Appendix D

Author

Affiliation

Appendix D

This appendix describes the custom `R` functions adapted from Carter et al. (2019) that were used to simulate the three *p*-hacking mechanisms introduced in the main text. We made modifications to these functions to align them with the goals of the present study. The appendix also describes the hierarchical structure of these functions and explains how they work together to simulate the effects of *p*-hacking on original study results.

1 Function Hierarchy

- **simMA():** The primary custom function. It contains all other custom functions, including `simData.QRP()`, `analyB()`, `expDataB()`, `testIt()`, and `outlier()`, as well as R's native functions `sample()`, `repeat()`, etc.
- **simData.QRP():** Contains `expDataB()` and `analyB()`.
- **analyB():** Contains `testIt()` and `outlier()`.

Figure 1 illustrates how the three *p*-hacks are implemented:

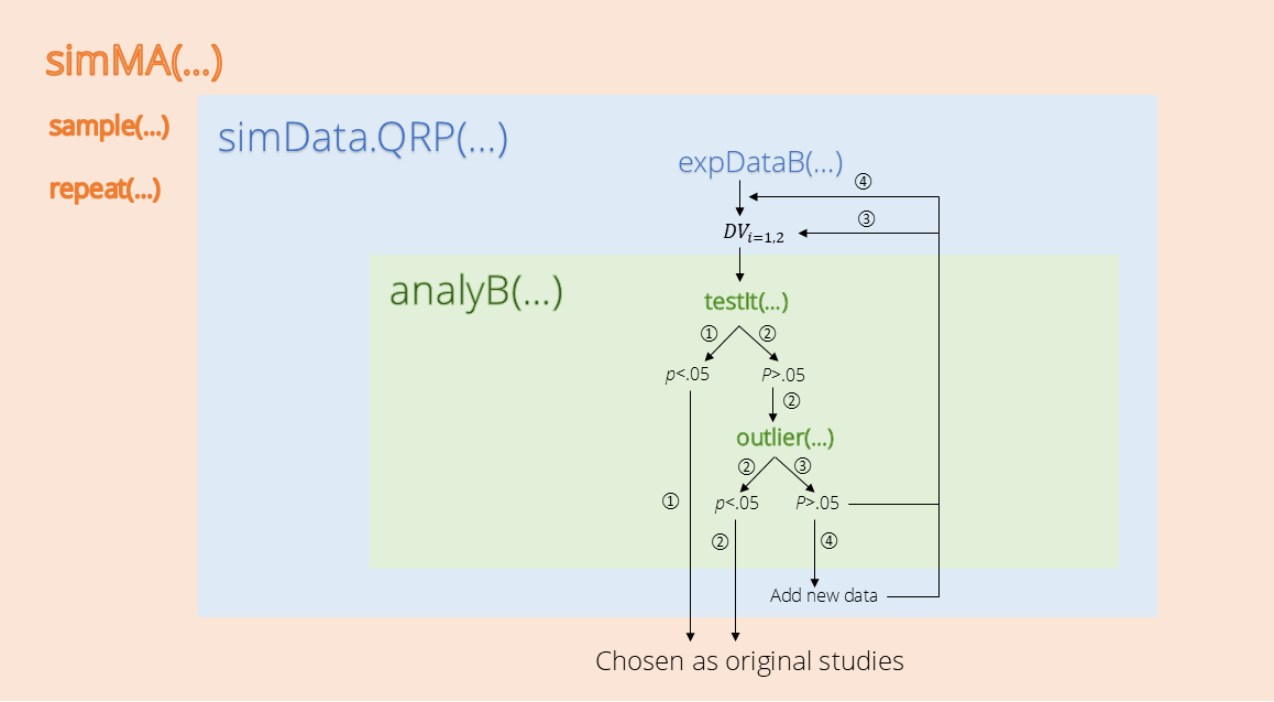

Figure 1

2 Simulation Process

- **Data Generation:** `expDataB()` creates two correlated dependent variables.
- **Initial Testing:** `testIt()` conducts an independent-samples *t*-test on the first dependent variable.
  - If the result is statistically significant and positive, the study is stored in an R data frame as an original study (Path ①① in the diagram).
  - If not, the `outlier()` function identifies and removes outliers, and the modified data are reassessed in `testIt()`. If this reanalysis yields a positive and significant result, the study is saved as an original study (Path ②②②②).
- **Secondary Variable Testing:** If the results are still not positive and significant, the process repeats with the second dependent variable (Path ③③).
- **Data Augmentation:** If no positive and significant effect is found for either dependent variable, `simData.QRP()` supplements additional data and the process is repeated (Path ④④) until the `sample()` and `repeat()` functions determine that the targeted number ($k = 500$) of primary studies have been assembled.

These functions achieve *p*-hacking effects primarily through the implementation of `for` loops and conditional `if-else` statements. A detailed description of these functions is provided below.

3 Function Descriptions

1. **outlier() Function:**
   - **Purpose:** Evaluates whether a generated datum qualifies as an outlier, defined as being larger than 2 standard deviations from the mean.
2. **expDataB() Function:**

- **Purpose:** Supplies data to the `simData.QRP()` function (described below), which subsequently generates p-hacked results.
- **Output:** Generates raw, participant-level data for treatment and control groups, consisting of outcomes of two correlated dependent variables per participant drawn from multivariate normal distributions based on several parameters, including the correlation coefficient between dependent variables (denoted as `cbdv` in the R output), the overall true effect size (denoted as `delta`), individual true effect sizes (denoted as `delta_i`), and the between-study standard deviation (`tau`).

3. **testIt() Function:**

   - **Purpose:** Conducts an independent-samples *t*-test on data derived from `expDataB()` and then calculates effect sizes.
   - **Output:** Reports the effect size (`d`), observed *t*-statistic (`t`) and its associated *p*-value (`p`), total sample size (`N`), and sample sizes for both the treatment and control groups (`n_1`, `n_2`).

4. **analyB() Function:**

   - **Purpose:** Determines whether the results produced by `testIt()` require *p*-hacking to achieve statistical significance, implementing optional selection of dependent variables and outlier exclusion if necessary.
   - **Output:** Identifies and reports the best p-hacked outcomes, including the effect size (`d`) and its variance (`d_v`), observed *t*-statistic (`t`) and its associated *p*-value (`p`), study power (`pwr`), total sample size (`N`), and the matrix of individual true effect sizes (`D`).

5. **simData.noQRP() Function:**

   - **Purpose:** Generates raw, participant-level data for two independent groups: a control group from a standard normal distribution and a treatment group from a normal

distribution based on prespecified parameters (`delta`, `delta_i`, and `tau`). No QRPs are applied to the generated data.

- **Output:** Reports the effect size (`d`), its variance (`d_v`) and standard error (`d_se`), observed *t*-statistic (`t`) and its *p*-value (`p`), study power (`pwr`), total sample size (`N`), and the study-level true effect size (`delta_i`).

6. **simData.QRP() Function:**

   - **Purpose:** Integrates the functions mentioned above and employs the optional-stopping *p*-hacking technique on the generated data. Includes an argument specifying whether a medium or high level of *p*-hacking intensity is implemented.

7. **simMA() Function:**

   - **Purpose:** Simulates various *p*-hacking scenarios using `sample()` and `repeat()` functions. Different levels of *p*-hacking are specified within the `sample()` function by modifying the probability weights, which control how frequently `simData.noQRP()` and `simData.QRP()` are implemented. The `repeat()` function repeatedly adds a new non–*p*-hacked or *p*-hacked study from the corresponding environment until the desired number (`k`) of primary studies is collected as original studies.
   - **Output:** Reports primary studies' effect sizes (`d` and `g` and their variances (`v`) and standard errors (`se`), observed *t*-statistics (`t`) and their *p*-values (`p`), study power (`pwr`), total sample sizes (`N`), and study-level true effect sizes (`delta_i`).