS: A workflow for open reproducible code in science." *Data Science*, **4**(1), 29–49. doi:10.3233/DS-210031.

Wickham H, François R, Henry L, Müller K, Vaughan D (2025). "dplyr: A Grammar of Data Manipulation." URL https://dplyr.tidyverse.org.

Affiliation:

Tamás Nagy
ELTE Eötvös Loránd University
Institute of Psychology,
ELTE Eötvös Loránd University,
Budapest, Hungary
E-mail: nagy.tamas@ppk.elte.hu

Márton Kovács
Independent researcher
Budapest, Hungary
E-mail: marton.balazs.kovacs@gmail.com

Alexandra Sarafoglou
University of Amsterdam
Department of Psychology,
University of Amsterdam,
Amsterdam, The Netherlands
E-mail: a.s.g.sarafoglou@uva.nl

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd