Is Conscious Awareness A Prerequisite to Sense of Agency?

Yimeng (Amy) CHENG

To be done.

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

-- Attaching packages --------------------------------------- tidyverse 1.3.1 --

v ggplot2 3.3.6     v purrr   0.3.4
v tibble  3.2.1     v stringr 1.5.0
v tidyr   1.2.0     v forcats 0.5.1
v readr   2.1.2

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()


Attaching package: 'kableExtra'
```

```
The following object is masked from 'package:dplyr':

    group_rows


Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select
```

Introduction

What exactly is the relationship between self and consciousness? Are they developments that proceed in parallel, or do they follow a specific sequence? Or, could it be that both are merely illusions crafted by the human brain? Philosophers incline to hold the illusionist view, which argues that self-awareness is no more than a mental construction—an experience constructed as humans try to make sense of the world and develop a concept of self (Metzinger, 2004). According to this view, the self is a product of consciousness, a cognitive tool that serves to help us interpret our surroundings. But with recent advances in neuroscience, this functionalist perspective has encountered challenges, especially regarding the question of whether self-experience might have a genuine neural basis (Seth & Tsakiris, 2018; Dehaene, 2014; Tononi et al., 2016). Neuroscientific research has suggested that self-experience may not be simply a construct of consciousness, but rather something embedded in specific neural activity (Blanke & Metzinger, 2009). This development has led scholars to reconsider the relationship between self and consciousness, as well as their potential evolutionary paths. In response, two main perspectives have emerged: illusionist theory and independence theory. Illusion theory maintains that the self is merely a conceptual framework that humans created to understand the world; while it may be supported by certain neural mechanisms, these mechanisms are insufficient to provide a solid, physical basis for the self (Seth, 2021; Wegner & Wheatley, 1999). On the other hand, independence theory suggests that the sense of self can operate even without conscious awareness, functioning as a fundamental biological process rather than an extension of consciousness (Deacon & García-Valdecasas, 2023). To explore these views, a range of experimental and theoretical studies were conducted. Libet (1985)'s readiness potential study stands as a foundational example in support of illusion theory. Libet's work revealed that the brain initiates a readiness potential—a neural precursor to action—before an individual consciously decides to act (Libet, 1985). This finding opened the door to debates about free will and self-attribution, reinforcing the idea that the self may simply be a retrospective attribution. Nevertheless, Libet's study is still in debate, especially regarding whether neural activity alone represents genuine self-experience and whether unconscious processes of self-attribution might exist (Neafsey, 2021; Moore et al., 2010). Wegner and Wheatley

(1999)'s Illusion of Thought-Action theory further strengthens the illusionist stance. They proposed that the feeling of control over one's actions often comes as a retrospective attribution, largely influenced by external cues and expectations (Wegner & Wheatley, 1999). In contrast, research supporting independence theory explored the continuity of self-experience across conscious and unconscious states. Tsakiris and Haggard (2005)'s rubber hand illusion experiment demonstrated how self-attribution can be both flexible and automatic. This effect highlights how self-experience and bodily perception interact, suggesting that self-awareness might not rely entirely on conscious construction but could have a deeper neural basis, potentially independent of conscious oversight. To further illustrate the underlying assumptions between the Illusion Theory and Independence Theory, think about an experience where you are absent-mindedly cutting vegetables and accidentally cut your finger. Seeing the cut, you immediately feel 'I caused this,' even though you may not have been consciously aware of when it happened. Illusion theory assumes that consciousness precedes the self, hence would interpret this as a post-hoc attribution created by consciousness, while independence theory suggests that, even without being conscious of the entire process, your body's underlying mechanisms would automatically attribute the action to self. The implications behind these differing explanations are worth noting. If the self is merely a construct of consciousness, then mental health conditions like dissociative identity disorder or post-traumatic stress disorder (PTSD) could be seen as misconstructions by consciousness, where the focus of treatment would rely on building a healthier self-concept to help patients integrate their sense of self (Gallagher, 2000). However, if the self can exist at an unconscious level, then it may be that deeper, automatic mechanisms of self-attribution are at work in such disorders (Gallagher, 2000). Literature Gap and Research Question While previous research has provided in-depth insights into the relationship between self and consciousness, studies directly investigating whether the sense of self can exist without conscious awareness is lacking. One promising effect for investigating this topic is Intentional Binding (IB; Haggard et al., 2002; Ruess et al., 2018), which occurs when people perceive the time interval between their action and its outcome as shorter, reflecting a sense of agency or causation. For instance, if a person presses a button and a circle appears after 0.5 seconds, they may feel their action caused the circle to appear instantly, with no time gap. When people feel agency, they tend to perceive this interval as shorter. Utilizing this effect, we aim to investigate whether participants can experience a sense of agency (SoA)—indicated by a perceived shortening of time between action and outcome (intentional binding)—when the outcome of their action is masked from conscious awareness?

Hypotheses

Hypothesis 1: Positive Control

Hypothesis 1 serves as the foundational hypothesis, verifying that intentional binding can reliably occur under conscious conditions:

If, in the unmasked operant condition (where participants can consciously see the target stimulus), participants perceive their button press time as closer to the outcome time compared to the unmasked baseline condition (where no target stimulus appears), this would confirm that the experimental design successfully elicits the intentional binding effect, indicating a sense of agency when the outcome is consciously perceived.

Note: If Hypothesis 1 is not supported, the results from Hypothesis 2 would be inconclusive, as we would lack evidence in the design's ability to measure SoA.

Hypothesis 2: Unconscious Conditions

If in the masked operant condition (where the target is suppressed from awareness), participants still perceive a time shortening, then this indicates that SoA occurs even without conscious perception.

Hypothesis 3: Hypothesis on Enhanced Agency with Conscious Awareness

This hypothesis aims to test whether consciousness enhances SoA. If participants' perceived time shift in the unmasked operant condition is greater than in the masked operant condition, then this suggests that conscious awareness strengthens SoA.

Method

Design

This cross-sectional study uses Continuous Flash Suppression (CFS; Pournaghdali & Schwartz, 2020), a technique to mask visual stimuli from conscious awareness, to examine whether people can feel a sense of agency without consciously seeing an outcome.

Ethical Approval

The Social and Behavioral Sciences Institutional Review Board at the University of Chicago has approved this study (IRB23-1353). Informed consent will be obtained from all participants before the experiment, and they will be informed of their right to withdraw at any time. Participants will receive 1.5 course credits for participation.

Participants

Eighty undergraduate University of Chicago students with normal color vision will be recruited to allow detection of medium effect sizes with a power of 0.80 assuming an effect size of $=$ 0.5 and a significance level of 0.01. Participants will be asked about their age, biological sex and handedness.

Procedure

The experiment comprises five within-subject blocks, as follows:

1. Calibration Block: The purpose of this block is to determine the visual contrast threshold for each participant to ensure the target stimulus is effectively suppressed from conscious awareness. Participants will be required to wear red-blue anaglyph glasses. The left eye sees rapidly changing red-colored masks (10 Hz), and the right eye sees a low-contrast blue circle. In 100 trials, a blue circle appears in a random corner of the masked area between 0.25 and 1.75 seconds after the mask onset. Participants indicate which side of the screen they think the blue circle appeared on.

2. Baseline-Masked Block: The CFS masks are presented in the middle of a rotating on-screen clock, with participants still wearing red-blue glasses. Participants will be required to press the space bar at a moment of their choosing, stopping the CFS mask 1-2 seconds later. The clock stops, and after a delay, the clock hand reappears at a random position. Participants then adjust the clock hand to where they believe it was at the time of their button press. For additional check, participants will be asked after each trial if they saw a blue circle; this serves to confirm the effectiveness of the CFS mask in suppressing conscious perception of the target stimulus.

3. Operant-Masked Block: The set up is similar to the baseline-masked block, but with an added target stimulus. After participants press the space bar, a blue circle (target stimulus) appears in a predetermined corner of the masked area 150 ms after the press, remaining on-screen for 200 ms. Participants then complete the same clock-adjustment task to report their perceived time of the button press. As with the baseline-masked block, participants report after each trial whether they saw a blue circle. A few "decoy" trials with a full-contrast circle are randomly added to monitor participants' awareness levels.

4. Baseline-Unmasked Block: This condition is identical to the baseline-masked block, except no CFS masking is used. Participants are not expected to see any target stimulus. Participants will be required to press the button at a self-chosen time, the clock stops, and they adjust the clock hand to the perceived time of their button press, as in previous blocks.

5. Operant-Unmasked Block: This condition mirrors the operant-masked block but without CFS masking, there fore the blue circle (target stimulus) is clearly visible to participants.

2x2 Design Table

Table 1: Table 1. Experimental Design: 2 × 2

Condition	Blue.Circle.Appear	CFS.Masking
2. Baseline-Masked	No	Yes
3. Operant-Masked	Yes	Yes
4. Baseline-Unmasked	No	No
5. Operant-Unmasked	Yes	No

Experimental Design and Corresponding Hypotheses

Table 2: Table 2. Testing Hypotheses

Hypotheses	Conditions.Compared	Question
H1: Positive Control	4 vs. 5	Does binding occur when the outcome is visible?
H2: Unconscious SoA	2 vs. 3	Can binding occur without conscious awareness?
H3: Enhanced SoA	3 vs. 5	Does conscious awareness enhance binding?

Data Analysis Plan (for hypothesis 2)

Note: So far we have only collected data from 65 participants. Therefore, only these participants' data will be analysed in this report.

The primary objective of this data analysis is to investigate whether operant stimuli (in this case, the appearance of a blue circle) influence temporal estimation biases under masked and unmasked conditions. Specifically, we aim to determine if these biases are modulated by unconscious processing, ensuring that conscious awareness of the stimulus does not confound the results. To achieve this, we implement a rigorous screening procedure to exclude trials where participants reported awareness of the operant stimulus.

The data analysis is structured as follows:

Screening Procedure using Fisher's Exact Test:

To ensure that estimation biases are due to unconscious processing, we conduct Fisher's Exact Test to compare awareness rates between operant-masked and baseline-masked conditions for each participant. If a participant reports significantly more awareness in the operant-masked condition than in the baseline-masked condition, their masked trials are excluded from further analysis.

Data Preprocessing and Filtering:

Exclude practice and catch trials that could introduce learning effects or noise. Remove trials where participants reported awareness of the operant stimulus, ensuring that only unconscious processing is measured. Calculation of Huber Mean and Paired Differences (Δ_masked):

Huber Mean is calculated for each participant's temporal estimation bias within both baseline-masked and operant-masked conditions. This robust measure minimizes the influence of outliers, providing a more accurate central tendency estimate than the arithmetic mean. Paired differences (Δ_masked) are computed to assess the influence of operant stimuli on estimation biases.

Paired t-test and Effect Size Calculation (Cohen's d):

We perform paired t-tests on the paired differences (Δ_masked) to test if they significantly differ from zero. Cohen's d is calculated to determine the effect size, providing insight into the practical significance of the operant stimulus's influence on temporal estimation.

Bootstrapped Confidence Intervals and Visualization:

Bootstrapping (5,000 samples) is conducted to calculate 95% confidence intervals for paired differences and Cohen's d. The DABEST package is used for estimation statistics and visualization, enabling a more comprehensive interpretation of the data.

By systematically controlling for conscious awareness and utilizing robust statistical methods, this analysis aims to provide compelling evidence on whether operant stimuli can implicitly modulate temporal estimation biases. This approach enhances the validity and reliability of the findings, contributing to our understanding of the relationship between unconscious processing and temporal perception.

Step-by-Step Analysis

The analysis workflow involves the following steps:

Step 1: Automated Data Loading

Step 2: Screening Procedure using Fisher's Exact Test

Step 3: Data Preprocessing and Filtering

Step 4: Calculation of Huber Mean and Paired Differences (Δ_masked)

Step 5: Paired t-test and Effect Size Calculation (Cohen's d)

Step 6: Bootstrapped Confidence Intervals and Visualization

Step 1: Automated Data Loading

This step involves automatically loading all participant data stored in subfolders under the Awareness_and_Agency_Project_data directory. Each participant's data is organized as follows:

```
Awareness_and_Agency_Project_data/
   sub-01/
       beh/
           sub-01_task-discrimination_beh.tsv
           sub-01_task-masked_beh.tsv
           sub-01_task-unmasked_beh.tsv
   sub-02/
       beh/
           sub-02_task-discrimination_beh.tsv
           sub-02_task-masked_beh.tsv
           sub-02_task-unmasked_beh.tsv
   sub-XX/
        beh/
           sub-xx_task-discrimination_beh.tsv
            sub-XX_task-masked_beh.tsv
            sub-XX_task-unmasked_beh.tsv
```

```r
# Set the main data directory

data_dir <- "Awareness_and_Agency_Project_data"

# Create a file list, which automatically find all _beh.tsv files within subfolders

file_list <- list.files(data_dir, pattern = "-masked_beh.tsv$", recursive = TRUE, full.names

# Read all the tsv files into a list of data frames

data_list <- lapply(file_list, read_tsv)
```

```
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
```

```
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
```

```
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
```

```
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
```

```
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
```

```
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
```

```
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
```

```
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
```

```
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification ---------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Extract and assign descriptive names to each data frame, using regular expression to match

names(data_list) <- gsub(".*/(sub-\\d+)/beh/(sub-\\d+_task-.*).tsv", "\\2", file_list)


# Display the structure of all loaded data frames

str(data_list)
```

Step 2: Screening Procedure using Fisher's Exact Test

Data Screening: Exclusion Criteria

In this study, a screening criterion was implemented to ensure that participants' estimations of event timing were not influenced by conscious awareness of the operant stimulus, which in this experiment was the brief appearance of a blue circle.

Specifically, participants were excluded from further analysis if they reported perceiving the blue circle significantly more often in the Operant-Masked condition than in the Baseline-Masked condition.

This criterion was established because the Baseline-Masked condition presented no operant stimuli, whereas the Operant-Masked condition presented the blue circle in a manner intended to remain subliminal.

If a participant's awareness of the blue circle was significantly higher in the Operant-Masked condition, it would indicate that the masking was ineffective for that participant, and thus their temporal estimations could be influenced by conscious perception.

To evaluate this, **Fisher's exact test** was used to compare the frequency of reported perceptions between the two conditions.

If the difference was statistically significant ($p < 0.05$), it was concluded that the participant was consciously aware of the blue circle in the Operant-Masked condition, leading to the exclusion of all their trials in the masked conditions from further analysis.

This approach ensures that the estimation biases analyzed are reflective of unconscious processing rather than being confounded by conscious awareness of the operant stimulus.

```
#Let's first try with one participant's data

sub.01.masked <-read_tsv("Awareness_and_Agency_Project_data/sub-01/beh/sub-01_task-masked_bel
```

```
Rows: 100 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): stimulus_position
dbl (9): trial, onset, contrast, event_t, event_angle, resp_angle, overest_t...
lgl (5): masked, operant, practice, catch, aware

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
operant_aware <- sum(sub.01.masked$aware == TRUE & sub.01.masked$operant == TRUE)

operant_unaware <- sum(sub.01.masked$aware == FALSE & sub.01.masked$operant == TRUE)

baseline_aware <- sum(sub.01.masked$aware == TRUE & sub.01.masked$operant == FALSE)

baseline_unaware <- sum(sub.01.masked$aware == FALSE & sub.01.masked$operant == FALSE)

print(operant_aware)
```

```
[1] 26
```

```
print(operant_unaware)
```

```
[1] 24
```

```
print(baseline_aware)
```

```
[1] 5
```

```
print(baseline_unaware)
```

```
[1] 45
```

We can see that for subject 01, there are: 26 aware in operant-masked condition; 24 unaware in operant-masked condition; 5 awarenesses in baseline-masked condition; 45 awarenesses in baseline-masked condition.

Let's then produce a contingency table:

```
contingency_table <- matrix(
  c(operant_aware, operant_unaware,
    baseline_aware, baseline_unaware),
  nrow = 2,
  byrow = TRUE
)
rownames(contingency_table) <- c("Operant-Masked", "Baseline-Masked")
colnames(contingency_table) <- c("Aware", "Unaware")
```

Let's conduct Fisher's Exact Test

```
fisher_result <- fisher.test(contingency_table, alternative = "greater") #one-tailed
print(fisher_result)
```

```
    Fisher's Exact Test for Count Data

data:  contingency_table
p-value = 4.286e-06
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
```

```
 3.570692      Inf
sample estimates:
odds ratio
  9.511282
```

The results shows that participant 01 has significantly more "aware" in operant-masked condition than in baseline-masked condition, hence this participant should be excluded.

Let's then create a function to apply Fisher's Exact Test on all participants:

```r
fisher_test_and_screen <- function(df) {
  operant_aware <- sum(df$aware == TRUE & df$operant == TRUE)
  operant_unaware <- sum(df$aware == FALSE & df$operant == TRUE)
  baseline_aware <- sum(df$aware == TRUE & df$operant == FALSE)
  baseline_unaware <- sum(df$aware == FALSE & df$operant == FALSE)

  contingency_table <- matrix(
    c(operant_aware, operant_unaware,
      baseline_aware, baseline_unaware),
    nrow = 2,
    byrow = TRUE
  )

  fisher_result <- fisher.test(contingency_table, alternative = "greater")

  p_value <- fisher_result$p.value

  return(p_value > 0.05)
}


filtered_data_list <- data_list[lapply(data_list, fisher_test_and_screen) == TRUE]

length(filtered_data_list)
```

```
[1] 53
```

After screening, there are 53 participants left.

Step 3 Data Preprocessing and Filtering

Define a function to only inclue real trials and exclude catch and practice trials. Also exclude trials where participants report that they are aware of the blue circle.

```
preprocess_data <- function(df) {
  df <- df %>%
    # Exclude Practice and Catch Trials
    filter(practice == FALSE, catch == FALSE) %>%
    # Exclude Trials Where Participants Reported Awareness
    filter(aware == FALSE)
  return(df)
}

# Apply this function to all dataframe

processed_data_list <- lapply(filtered_data_list, preprocess_data)
```

Step 4: Calculation of Huber Mean and Paired Differences (Δ_masked)

We calculate the **Huber Mean** for estimation bias within each condition, which is a robust measure of central tendency that minimizes the impact of outliers.

4.1 Huber Mean Calculation

```
#Create a new condition column for every table to benefit later comparasion

processed_data_list <- lapply(processed_data_list, function(data) {
  data <- data %>%
    mutate(
      condition = case_when(
        masked == TRUE & operant == TRUE ~ "Operant-Masked",
        masked == TRUE & operant == FALSE ~ "Baseline-Masked",
        masked == FALSE & operant == TRUE ~ "Operant-Unmasked",
        masked == FALSE & operant == FALSE ~ "Baseline-Unmasked"
      )
    )
  return(data)
})
```

```r
#Calculate huber mean. Using "if" conditions to deal with 0s.
calculate_huber <- function(df) {
  if (!("condition" %in% names(df)) | !("overest_t" %in% names(df))) {
    stop("Data does not contain required columns.")
  }

  masked_vals <- df$overest_t[df$condition == "Operant-Masked"]
  baseline_vals <- df$overest_t[df$condition == "Baseline-Masked"]

  if (length(masked_vals) == 0 || all(is.na(masked_vals)) || mad(masked_vals, na.rm = TRUE) =
    huber_masked <- NA
  } else {
    huber_masked <- huber(masked_vals)$mu
  }

  if (length(baseline_vals) == 0 || all(is.na(baseline_vals)) || mad(baseline_vals, na.rm = T
    huber_baseline <- NA
  } else {
    huber_baseline <- huber(baseline_vals)$mu
  }
  df$masked <- huber_masked
  df$baseline <- huber_baseline
  return(c(masked = huber_masked, baseline = huber_baseline))
}

huber_results <- lapply(processed_data_list, calculate_huber)

print(huber_results)
```

```
$`sub-02_task-masked_beh`
     masked    baseline
-0.07777055 -0.08140090

$`sub-03_task-masked_beh`
   masked  baseline
0.8395052 0.1839998

$`sub-04_task-masked_beh`
     masked    baseline
-0.01887973  0.07716013

$`sub-05_task-masked_beh`
```

```
     masked    baseline
-0.02434487 -0.04404212

$`sub-06_task-masked_beh`
     masked    baseline
-0.04054244 -0.03593740

$`sub-08_task-masked_beh`
     masked    baseline
-0.05624423 -0.02436646

$`sub-09_task-masked_beh`
      masked     baseline
-0.042810422 -0.009908936

$`sub-11_task-masked_beh`
     masked    baseline
-0.02600677 -0.03357972

$`sub-12_task-masked_beh`
     masked    baseline
-0.05953439 -0.09578084

$`sub-13_task-masked_beh`
     masked    baseline
0.076982165 0.009710054

$`sub-14_task-masked_beh`
    masked   baseline
0.05781542 0.03356385

$`sub-15_task-masked_beh`
      masked     baseline
 0.005567239 -0.003364169

$`sub-16_task-masked_beh`
     masked    baseline
-0.06809820 -0.03347549

$`sub-17_task-masked_beh`
     masked    baseline
 0.04449899 -0.09850422
```

```
$`sub-18_task-masked_beh`
     masked    baseline
-0.06142437 -0.08615760

$`sub-20_task-masked_beh`
     masked    baseline
-0.04741460 -0.02138735

$`sub-21_task-masked_beh`
    masked   baseline
0.08459439 0.08911036

$`sub-22_task-masked_beh`
      masked     baseline
 0.006301734 -0.019766999

$`sub-23_task-masked_beh`
     masked    baseline
0.006815872 0.031223468

$`sub-25_task-masked_beh`
     masked    baseline
-0.01765858  0.03285490

$`sub-26_task-masked_beh`
    masked   baseline
0.03198587 0.04455082

$`sub-27_task-masked_beh`
     masked    baseline
-0.03146107 -0.03157380

$`sub-28_task-masked_beh`
    masked   baseline
0.05390762 0.04715404

$`sub-29_task-masked_beh`
     masked    baseline
-0.02220443  0.02855456

$`sub-30_task-masked_beh`
      masked     baseline
-0.002620706 -0.012061157
```

```
$`sub-33_task-masked_beh`
      masked     baseline
 0.001797892 -0.031910644

$`sub-34_task-masked_beh`
     masked    baseline
-0.02233763 -0.04171829

$`sub-35_task-masked_beh`
       masked      baseline
-0.0287404938  0.0002815778

$`sub-36_task-masked_beh`
     masked    baseline
-0.06641223 -0.04843388

$`sub-37_task-masked_beh`
      masked     baseline
-0.027534314  0.008475643

$`sub-39_task-masked_beh`
      masked     baseline
-0.045445727  0.005274499

$`sub-40_task-masked_beh`
     masked    baseline
-0.02314523 -0.04402650

$`sub-41_task-masked_beh`
     masked    baseline
-0.07900514  0.02553953

$`sub-42_task-masked_beh`
      masked     baseline
-0.019277735  0.006187309

$`sub-45_task-masked_beh`
     masked    baseline
 0.04348129 -0.06375877

$`sub-46_task-masked_beh`
    masked   baseline
```

```
0.02444495 0.03464431

$`sub-47_task-masked_beh`
     masked    baseline
-0.02228577 -0.01207329

$`sub-48_task-masked_beh`
       masked      baseline
-0.0376703752  0.0002593011

$`sub-49_task-masked_beh`
      masked     baseline
-0.007801433 -0.013197171

$`sub-51_task-masked_beh`
     masked    baseline
-0.05611392 -0.10680393

$`sub-52_task-masked_beh`
     masked    baseline
-0.03059546  0.10730466

$`sub-53_task-masked_beh`
    masked   baseline
0.02312003 0.06036439

$`sub-54_task-masked_beh`
     masked    baseline
-0.05236375  0.03495504

$`sub-55_task-masked_beh`
    masked   baseline
0.07710847 0.05232996

$`sub-56_task-masked_beh`
     masked    baseline
0.002907757 0.002763266

$`sub-57_task-masked_beh`
      masked     baseline
-0.012310667 -0.007378135

$`sub-58_task-masked_beh`
```

```
      masked     baseline
 0.002446029 -0.009619935

$`sub-59_task-masked_beh`
     masked    baseline
-0.02324531 -0.04600326

$`sub-60_task-masked_beh`
     masked    baseline
-0.02003996 -0.01125864

$`sub-62_task-masked_beh`
     masked    baseline
-0.04285290 -0.03358288

$`sub-63_task-masked_beh`
      masked     baseline
-0.001595087  0.010448228

$`sub-64_task-masked_beh`
     masked    baseline
0.022525990 0.005004993

$`sub-65_task-masked_beh`
    masked   baseline
0.04500341 0.05930949
```

4.2: Paired Differences (Δ_masked) Calculation

```r
calculate_diff <- function(huber_vals) {
  delta_masked <- huber_vals["masked"] - huber_vals["baseline"]
  return(delta_masked)
}

delta_masked_list <- lapply(huber_results, calculate_diff)

delta_masked <- unlist(delta_masked_list)

print(delta_masked)
```

```
sub-02_task-masked_beh.masked sub-03_task-masked_beh.masked
```

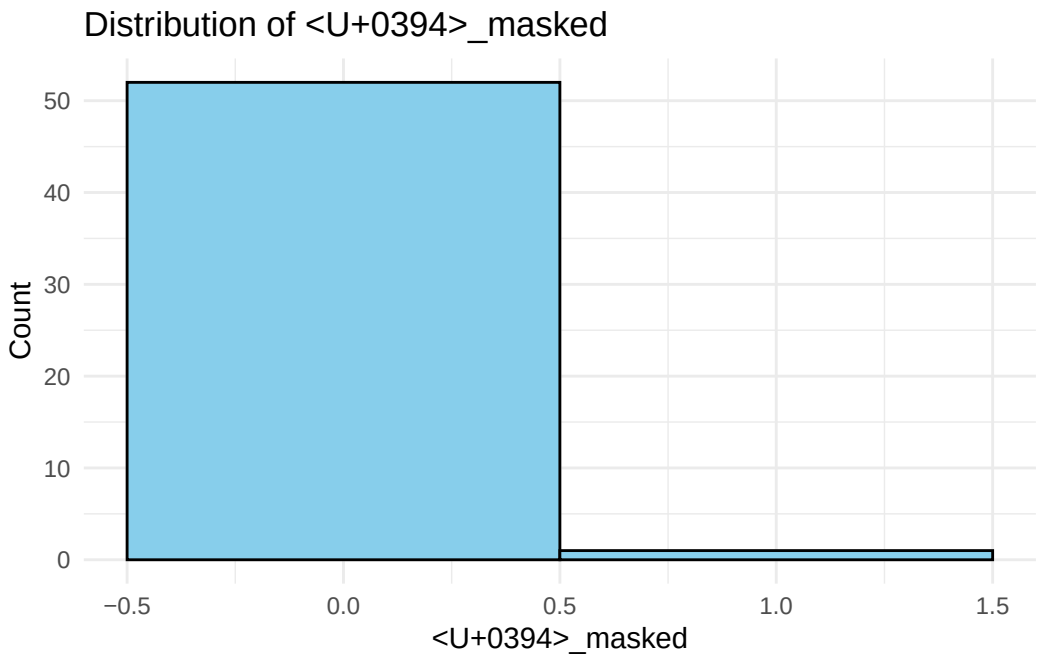
```
                0.0036303499                 0.6555054675
sub-04_task-masked_beh.masked sub-05_task-masked_beh.masked
               -0.0960398575                 0.0196972533
sub-06_task-masked_beh.masked sub-08_task-masked_beh.masked
               -0.0046050387                -0.0318777678
sub-09_task-masked_beh.masked sub-11_task-masked_beh.masked
               -0.0329014859                 0.0075729497
sub-12_task-masked_beh.masked sub-13_task-masked_beh.masked
                0.0362464473                 0.0672721108
sub-14_task-masked_beh.masked sub-15_task-masked_beh.masked
                0.0242515701                 0.0089314082
sub-16_task-masked_beh.masked sub-17_task-masked_beh.masked
               -0.0346227118                 0.1430032094
sub-18_task-masked_beh.masked sub-20_task-masked_beh.masked
                0.0247332291                -0.0260272455
sub-21_task-masked_beh.masked sub-22_task-masked_beh.masked
               -0.0045159743                 0.0260687329
sub-23_task-masked_beh.masked sub-25_task-masked_beh.masked
               -0.0244075954                -0.0505134891
sub-26_task-masked_beh.masked sub-27_task-masked_beh.masked
               -0.0125649486                 0.0001127217
sub-28_task-masked_beh.masked sub-29_task-masked_beh.masked
                0.0067535871                -0.0507589927
sub-30_task-masked_beh.masked sub-33_task-masked_beh.masked
                0.0094404509                 0.0337085355
sub-34_task-masked_beh.masked sub-35_task-masked_beh.masked
                0.0193806665                -0.0290220716
sub-36_task-masked_beh.masked sub-37_task-masked_beh.masked
               -0.0179783503                -0.0360099566
sub-39_task-masked_beh.masked sub-40_task-masked_beh.masked
               -0.0507202261                 0.0208812695
sub-41_task-masked_beh.masked sub-42_task-masked_beh.masked
               -0.1045446720                -0.0254650445
sub-45_task-masked_beh.masked sub-46_task-masked_beh.masked
                0.1072400601                -0.0101993584
sub-47_task-masked_beh.masked sub-48_task-masked_beh.masked
               -0.0102124769                -0.0379296763
sub-49_task-masked_beh.masked sub-51_task-masked_beh.masked
                0.0053957376                 0.0506900126
sub-52_task-masked_beh.masked sub-53_task-masked_beh.masked
               -0.1379001184                -0.0372443667
sub-54_task-masked_beh.masked sub-55_task-masked_beh.masked
               -0.0873187940                 0.0247785103
```

```
sub-56_task-masked_beh.masked sub-57_task-masked_beh.masked
                0.0001444908                 -0.0049325318
sub-58_task-masked_beh.masked sub-59_task-masked_beh.masked
                0.0120659637                  0.0227579577
sub-60_task-masked_beh.masked sub-62_task-masked_beh.masked
               -0.0087813206                 -0.0092700259
sub-63_task-masked_beh.masked sub-64_task-masked_beh.masked
               -0.0120433147                  0.0175209971
sub-65_task-masked_beh.masked
               -0.0143060774
```

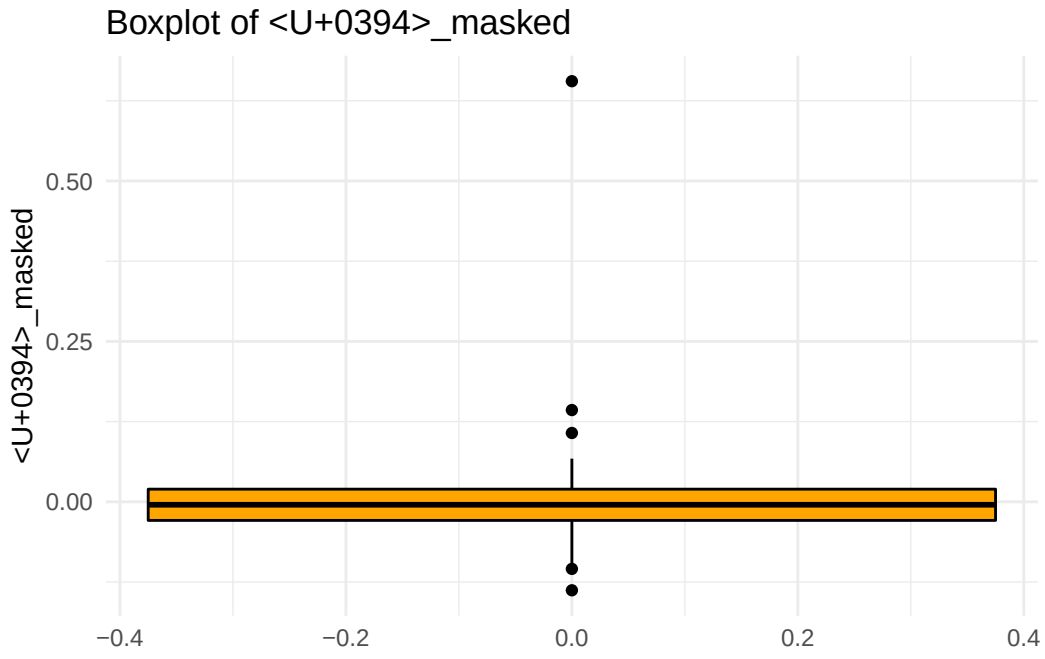
Visualization

```r
delta_data <- data.frame(delta_masked = delta_masked)

# Histogram
ggplot(delta_data, aes(x = delta_masked)) +
  geom_histogram(binwidth = 1, color = "black", fill = "skyblue") +
  labs(title = "Distribution of Δ_masked", x = "Δ_masked", y = "Count") +
  theme_minimal()
```



```r
# Boxplot
ggplot(delta_data, aes(y = delta_masked)) +
  geom_boxplot(fill = "orange", color = "black") +
  labs(title = "Boxplot of Δ_masked", y = "Δ_masked") +
  theme_minimal()
```



Step 5: Paired t-test and Effect Size Calculation (Cohen's d) [To be compeleted]

Objective: Calculate Paired Differences (Δ_masked):

Δ_masked = Operant-Masked Huber Mean - Baseline-Masked Huber Mean

Measures the effect of intentional action on time estimation bias in the Masked condition.

Statistical Tests:

One-tailed Paired t-test: Test the hypothesis: Δ_masked > 0 (indicating intentional binding).

Effect Size Calculation:

Cohen's d to measure the magnitude of the effect.