

**AI-Powered Statistical Analysis Tool For Single Subject Research in Behavioral Studies:  
Phase II**

Venkat Abhiram Nelluri

Project advisors  
Dr. Hanieh Shabanian,  
Dr. Sergey Butakov  
Dr. Sara Peck

Project report

Submitted to the Faculty of Arts and Science,  
Western New England University Springfield, MA

in Partial Fulfillment of the Requirements for the  
Capstone Project for the Degree

MASTER OF COMPUTER SCIENCE

Western New England University

May 2025

# 1. INTRODUCTION

A collaborative effort to improve statistical approaches in Single-Case Design (SCD) research is the AI-Powered Statistical Analysis Tool for Single-Subject Research in Behavioral Studies. This initiative tackles important behavioral research obstacles such as small sample sizes, non-normal data distributions, and autocorrelation problems under the direction of Drs. Haineh Shabanian, Sergey Butakov, and Sara Peck. Because of the high level of skill required for current statistical analysis approaches, practitioners face obstacles.

Single-Case Design (SCD) studies are widely used in behavioral research to evaluate the impact of interventions on individuals over time. However, they pose significant statistical challenges, such as small sample sizes, autocorrelation, non-normal data distributions, and variability across phases. Traditional statistical methods used in this context often require advanced expertise and manual configuration, creating barriers for many practitioners and researchers. To address these limitations, this project aims to develop an **AI-powered statistical analysis tool** that simplifies the analytical process while preserving statistical rigor and enhancing usability.

The primary objective of this project is to automate statistical decision-making in SCD research by integrating advanced statistical methods including Bayesian inference, bootstrapping, and mixed-effects modeling into an intuitive, web-based platform. This solution not only eliminates the need for deep statistical knowledge but also increases the speed, accuracy, and reliability of SCD analysis. By tailoring test selection to the structure and characteristics of the uploaded dataset, the tool reduces the risk of misapplication and helps users generate reproducible, evidence-based insights.

To guide development, the project is organized into six structured phases:

1. **Research Phase** – Reviewing existing statistical techniques used in single-subject research and identifying integration opportunities.
2. **Documentation Phase** – Outlining system requirements, defining features, and collecting user needs.
3. **Design Phase** – Developing the system architecture, data flow, and user interface prototypes.
4. **Coding Phase** – Implementing the core analytical functions, including effect size metrics, hypothesis testing methods, and visualizations.
5. **Testing Phase** – Verifying functional accuracy using behavioral datasets and synthetic trials.
6. **User Feedback Phase** – Iteratively refining features based on input from behavioral researchers and domain experts.

As the lead developer, responsibilities include algorithm design, front-end development using Streamlit, implementation of statistical models, and optimization of the user interface for ease of use. The tool currently supports three major experimental designs ABA Reversal, Alternating Treatment, and Multiple Baseline and includes a dedicated Bayesian module for probabilistic outcome evaluation. Ultimately, this project contributes to the behavioral science community by delivering a scalable and robust application that automates statistical processes, standardizes reporting formats, and empowers users to conduct high-quality analyses with minimal effort. The following sections detail the foundational literature, methodology, implemented features, and achieved outcomes of this AI-powered analytical platform.

## 2. RELATED WORKS

Single-Case Design (SCD) has long been a fundamental methodology in behavioral research, particularly within applied behavior analysis, special education, and clinical psychology. Historically, evaluating treatment effects in SCD relied heavily on visual analysis, a method often criticized for its subjectivity, variability among raters, and challenges in managing statistical complexities such as autocorrelation, non-normality, and small sample sizes. Recognizing the need for an objective, standardized, and accessible analytical framework, researchers have increasingly explored statistical modeling and artificial intelligence for SCD analysis. This literature review synthesizes key studies that have influenced the development of the current AI-powered SCD analysis tool, incorporating advancements in Bayesian inference, effect size estimation, hierarchical modeling, machine learning, and user-friendly tool design.

William R. Shadish ([Shadish, 2014]) critically examined the limitations of relying solely on visual inspection in SCD and advocated for incorporating formal statistical methodologies. He argued that traditional effect size metrics and overlap statistics lack rigor, particularly when faced with autocorrelation or uneven phase lengths. His work highlighted the importance of standardized effect sizes, such as Hedges'  $g$ , and the adoption of multilevel and Bayesian models to account for both within-subject and between-subject variability. These insights strongly influenced this project by reinforcing the necessity of automated statistical analyses that enhance methodological robustness and align with evidence-based practice.

Building on this need for statistical rigor, Lu, Scott, and Raghavan ([Lu, Scott, & Raghavan, 2018]) introduced a Bayesian hierarchical framework specifically designed for single-subject research in clinical applications, such as post-stroke rehabilitation. Their approach used posterior predictive distributions to establish template null distributions, providing clinicians with a reliable means of comparing test subjects' behavior to that of a healthy population. By accommodating variability between individuals and addressing small sample sizes, their methodology overcame limitations of conventional parametric tests. The present project integrates this approach by implementing Bayesian posterior simulations within an AI interface, streamlining hypothesis testing and reducing the burden of manual statistical calculations.

Pustejovsky, Hedges, and Shadish ([Pustejovsky, Hedges, & Shadish, 2014]) contributed a framework for computing design-comparable effect sizes in multiple baseline studies. Their research demonstrated how hierarchical linear models could estimate standardized effect sizes that remain consistent across different experimental designs, facilitating the synthesis of findings from both SCDs and larger group-based studies. Their approach to managing autocorrelation and estimating random effects directly informs the effect size computation mechanisms of the AI tool, ensuring statistically sound results across various SCD structures.

Machine learning has emerged as a valuable addition to SCD analysis, offering improved objectivity in evaluating intervention effects. Lanovaz, Rapp, and Fletcher ([Lanovaz, Rapp, & Fletcher, 2020]) explored machine learning applications, including dense neural networks, support vector machines, and random forests, to analyze ABAB graph data. Their findings indicated that these models outperformed conventional structured visual methods, such as the dual-criteria (DC) approach, in terms of accuracy, Type I error rates, and statistical power. By training models on extensive datasets of published SCD graphs, they paved the way for automating phase change detection and intervention impact assessment. This project incorporates these advances by leveraging supervised learning models for trend classification and phase segmentation.

The broader movement toward accessible and standardized SCD methodology is also a key consideration. Barnard-Brak and Richman ([Barnard-Brak & Richman, 2022]) emphasized the importance of statistical

standardization, urging researchers to adopt hierarchical modeling, meta-analytic techniques, and automated decision-support systems. Their recommendations underscore the necessity for software tools that simplify complex analyses and facilitate evidence synthesis across studies. The AI tool aligns with this perspective by providing an intuitive interface that automates statistical tests, performs methodological checks, and generates APA-style reports, making sophisticated analyses accessible to researchers with varying levels of statistical expertise.

Randomization tests offer another robust statistical approach for small-N designs. Tang and Landes ([Tang & Landes, 2020]) demonstrated how these tests, which do not rely on traditional assumptions of normality or independence, are well-suited to behavioral research with skewed and serially dependent data. Their study confirmed that randomization methods yield reliable inference even with limited data points. This project integrates these insights by including randomization testing as an optional method, expanding the flexibility and reliability of hypothesis testing within the tool.

Innovations in behavioral coding further contribute to the automation of SCD analysis. Lønfeldt et al. ([Lønfeldt et al., 2022]) applied machine learning techniques such as facial expression recognition, gaze tracking, and motion analysis to psychiatric assessments, reducing human coder bias and improving diagnostic accuracy. Their approach demonstrated the potential of AI to standardize behavioral measurement. In response, the current tool supports semi-automated behavioral tagging and pattern recognition, minimizing the reliance on manual data annotation.

The integration of technology into SCED practice is exemplified by the mobile application SCD-MVA, described by Moeyaert et al. ([Moeyaert et al., 2021]). This app features real-time data visualization, masked visual analysis, and automated randomization procedures, enhancing accessibility for researchers operating in remote or virtual environments. Expanding on these capabilities, the AI-powered platform developed in this project introduces real-time Bayesian updating, group-level modeling, and downloadable statistical reports, further improving usability and reproducibility.

In summary, the AI-powered statistical tool for SCD research builds upon a strong foundation of methodological advancements. It incorporates Bayesian inference, effect size estimation, hierarchical modeling, randomization testing, and machine learning to provide a comprehensive and automated solution for behavioral researchers. By integrating insights from established literature and prioritizing accessibility and standardization, the tool addresses long-standing challenges in SCD analysis, advancing the field and supporting evidence-based behavioral science.

## KEYWORDS:

**Behavioral Data Analysis:** Evaluating behavioral interventions becomes a journey of discovery when we combine statistical methods and AI-powered tools. By closely examining patterns, trends, and the effects of treatments in Single-Case Experimental Designs (SCEDs), we gain valuable insights that empower data-driven decisions. It's about turning raw data into meaningful stories that shape better strategies for individuals' well-being.

**Bayesian Inference:** Imagine a method that evolves and adapts as new evidence comes in, refining probabilities to help make better decisions even when the future feels uncertain. Instead of sticking to rigid, fixed estimates, it creates a dynamic picture a posterior probability distribution that captures the richness of the data. This makes it an incredibly powerful tool for unraveling the complexities of behavioral data and gaining deeper insights.

**Randomization Tests:** Picture a flexible approach to analyzing treatment effects in Single-Case Experimental Designs (SCEDs) one that doesn't rely on rigid assumptions like normality or data independence. Instead, these non-parametric methods shuffle and reassign data points to create a variety of possible outcome distributions. This allows for strong, reliable hypothesis testing, even in complex scenarios, making it a powerful tool for uncovering meaningful patterns in behavioral research.

**Effect Size Computation:** Think of this to measure just how impactful an intervention is in Single-Case Experimental Designs (SCEDs). Unlike p-values, which only tell us whether something is statistically significant, effect sizes go a step further by highlighting the practical importance of the results. They make it easier to compare findings across different studies, giving a clearer and more meaningful interpretation of the data's real-world implications.

**Logistic Regression for Single-Case Analysis:** Imagine a method designed to predict outcomes with only two possibilities like success or failure in Single-Case Experimental Designs (SCEDs). This statistical technique estimates the likelihood of an event happening based on certain predictors, making it a valuable tool for assessing whether an intervention has caused meaningful behavioral changes. It transforms complex data into actionable insights, helping to uncover the true impact of an intervention.

Building upon these existing frameworks and innovations, the project's methodology was structured to leverage the most relevant tools, development environments, and agile practices to implement a robust solution.

**Table 1:** Comparison of Statistical Methods for Analyzing Single-Case Designs in Behavioral Research.

Method	Strengths	Limitations	Relevance to This Project
<b>Generalized Additive Models (GAMs)</b> (Shadish, 2014)	Captures nonlinear relationships; flexible trend analysis	Requires statistical expertise for implementation	Helps model behavioral trends more accurately in single-case designs
<b>Bayesian Analysis</b> (Lu, Scott, & Raghavan, 2018)	Provides posterior probability estimation; effective for small samples	Computationally demanding; requires expertise	Supports <b>automated effect size estimation and hierarchical modeling</b>
<b>Multilevel Modeling (HLM)</b> (Shadish, 2014)	Accounts for within-subject and between-subject variability in longitudinal studies	Complex modeling structure; requires large computational power	Improves accuracy in analyzing intervention effects across multiple baselines
<b>Randomization Tests</b> (Tang & Landes, 2020)	No assumptions about normality; robust for small samples	Requires high computational resources	Useful for <b>validating treatment effects</b> in SCED studies
<b>Effect Size Computation</b> (Barnard-Brak et al., 2022)	Standardized measure for intervention impact; improves comparability	Some methods depend on independence assumptions	Ensures <b>meta-analytic consistency</b> in SCED research
<b>Machine Learning for Behavioral Coding</b> (Lønfeldt et al., 2022)	Automates behavioral classification; reduces human coder bias	Accuracy depends on the quality of training data	Enables <b>AI-driven behavior tracking and trend detection</b>
<b>Monte Carlo Simulations</b> (Friedel et al., 2022)	Models distribution of possible results based on repeated sampling	Computationally expensive	Supports <b>probabilistic modeling</b> of intervention effects
<b>PND &amp; Advanced Testing</b> (Tarlow & Penland, 2016)	Provides a structured, visual method for analyzing intervention impact	Relies on observation of independence assumptions	Improves <b>statistical rigor</b> in evaluating treatment effectiveness

Method	Strengths	Limitations	Relevance to This Project
<b>SCD-MVA Mobile Application</b> (Moeyaert et al., 2021)	Enables remote SCED research; real-time data visualization	Limited to mobile platforms	Supports <b>automated experimental design and statistical testing</b>
<b>AI-Powered Single-Case Analysis</b> (Lanovaz et al., 2020)	Reduces interrater disagreement; enhances reproducibility in SCD analysis	Limited to binary classification	Strengthens <b>automated decision-making and machine learning integration</b>

## 3. METHODOLOGY

### 3.1 Tools:

In developing the AI-powered statistical analysis tool for single-case design (SCD) research, we strategically selected tools that align with both statistical rigor and ease of user interaction. The project is implemented entirely in Python, chosen for its readability and its rich ecosystem of scientific libraries that support advanced statistical computation and data processing.

#### Development Environment

**Visual Studio Code** proved to be a powerful ally for writing, debugging, and testing Python code. Its versatile features, such as Python-specific extensions, GitHub integration, and robust debugging tools, made the development process smoother and more efficient. These capabilities streamlined the creation of statistical models, machine learning algorithms, and user interface integration while ensuring seamless version control.

Why we use it: Visual Studio Code (VS Code) was chosen for its efficient, lightweight, and versatile environment tailored to writing, debugging, and testing Python scripts. The built-in debugger played a critical role in identifying and fixing errors in statistical computations, such as Bayesian logistic regression, mixed-effects models, and bootstrapping functions, ensuring accurate results. Its integrated terminal allowed for seamless execution of the Streamlit-based UI, supporting real-time testing of dataset uploads and study design detection. Furthermore, VS Code's GitHub integration made version control and collaboration straightforward, enhancing the efficiency of managing updates and refining code.

- **Python:** Serves as the core programming language for both logic and interface due to its simplicity and powerful data science capabilities.
- **Streamlit:** Selected as the front-end framework to rapidly build an interactive web-based interface, allowing users to upload datasets, perform analyses, and view results in real time.

#### Server and Front-End Framework

- The tool is designed as a single-page web app using **Streamlit**, which inherently serves both front-end and back-end needs for data science applications. No external Flask server was used, simplifying deployment.

#### Libraries and Frameworks

- **NumPy:** Handles numerical computations and supports the implementation of effect size calculations like PND, PEM, and IRD.
- **Pandas:** Used for dataset manipulation, especially for trimming session data and pivoting phases across designs.
- **SciPy:** Provides statistical functions such as t-tests, Wilcoxon tests, and randomization methods, including Shapiro-Wilk tests for normality.



- **Statsmodels:** Enables advanced modeling, including the implementation of Mixed Effects Models for hierarchical data structures.
- **Matplotlib & Seaborn:** Power the visualizations, including histograms, Q-Q plots, and boxplots for data distribution and normality checking.

This combination of tools ensures robust statistical functionality while maintaining an intuitive user interface. The modular design and dependency on widely used open-source libraries also facilitate future enhancements and collaborative development.

## **3.2 Development Process**

### **Development Process Used and Justification**

#### **Agile Development Methodology**

The development of the AI-powered SCD Analysis Tool followed an Agile development approach, selected for its flexibility and responsiveness to evolving project needs—particularly valuable when dealing with iterative improvements in statistical modeling and behavioral data analysis workflows.

#### **Implementation of Agile Principles**

**Iterative and Flexible Development** The tool was built in structured sprints. Early sprints focused on foundational components such as data upload mechanics, Streamlit UI configuration, and initial statistical computation (e.g., t-tests, Wilcoxon). Later sprints incrementally integrated advanced capabilities, such as Bayesian logistic regression, bootstrap CI visualization, and multi-phase design support (ABA, Alternating Treatment, Multiple Baseline).

#### **Adaptability and Refinement**

Insights from initial builds led to prioritizing features such as APA-style output formatting, interactive normality checks, and last-5-session trimming to better reflect practitioner needs. Frequent testing of statistical output readability led to enhanced display functions (`display_results()`), graphical summaries, and structured feedback sections.

#### **Sprint Goals and Deliverables**

- **Goal:** Integrate flexible effect size metrics (PND, PEM, IRD)  
**Outcome:** Added selectable effect size logic in the sidebar with automated computation and dynamic output display.
- **Goal:** Enable Bayesian modeling for intervention analysis  
**Outcome:** Implemented Bayesian posterior visualization and logistic probability estimation using `scipy.stats.beta`.
- **Goal:** Improve usability for behavioral researchers  
**Outcome:** Designed a clean, tab-based Streamlit interface; added input validation, normality plots, and summary statistics.

## **Collaboration and User Feedback**

Feedback from early testers, including behavioral research students and advisors, was vital. Their request for clearer output led to refinements in APA formatting and interpretive feedback for each statistical result. Additional requests to view intervention effectiveness prompted the inclusion of dynamic graphs such as boxplots and Bayesian posterior curves.

## **Feedback-Driven Refinement**

Throughout development, feedback loops enabled rapid changes. For example:

- Requests for comparative Bayesian analysis led to a dedicated “Bayesian Logistic Add-on” tab.
- Confusion around statistical significance led to adding automated interpretations (e.g., “Significant” vs. “Not Significant”) in results output.

## **Risk Mitigation**

Sprint reviews identified and resolved common issues:

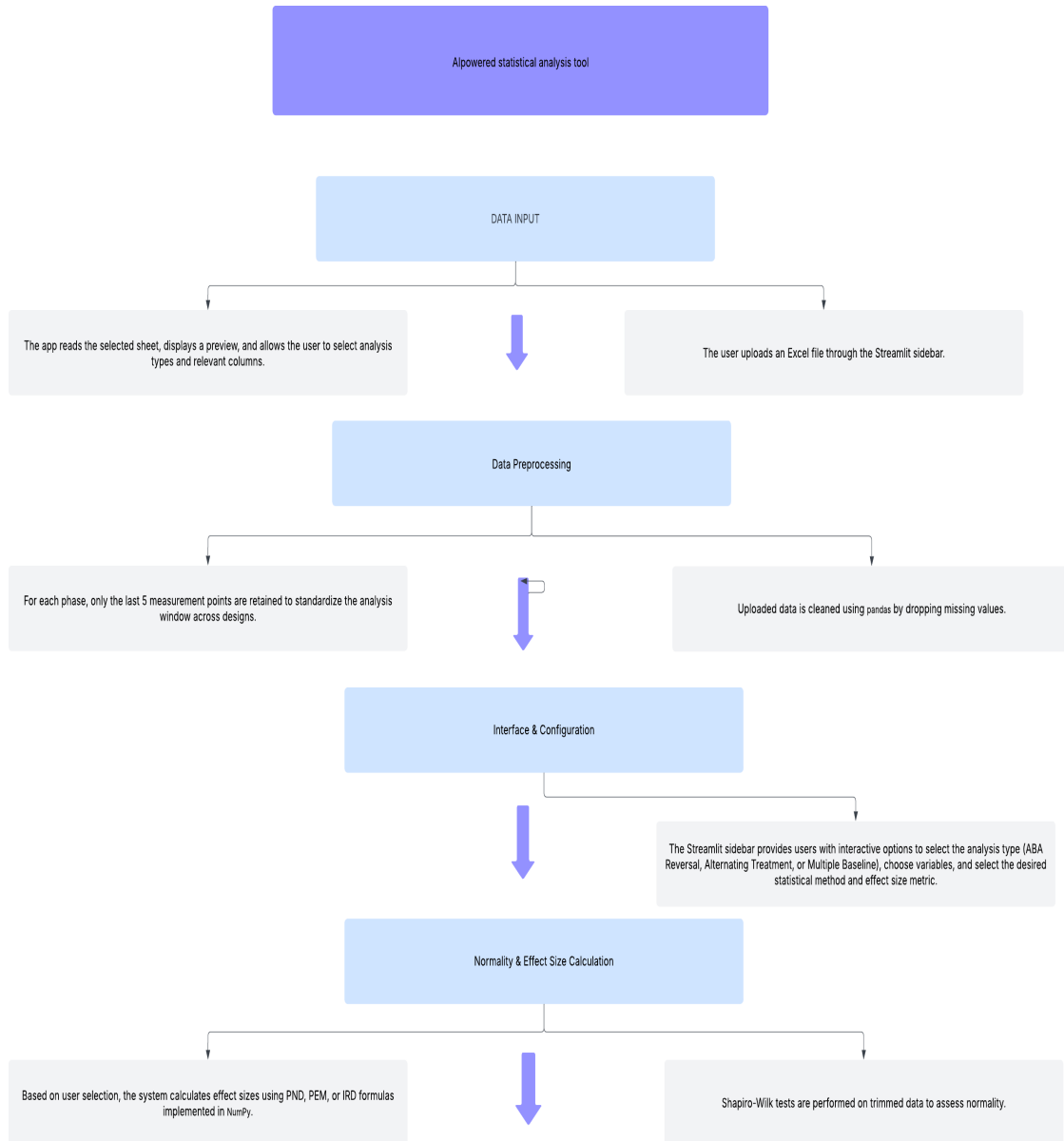
- Issue: Inconsistent behavior when switching between design types  
Fix: Refactored toggle logic to ensure state isolation across ABA, MBD, and AT analysis blocks.
- Issue: Confusing statistical outputs  
Fix: Created the `display_results()` wrapper for standardized APA-style formatting and streamlined feedback presentation.

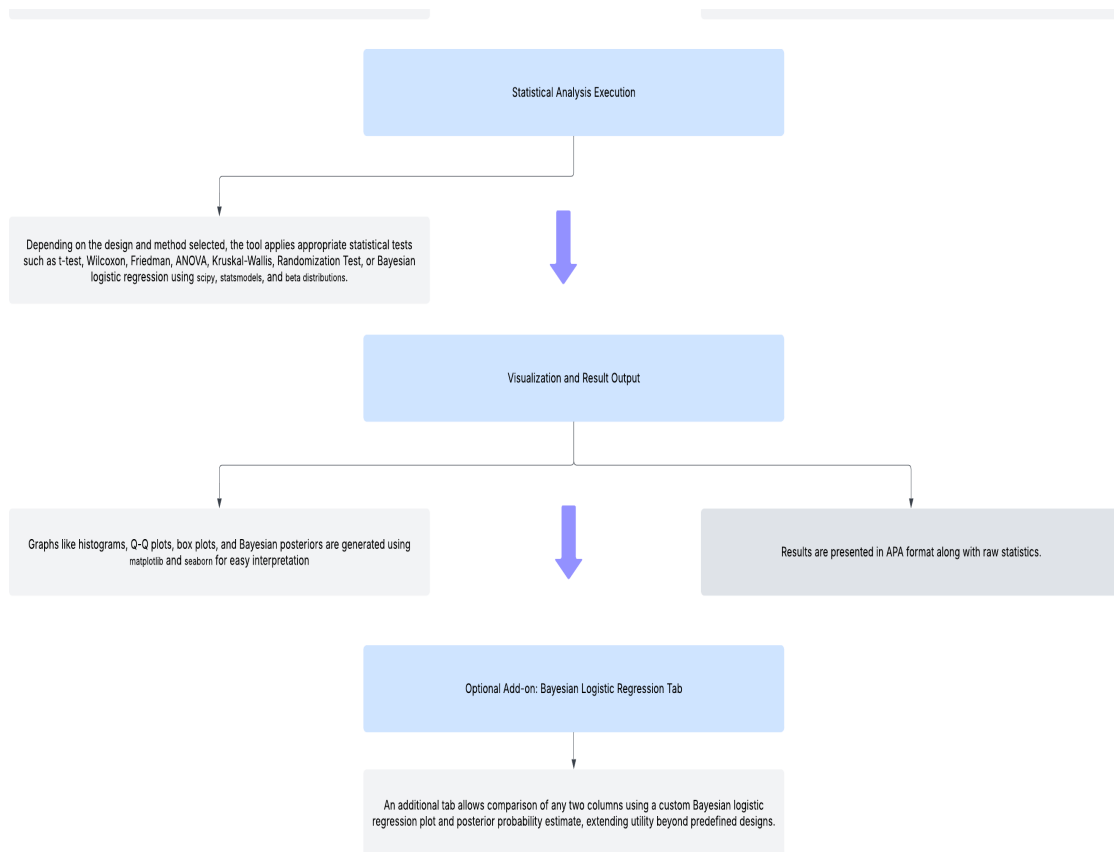
## **Conclusion**

By adopting an Agile framework with iterative sprints, responsive updates, and user-centered feedback cycles, the tool matured into a robust, modular, and highly usable application. Its flexible architecture allows for future expansion while ensuring it remains a valuable asset for behavior analysts and single-case researchers.

With the technical foundation established, the next section outlines the specific statistical methods embedded in the tool—each carefully selected and adapted to suit the needs of behavioral researchers working with Single-Case D.

### 3.3 : - Information Flow:





## 4. STATISTICAL METHODS

The AI-Powered SCD (Single-Case Design) Analysis Tool is designed to make complex behavioral data analysis more accessible and efficient for researchers, clinicians, and educators. Built using Streamlit, this user-friendly platform supports three major experimental designs ABA Reversal, Alternating Treatment, and Multiple Baseline allowing users to upload their datasets seamlessly.

With built-in automation, the tool helps users trim data (e.g., last 5 sessions per phase) and apply the most suitable statistical tests based on the dataset's structure and normality. Its intuitive sidebar and tabbed interface make navigating the analysis process straightforward, offering visualization tools like histograms, Q-Q plots, and box plots to explore data trends. Users can select from a rich set of statistical methods, including t-tests, Wilcoxon tests, Randomization Tests, Bayesian Analysis, and Friedman Tests.

The tool also supports APA-style reporting and effect size calculations (PND, PEM, IRD) to ensure rigorous, evidence-based interpretation. Additionally, it features a specialized Bayesian Logistic Add-on, allowing direct probability estimation of intervention success between two conditions. Whether you're a seasoned analyst or just beginning your statistical journey, this tool bridges the gap between advanced modeling and practical decision-making, helping you generate meaningful insights for publication and applied research.

```
13
14 #Importing Required Libraries:
15 import streamlit as st
16 import pandas as pd
17 import numpy as np
18 import matplotlib.pyplot as plt
19 import seaborn as sns
20 import scipy.stats as stats
21 from scipy.stats import ttest_rel, wilcoxon, friedmanchisquare, ttest_ind, mannwhitneyu, shapiro, beta as beta_dist, bootstrap
22 from statsmodels.regression.mixed_linear_model import MixedLM
23
24 # 📄 Streamlit Page Configuration and App Title
25 st.set_page_config(page_title="SCD Analysis Suite", layout="wide")
26 st.title("🧠 AI-Powered SCD Analysis Tool")
27
```

*Figure 1: Library Imports and Streamlit Configuration*

The foundation of the AI-powered Single-Case Design (SCD) Analysis Tool begins with setting up a robust environment for seamless data processing, statistical analysis, and visualization. It leverages essential Python libraries pandas and numpy for data manipulation, matplotlib and seaborn for clear and insightful plotting, and scipy.stats along with statsmodels to perform key statistical tests, including t-tests, Wilcoxon, Friedman, and mixed-effects models.

To ensure an intuitive experience, Streamlit configures the application with a well-structured layout, optimizing the interface for a wide view and displaying a clear main title. This setup

guarantees that users can effortlessly interact with the tool, exploring their data and conducting analyses through a smooth and user-friendly platform.

```
28 # 📄 User Instructions Section
29 st.markdown("""
30 ### 📁 How to Upload Your Dataset
31 1. Prepare your dataset in Microsoft Excel (.xlsx) format.
32 2. Ensure each phase or condition is in its own column (e.g., `Baseline`, `Intervention`, etc.).
33 3. Make sure each row represents a measurement point.
34 4. If you're analyzing subject-level data, include a `Subject` column.
35 5. Use the sidebar on the left to upload your Excel file.
36 6. After uploading, select the sheet and configure settings as needed.
37 """)
38
39 # 📁 File Upload and Tab Configuration
40 uploaded_file = st.sidebar.file_uploader("Upload Excel File", type=["xlsx"])
41 tabs = st.tabs(["Main Analysis", "Bayesian Logistic Add-on"])
42
```


**Figure 2:** *User Instructions and File Upload Configuration*

This section guides users through uploading their datasets and setting up the interface for file input and analysis. The `st.markdown()` function presents clear, formatted instructions in the Streamlit sidebar, ensuring users prepare their Excel files correctly. It specifies that data should be in .xlsx format, with separate columns for each phase (e.g., Baseline, Intervention) and rows representing individual measurement points. If subject-level data is included, a 'Subject' column is required for proper processing.

To facilitate dataset upload, the `st.sidebar.file_uploader()` widget allows users to import Excel files directly into the tool. Once uploaded, the interface adapts using `st.tabs()`, organizing the workflow into a "Main Analysis" tab and a "Bayesian Logistic Add-on" module. This setup ensures smooth navigation, balancing structured guidance with an intuitive user experience for streamlined data analysis.


```
#  Normality Check Function

def check_normality(data):
    stat, p_value = shapiro(data)
    return {"statistic": stat, "p_value": p_value, "is_normal": p_value > 0.05}

#  Normality Visualization Function
def visualize_normality(data, title):
    fig, axes = plt.subplots(1, 3, figsize=(18, 5))
    sns.histplot(data, bins=10, kde=True, ax=axes[0], color='skyblue')
    axes[0].set_title(f"Histogram of {title}")
    stats.probplot(data, dist="norm", plot=axes[1])
    axes[1].set_title(f"Q-Q Plot of {title}")
    sns.boxplot(data=data, ax=axes[2], color='lightblue')
    axes[2].set_title(f"Box Plot of {title}")
    plt.tight_layout()
    return fig
```

**Figure 3:** *Normality Check and Visualization*

This section introduces functions designed to assess and visualize the normality of a dataset—an essential step before applying parametric statistical tests. The `check_normality()` function utilizes the Shapiro-Wilk test from the `scipy.stats` library to statistically evaluate whether the data follows a normal distribution. It returns a dictionary containing the test statistic, p-value, and a Boolean flag (`is_normal`), which signals normality when the p-value exceeds 0.05. To complement this statistical assessment, the `visualize_normality()` function offers a graphical representation of the dataset's distribution, generating three key plots: a histogram with a kernel density estimate, a Q-Q plot comparing the sample to a normal distribution, and a box plot displaying data spread and potential outliers. These plots are arranged in a single row using matplotlib subplots, ensuring a thorough visual inspection. By integrating both statistical testing and graphical exploration, this approach enables users to make informed decisions about whether parametric tests are suitable for their data.

```
#  Results Display Function
def display_results(results, effect_size_label, effect_size_value ):
    st.write("### Raw Results")
    st.code(results["raw"])
    st.write("### APA-Style Results")
    st.write(results["apa"])
    st.write("### Interpretation and Feedback")
    st.write(results["feedback"])
    st.write("### Intervention Effect")
    st.write(results["effect"])
    st.write(f"### {effect_size_label} Value: {effect_size_value:.1f}%")
```

**Figure 4:** *Results Display Function*

The `display_results()` function plays a key role in presenting analysis outcomes clearly and efficiently within the Streamlit app. It takes three inputs: a results dictionary containing various outputs, an effect size label, and the corresponding effect size value. To enhance readability, the function structures the results into distinct sections. First, the raw results are displayed using `st.code()`, ensuring proper formatting and accessibility. Next, an APA-style summary is presented under a dedicated heading ideal for researchers needing publication-ready output. To further aid interpretation, the function includes a feedback section, helping users grasp the meaning and implications of their results. Additionally, a block for intervention effects summarizes the overall impact of the treatment or condition. Finally, the function neatly presents the chosen effect size label and its numeric value, formatted to one decimal place for precision. By organizing the results in a structured, intuitive way, this function ensures that all key analysis components are effectively communicated to the user, making interpretation seamless and actionable.

```

74 # 📌 Effect Size Computation Functions
75 def compute_pnd(baseline, intervention):
76     max_baseline = np.max(baseline)
77     non_overlapping = np.sum(intervention >= max_baseline)
78     return (non_overlapping / len(intervention)) * 100
79
80 def compute_pem(baseline, intervention):
81     median_baseline = np.median(baseline)
82     improved = np.sum(intervention > median_baseline)
83     return (improved / len(intervention)) * 100
84
85 def compute_ird(baseline, intervention):
86     baseline_fail = np.sum(baseline >= np.min(intervention))
87     intervention_success = np.sum(intervention > np.max(baseline))
88     n_baseline = len(baseline)
89     n_intervention = len(intervention)
90     if n_baseline + n_intervention == 0:
91         return np.nan
92     return ((intervention_success / n_intervention) - (baseline_fail / n_baseline)) * 100
93

```

**Figure 5:** *Effect Size Computation Functions*

This section presents three key functions used in Single-Case Design (SCD) research to measure the effectiveness of an intervention: PND, PEM, and IRD. Each function evaluates baseline and intervention phase data to quantify the degree of improvement. The `compute_pnd()` function calculates the Percentage of Non-overlapping Data (PND) by determining how many intervention data points exceed the highest baseline value, where a higher percentage signifies stronger intervention effects. The `compute_pem()` function computes the Percentage of Data Points Exceeding the Median (PEM) of the baseline, providing a more stable measure than PND by using the median instead of the maximum. Lastly, the `compute_ird()` function determines the Improvement Rate Difference (IRD) by considering both successes during intervention and failures during baseline. It calculates the difference between the intervention success rate and baseline failure rate, offering a more balanced evaluation of improvement. These functions express effect sizes in percentage form, assisting researchers in quantifying behavioral changes and understanding the impact of an intervention through both statistical analysis and visual representation.



```

93
94 # 📊 Descriptive Statistics Functions
95 def show_descriptive_statistics(data, label=""):
96     st.write(f"### Descriptive Statistics {f'for {label}' if label else ''}")
97     stats_df = data.agg(['mean', 'median', 'std', 'min', 'max']).to_frame().T
98     stats_df.columns = ['Mean', 'Median', 'Std Dev', 'Min', 'Max']
99     st.dataframe(stats_df)
100
101
102 def descriptive_stats(df):
103     stats = df.agg(['mean', 'median', 'std', 'min', 'max']).T
104     stats.columns = ['Mean', 'Median', 'SD', 'Min', 'Max']
105     return stats

```

**Figure 6:** *Descriptive Statistics Functions*

This section introduces functions designed to compute and present basic descriptive statistics, offering a concise summary of the dataset before further analysis. The `show_descriptive_statistics()` function formats and displays a table within the Streamlit app, presenting key metrics such as mean, median, standard deviation, minimum, and maximum values. If a label (e.g., group or condition name) is provided, it appears in the heading for added clarity. The statistics are computed using the `agg()` function and organized into a transposed DataFrame, ensuring a clean and structured presentation. Similarly, the `descriptive_stats()` function serves the same purpose but returns the statistics as a DataFrame rather than displaying them directly, making it particularly useful for back-end computations or deferred result presentations. These functions provide users with a clear understanding of data distribution and variability, enabling informed interpretation and ensuring the selection of appropriate statistical tests for further analysis.

```

# 📊 Bootstrap Confidence Interval with Visualization
def bootstrap_ci_with_plot(data):
    result = bootstrap((data,), np.mean, confidence_level=0.95, n_resamples=1000, random_state=0)
    fig, ax = plt.subplots()
    ax.hist(result.bootstrap_distribution, bins=30, alpha=0.7, color='purple')
    ax.axvline(result.confidence_interval.low, color='red', linestyle='--')
    ax.axvline(result.confidence_interval.high, color='green', linestyle='--')
    ax.set_title("Bootstrap Distribution of Effect Size")
    ax.set_xlabel("Mean Value")
    ax.set_ylabel("Frequency")
    return result.confidence_interval, fig

```

**Figure 7:** *Bootstrap Confidence Interval with Visualization*

The `bootstrap_ci_with_plot()` function provides an estimate of a confidence interval using the bootstrap method, a powerful resampling technique for evaluating the variability of a statistic—specifically the mean in this case. It executes 1000 bootstrap resamples at a 95% confidence level, utilizing the `bootstrap()` function from `scipy.stats` to generate a reliable interval estimate. To

enhance interpretability, the function produces a histogram displaying the bootstrap distribution of resampled means, visually representing their spread. Vertical dashed lines highlight the lower and upper confidence bounds, with red and green markers for clear distinction. Additionally, the plot is designed with a descriptive title and labeled axes, ensuring the output remains informative and easy to understand. Finally, the function returns both the computed confidence interval and the visual plot, allowing users to seamlessly incorporate these results into statistical summaries and research reports, making the findings more accessible and actionable.

```
# 📁 Main Analysis Tab Logic
with tabs[0]:
    if uploaded_file:
        sheet_names = pd.ExcelFile(uploaded_file).sheet_names
        selected_sheet = st.sidebar.selectbox("Select Sheet", sheet_names)
        data = pd.read_excel(uploaded_file, sheet_name=selected_sheet)
        st.write("### Data Preview")
        st.dataframe(data.head())

        analysis_type = st.sidebar.radio("Select Analysis Type", ["ABA Reversal", "Alternating Treatment", "Multiple Baseline"])

# 📁 ABA Reversal Design Analysis
if analysis_type == "ABA Reversal":
    phases = st.sidebar.multiselect("Select Phases", data.columns.tolist(), default=data.columns.tolist())
    if len(phases) >= 2:
        trimmed_data = [data[phase].dropna().tail(5) for phase in phases]
        min_len = min(len(p) for p in trimmed_data)
        trimmed_data = [p[:min_len] for p in trimmed_data]
        normality = [check_normality(p) for p in trimmed_data]
        baseline, intervention = trimmed_data[0], trimmed_data[1]
```

**Figure 8:** *Main Analysis Workflow and ABA Reversal Logic*

This section outlines the core logic for uploading and processing data within the Main Analysis tab, ensuring a streamlined and user-friendly experience. The process begins by checking if a file has been uploaded. If detected, the system reads all sheet names from the Excel document, allowing users to select a specific sheet via a sidebar dropdown and preview the dataset using `st.dataframe()`. Users can then choose from three analysis types ABA Reversal, Alternating Treatment, or Multiple Baseline using radio buttons to specify their study design. If ABA Reversal is selected, users must identify the relevant phases (such as Baseline, Intervention, and Return to Baseline) from the dataset columns, ensuring that at least two phases are chosen for valid analysis. For each selected phase, the data is trimmed to the last five entries, focusing on the most stable portion of the dataset. To maintain consistency, phase lengths are equalized by identifying the shortest phase. Additionally, normality checks are performed on each trimmed dataset to determine the appropriate statistical approach. Finally, the system separates baseline and intervention data for further statistical analysis. By organizing dataset processing in a structured way, this workflow ensures accuracy, consistency, and meaningful comparisons within ABA Reversal designs, providing a reliable framework for behavioral research.

```

# 📄 Effect Size Calculation Based on User Selection
if effect_size_method == "PND":
    effect = compute_pnd(baseline, intervention)
elif effect_size_method == "PEM":
    effect = compute_pem(baseline, intervention)
else:
    effect = compute_ird(baseline, intervention)

st.write("### Normality Checks")
for name, norm, values in zip(phases, normality, trimmed_data):
    st.pyplot(visualize_normality(values, name))
    st.write(f"{name} - Shapiro-Wilk Test: W = {norm['statistic']:.2f}, p = {norm['p_value']:.3f}")

if len(phases) == 2:
    valid_methods = ["Paired t-test", "Wilcoxon Signed-Rank Test", "Randomization Test", "Bayesian Analysis"]
else:
    valid_methods = ["Friedman Test", "Bayesian Analysis"]

method = st.sidebar.selectbox("Select Statistical Method", valid_methods)
baseline, intervention = trimmed_data[0], trimmed_data[1]

```

**Figure 9:** *Effect Size Calculation and Statistical Test Selection*

This section dynamically calculates the effect size and suggests appropriate statistical tests based on user input and dataset characteristics. It begins by determining the chosen effect size method—PND, PEM, or IRD—and then applies the respective function (`compute_pnd`, `compute_pem`, or `compute_ird`) to assess the intervention's impact by comparing baseline and intervention phase data. Once the effect size is computed, the system performs normality checks using the Shapiro-Wilk test and presents results through histograms, Q-Q plots, and box plots, making it easier for users to interpret the distribution of their data. The associated test statistics and p-values are displayed for transparency and deeper analysis. Based on the number of selected phases, the system filters available statistical tests accordingly. If two phases are chosen, users can select from Paired t-test, Wilcoxon Signed-Rank Test, Randomization Test, or Bayesian Analysis, all tailored for paired data comparisons. When more than two phases are present, the system narrows options to Friedman Test and Bayesian Analysis, which are more appropriate for multiple-condition evaluations. Users then finalize their statistical method through a dropdown menu, ensuring their selections align with the dataset structure and statistical assumptions, optimizing the accuracy and relevance of their analysis.

```

191
192     if method == "Paired t-test":
193         stat, p_value = ttest_rel(baseline, intervention)
194         df = len(baseline) - 1
195         results = {"raw": f"t-test Statistic: {stat:.3f}, p = {p_value:.3f}",
196                   "apa": f"t({len(baseline)-1}) = {stat:.2f}, p = {p_value:.3f}",
197                   "feedback": "Compares paired samples.",
198                   "effect": "Significant" if p_value < 0.05 else "Not Significant"}
199         display_results(results, effect_size_method, effect)
200
201     elif method == "Wilcoxon Signed-Rank Test":
202         stat, p_value = wilcoxon(baseline, intervention)
203         show_descriptive_statistics(baseline, "Baseline")
204         show_descriptive_statistics(intervention, "Intervention")
205         results = {"raw": f"Wilcoxon Statistic: {stat:.3f}, p = {p_value:.3f}",
206                   "apa": f"Wilcoxon W = {stat:.2f}, p = {p_value:.3f}",
207                   "feedback": "Non-parametric for paired samples.",
208                   "effect": "Significant" if p_value < 0.05 else "Not Significant"}
209         display_results(results, effect_size_method, effect)
210
211     elif method == "Randomization Test":
212         diffs = []
213         observed_diff = np.mean(intervention) - np.mean(baseline)
214         combined = np.concatenate([baseline, intervention])
215         show_descriptive_statistics(baseline, "Baseline")
216         show_descriptive_statistics(intervention, "Intervention")
217         for _ in range(1000):
218             np.random.shuffle(combined)
219             new_baseline = combined[:len(baseline)]
220             new_intervention = combined[len(baseline):]
221             diffs.append(np.mean(new_intervention) - np.mean(new_baseline))
222         p_value = np.mean(np.abs(diffs) >= np.abs(observed_diff))
223         results = {"raw": f"Observed Diff: {observed_diff:.3f}, p = {p_value:.3f}",
224                   "apa": f"Randomization Diff = {observed_diff:.2f}, p = {p_value:.3f}",
225                   "feedback": "Label-shuffling nonparametric test.",
226                   "effect": "Significant" if p_value < 0.05 else "Not Significant"}
227         display_results(results, effect_size_method, effect)
228

```

**Figure 10:** *Statistical Test Execution and Result Reporting*

This section conducts statistical tests based on the user's selected method—Paired t-test, Wilcoxon Signed-Rank Test, or Randomization Test—and presents results using the previously defined `display_results()` function. The Paired t-test computes the t-statistic and p-value, comparing means of two related groups, and formats results in APA style, including feedback on statistical significance. If normality cannot be assumed, the Wilcoxon Signed-Rank Test, a non-parametric alternative, is used for paired samples, displaying descriptive statistics for each phase and interpreting significance. The Randomization Test takes a resampling-based approach, shuffling data labels 1000 times to generate a mean difference distribution, comparing the observed difference to this reference to compute a p-value. Each method provides a structured output, including raw test statistics, APA-style summaries, interpretations, and significance status, ensuring that users receive clear and actionable feedback regarding their analysis.



```

elif method == "Bayesian Analysis":
    posterior_mean, fig = bayesian_logistic_regression(baseline, intervention)
    st.pyplot(fig)
    show_descriptive_statistics(baseline, "Baseline")
    show_descriptive_statistics(intervention, "Intervention")
    results = {"raw": f"Posterior Mean: {posterior_mean:.3f}",
               "apa": f"Bayesian Posterior = {posterior_mean:.3f}",
               "feedback": "Estimates probability of improvement.",
               "effect": f"{posterior_mean:.2%} probability of effectiveness."}
    display_results(results, effect_size_method, effect)

elif method == "Friedman Test":
    stat, p_value = friedmanchisquare(*trimmed_data)
    show_descriptive_statistics(baseline, "Baseline")
    show_descriptive_statistics(intervention, "Intervention")
    results = {"raw": f"Statistic = {stat:.3f}, p = {p_value:.3f}",
               "apa": f"Friedman X^2 = {stat:.2f}, p = {p_value:.3f}",
               "feedback": "Compares >2 related phases.",
               "effect": "Significant" if p_value < 0.05 else "Not Significant"}
    display_results(results, effect_size_method, effect)

```

**Figure 11:** *Bayesian and Friedman Test Implementation*

This section presents two additional statistical methods Bayesian Analysis and the Friedman Test providing flexible options based on user selection. In Bayesian Analysis, the `bayesian_logistic_regression()` function estimates the probability of improvement between baseline and intervention phases. A posterior mean is computed and visually plotted, offering a probabilistic interpretation of intervention effectiveness. The results are formatted in APA style, with interpretation feedback clarifying the likelihood of intervention success. The Friedman Test, a non-parametric method, is used for comparing more than two related samples or phases. It calculates a test statistic and p-value to evaluate whether significant differences exist across conditions. Descriptive statistics for each phase are displayed, and the results are categorized as significant or not based on the p-value. Both methods provide statistical outputs, APA-formatted text, interpretations, and effect summaries, ensuring that users can effectively analyze diverse datasets while maintaining clarity and rigor in their findings.

```

# Alternating Treatment Design Analysis
if analysis_type == "Alternating Treatment":
    condition_col = st.sidebar.selectbox("Select Condition Column", data.columns)
    value_col = st.sidebar.selectbox("Select Value Column", [col for col in data.columns if col != condition_col])
    at_data = data[[condition_col, value_col]].dropna()
    groups = at_data[condition_col].unique()
    g1 = at_data[at_data[condition_col] == groups[0]][value_col]
    g2 = at_data[at_data[condition_col] == groups[1]][value_col]

    if effect_size_method == "PND":
        effect = compute_pnd(g1, g2)
    elif effect_size_method == "PEM":
        effect = compute_pem(g1, g2)
    else:
        effect = compute_ird(g1, g2)

# Show Descriptive Statistics
for g in groups:
    show_descriptive_statistics(at_data[at_data[condition_col] == g][value_col], str(g))

for g in groups:
    group_data = at_data[at_data[condition_col] == g][value_col]
    fig = visualize_normality(group_data, str(g))
    st.pyplot(fig)

# Choose method based on group count
if len(groups) == 2:
    at_methods = ["Independent t-test", "Mann-Whitney U Test", "Randomization Test", "Bayesian Analysis"]
else:
    at_methods = ["ANOVA", "Kruskal-Wallis Test", "Randomization Test"]

at_method = st.sidebar.selectbox("Select Statistical Method", at_methods)
group_values = [at_data[at_data[condition_col] == g][value_col].values for g in groups]

```

**Figure 12:** *Alternating Treatment Design Analysis*

This block outlines the analysis flow when Alternating Treatment is selected, ensuring the system adapts to varying data structures in treatment comparison designs. Users first select a condition column (e.g., treatment type) and a value column (e.g., outcome measure) from the uploaded dataset, allowing the tool to identify and store two treatment groups for comparison. The effect size is then computed using the chosen method PND, PEM, or IRD to quantify differences between the two groups. To support assumption checks and provide exploratory insights, the code generates descriptive statistics and normality plots for each group, offering a comprehensive view of the data distribution. Based on the number of groups, the system suggests appropriate statistical tests: for two groups, users can choose between the Independent t-test, Mann-Whitney U Test, Randomization Test, or Bayesian Analysis. If there are more than two groups, the options shift to ANOVA, Kruskal-Wallis Test, or Randomization Test, ensuring the selected test aligns with the dataset structure and statistical requirements.

```

if at_method == "Kruskal-Wallis Test":
    stat, p_value = stats.kruskal(*group_values)
    results = {
        "raw": f"Statistic: {stat:.3f}, P-value: {p_value:.3f}",
        "apa": f"Kruskal-Wallis H = {stat:.2f}, p = {p_value:.3f}",
        "feedback": "Non-parametric test comparing multiple groups.",
        "effect": "Significant difference." if p_value < 0.05 else "No significant difference."
    }
    display_results(results, effect_size_method, effect)

elif at_method == "ANOVA":
    stat, p_value = stats.f_oneway(*group_values)
    results = {
        "raw": f"F-statistic: {stat:.3f}, P-value: {p_value:.3f}",
        "apa": f"ANOVA F = {stat:.2f}, p = {p_value:.3f}",
        "feedback": "Parametric ANOVA comparing multiple groups.",
        "effect": "Significant difference." if p_value < 0.05 else "No significant difference."
    }
    display_results(results, effect_size_method, effect)

```

**Figure 13: Kruskal-Wallis and ANOVA Analysis**

This section conducts group comparison tests when multiple groups are analyzed within the Alternating Treatment design, ensuring a structured approach to evaluating statistical differences. If the Kruskal-Wallis Test is selected, the system applies this non-parametric method to calculate the test statistic and p-value, determining whether statistically significant differences exist among groups without assuming normality. Alternatively, if ANOVA is chosen, the code executes a parametric one-way analysis of variance, comparing group means while assuming normality and equal variances. Both methods present results in raw and APA styles, providing interpretive feedback to help users understand statistical significance and differences between groups. The results are then formatted and displayed using the `display_results()` function, ensuring clear presentation and structured output for data-driven decision-making.

This final section completes the Alternating Treatment analysis logic by introducing two advanced statistical methods Randomization Test and Bayesian Analysis providing robust alternatives when parametric test assumptions may not be met. In the Randomization Test, the system first verifies that exactly two groups are present, as the method applies solely to pairwise comparisons. It calculates the observed mean difference between groups and then performs 1,000 permutations, randomly shuffling group labels. Each shuffled iteration computes a new difference, forming a distribution used to determine a p-value, indicating whether the observed effect is statistically significant. In Bayesian Analysis, the system applies a logistic regression model to estimate the posterior probability that one group outperforms the other, generating a visual plot of the posterior distribution for clarity. The analysis output includes the posterior mean, APA-style summary, interpretation, and an optional PND effect size calculation, ensuring structured and comprehensive reporting. Together, these methods extend the tool's statistical flexibility, accommodating diverse data structures and analytical needs within treatment comparisons.

```

elif at_method == "Randomization Test":
    if len(groups) != 2:
        st.warning("Randomization Test currently supports only two groups.")
    else:
        g1 = at_data[at_data[condition_col] == groups[0]][value_col].values
        g2 = at_data[at_data[condition_col] == groups[1]][value_col].values
        observed_diff = np.mean(g2) - np.mean(g1)
        combined = np.concatenate([g1, g2])
        diffs = []
        for _ in range(1000):
            np.random.shuffle(combined)
            new_g1 = combined[:len(g1)]
            new_g2 = combined[len(g1):]
            diffs.append(np.mean(new_g2) - np.mean(new_g1))
        p_value = np.mean(np.abs(diffs) >= np.abs(observed_diff))
        results = {
            "raw": f"Observed Diff: {observed_diff:.3f}, P-value: {p_value:.3f}",
            "apa": f"Randomization Test: diff = {observed_diff:.2f}, p = {p_value:.3f}",
            "feedback": "Randomization test between two groups.",
            "effect": "Significant difference." if p_value < 0.05 else "No significant difference."
        }
        display_results(results, effect_size_method, effect)

elif at_method == "Bayesian Analysis":
    g1 = group_values[0]
    g2 = group_values[1]
    posterior_mean, fig = bayesian_logistic_regression(pd.Series(g1), pd.Series(g2))
    st.markdown("### Bayesian Logistic Regression")
    st.pyplot(fig)
    results = {
        "raw": f"Posterior Mean: {posterior_mean:.3f}",
        "apa": f"Bayesian Estimate = {posterior_mean:.3f}",
        "feedback": "Estimates probability of one group outperforming the other.",
        "effect": f"Bayesian analysis suggests {posterior_mean:.2%} probability of effectiveness.",
        "pnd": compute_pnd(g1, g2)
    }
    display_results(results, effect_size_method, effect)

```

**Figure 14:** *Alternating Treatment: Randomization and Bayesian Tests*

This section manages the analysis process for the Multiple Baseline Design, a research approach used to track behavioral changes across subjects or settings over time. Users begin by selecting subject, phase, and measurement columns from their dataset, which the system then filters, extracting the last five data points from each phase before reshaping the data into a pivoted table for comparison. If at least two distinct phases (e.g., baseline and intervention) are detected, the system splits the data accordingly and displays descriptive statistics to facilitate interpretation. If fewer than two phases exist, a warning alerts users about insufficient data for meaningful analysis. Depending on the selected effect size method PND, PEM, or IRD the tool calculates the



appropriate effect size to quantify differences between phases. This structured workflow ensures valid, phase-based comparisons in group- or subject-level analyses, supporting rigorous evaluations in multiple baseline research designs.

```
# Multiple Baseline Design
if analysis_type == "Multiple Baseline":
    subject_col = st.sidebar.selectbox("Select Subject Column", data.columns)
    phase_col = st.sidebar.selectbox("Select Phase Column", [c for c in data.columns if c != subject_col])
    value_col = st.sidebar.selectbox("Select Measurement Column", [c for c in data.columns if c not in [subject_col, phase_col]])
    filtered = data[[subject_col, phase_col, value_col]].dropna()
    last5 = filtered.groupby([subject_col, phase_col]).tail(5)
    pivoted = last5.pivot_table(index=subject_col, columns=phase_col, values=value_col)

    if pivoted.shape[1] >= 2:
        baseline = pivoted.iloc[:, 0].dropna()
        intervention = pivoted.iloc[:, 1].dropna()

        show_descriptive_statistics(baseline, "Baseline")
        show_descriptive_statistics(intervention, "Intervention")
    else:
        st.warning("Not enough phases to compute PND. Please check the uploaded data.")

    if effect_size_method == "PND":
        effect = compute_pnd(baseline, intervention)
    elif effect_size_method == "PEM":
        effect = compute_pem(baseline, intervention)
    else:
        effect = compute_ird(baseline, intervention)
```

**Figure 15: Multiple Baseline Design Analysis**

```
# All modules and UI run without missing analysis logic now.
# --- Bayesian Add-on Tab ---
with tabs[1]:
    st.header("Add-on: Bayesian Logistic Regression")
    if uploaded_file:
        sheet_names = pd.ExcelFile(uploaded_file).sheet_names
        selected_sheet = st.selectbox("Select sheet for Bayesian Add-on", sheet_names)
        data = pd.read_excel(uploaded_file, sheet_name=selected_sheet)

        if len(data.columns) >= 2:
            col1 = st.selectbox("Select Baseline Column", data.columns)
            col2 = st.selectbox("Select Comparison Column", [col for col in data.columns if col != col1])

            values1 = data[col1].dropna().reset_index(drop=True)
            values2 = data[col2].dropna().reset_index(drop=True)

            if len(values1) > 1 and len(values2) > 1:
                posterior_mean, fig = bayesian_logistic_regression(values1, values2)
                st.pyplot(fig)
                st.write(f"***Posterior Mean Probability that {col2} > {col1}**: {posterior_mean:.3f}")
                st.markdown("*** Interpretation:*** A higher posterior mean (closer to 1) suggests strong evidence that the second column outperforms the first.")
            else:
                st.warning("Not enough data in one of the columns.")
        else:
            st.warning("Please upload a sheet with at least 2 numeric columns.")
```

**Figure 16: Bayesian Add-on Tab: Logistic Regression Comparison**

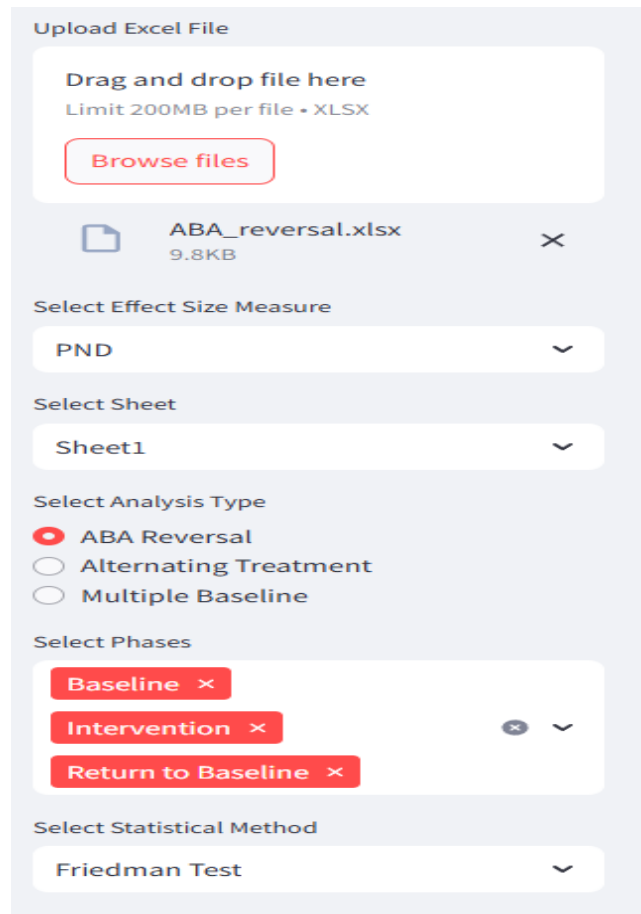
This section manages the analysis process for the Multiple Baseline Design, a research approach used to track behavioral changes across subjects or settings over time. Users begin by selecting subject, phase, and measurement columns from their dataset, which the system then filters,

extracting the last five data points from each phase before reshaping the data into a pivoted table for comparison. If at least two distinct phases (e.g., baseline and intervention) are detected, the system splits the data accordingly and displays descriptive statistics to facilitate interpretation. If fewer than two phases exist, a warning alerts users about insufficient data for meaningful analysis. Depending on the selected effect size method PND, PEM, or IRD the tool calculates the appropriate effect size to quantify differences between phases. This structured workflow ensures valid, phase-based comparisons in group- or subject-level analyses, supporting rigorous evaluations in multiple baseline research designs.

To ensure these complex statistical methods are accessible and actionable, the project emphasizes an intuitive user interface, which is detailed in the following section.

## **4.1 USER INTERFACE :**

The user interface of the AI-Powered SCD Analysis Tool plays a vital role in making advanced statistical analysis both accessible and user-friendly. It guides users step by step from uploading Excel files to selecting analysis types and statistical tests through an intuitive design featuring dropdown menus and clear instructions. To enhance interpretation, the UI provides visual outputs such as data previews, normality plots, and APA-style summaries, ensuring users can easily grasp their results. By streamlining interactions with complex methods, the tool allows for accurate, efficient, and interpretable analysis, empowering researchers and practitioners to perform sophisticated assessments without requiring coding knowledge.



The sidebar is titled "Upload Excel File" and contains the following sections:

- Drag and drop file here**  
Limit 200MB per file • XLSX  
A red-outlined button labeled "Browse files" is located below the instructions.
- File Upload:** A file named "ABA\_reversal.xlsx" (9.8KB) is shown with a document icon and a close button (X).
- Select Effect Size Measure:** A dropdown menu with "PND" selected.
- Select Sheet:** A dropdown menu with "Sheet1" selected.
- Select Analysis Type:** Three radio button options: "ABA Reversal" (selected), "Alternating Treatment", and "Multiple Baseline".
- Select Phases:** A container with three red buttons: "Baseline", "Intervention", and "Return to Baseline", each with a close button (X). A plus icon and a dropdown arrow are on the right.
- Select Statistical Method:** A dropdown menu with "Friedman Test" selected.

## AI-Powered SCD Analysis Tool

### How to Upload Your Dataset

1. Prepare your dataset in **Microsoft Excel** (.xlsx) format.
2. Ensure each phase or condition is in its **own column** (e.g., **Baseline**, **Intervention**, etc.).
3. Make sure each row represents a **measurement point**.
4. If you're analyzing subject-level data, include a **Subject** column.
5. Use the sidebar on the left to **upload your Excel file**.
6. After uploading, select the sheet and configure settings as needed.

**Figure 18:** *User Interface Overview and Dataset Configuration*

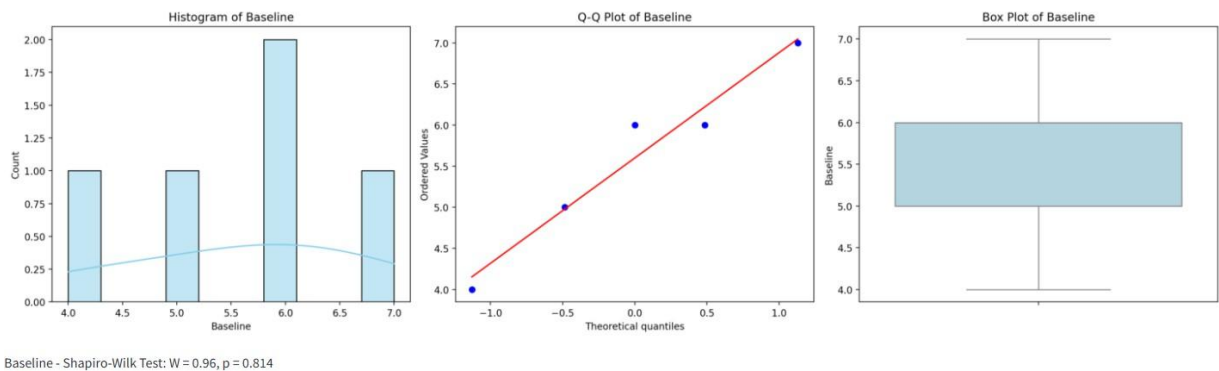
This visual section showcases the user interface of the AI-Powered SCD Analysis Tool, guiding users through the initial steps of uploading and configuring their dataset for analysis. The first image displays the instructional banner within the app, outlining how to properly format the Excel dataset, ensuring that each phase (e.g., Baseline, Intervention) is organized into separate columns, with each row representing a measurement point. It also advises users to include a "Subject" column for subject-level data when applicable. The second image highlights the sidebar panel, where users upload an Excel file, select the desired sheet, and configure settings such as the effect

size measure (e.g., PND), analysis type (e.g., ABA Reversal), relevant phases, and the statistical method (e.g., Friedman Test). This intuitive interface ensures a streamlined setup experience, allowing users to customize their analysis with precision while maintaining clarity and usability in their workflow.

Data Preview

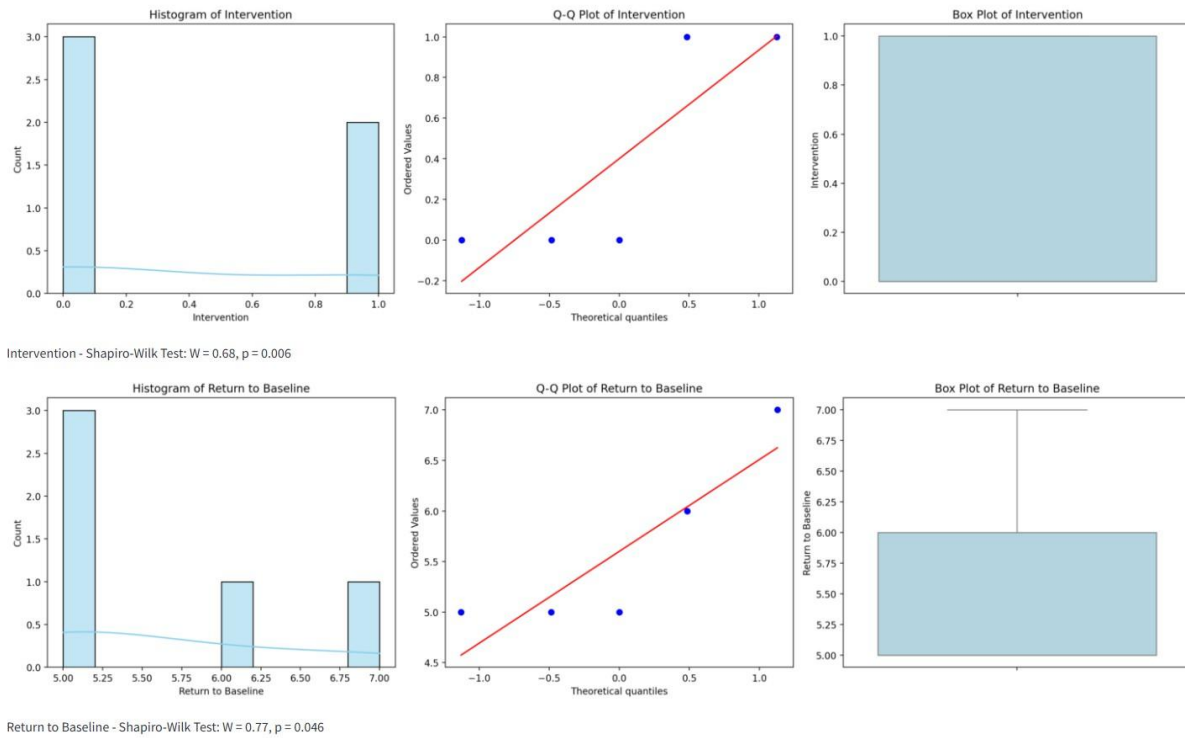
	Baseline	Intervention	Return to Baseline
0	6	5	3
1	7	3	5
2	4	5	6
3	6	2	5
4	5	4	5

Normality Checks



**Figure 19:** *Data Preview and Normality Check Visualization*

This section provides a preview of the uploaded dataset and conducts normality checks for the Baseline phase, ensuring the dataset is structured for valid statistical analysis. The table confirms that each phase Baseline, Intervention, and Return to Baseline contains appropriate measurement points, validating the integrity of the data. Below, three visual assessments examine the distribution of baseline values: a histogram illustrates the spread of the data, a Q-Q plot evaluates alignment with a normal distribution, and a box plot highlights central tendency and variability. The Shapiro-Wilk test result ( $W = 0.96$ ,  $p = 0.814$ ) suggests that the data is normally distributed, supporting the application of parametric tests for further analysis.



**Figure 20:** *Normality Checks for Intervention and Return to Baseline Phases*

This section evaluates the normality of the Intervention and Return to Baseline phases using histograms, Q-Q plots, and box plots, providing a visual and statistical assessment of their distributions. The Shapiro-Wilk test results indicate non-normal distributions in both phases: Intervention phase ( $W = 0.68, p = 0.006$ ) and Return to Baseline ( $W = 0.77, p = 0.046$ ), both falling below the 0.05 threshold, signaling significant deviation from normality. These findings suggest that parametric tests may not be appropriate, reinforcing the need for non-parametric statistical methods in analyzing these phases. The combined graphical and statistical insights ensure researchers can make informed decisions when selecting analytical approaches for further investigation.

Descriptive Statistics for Baseline

	Mean	Median	Std Dev	Min	Max
Baseline	5.6	6	1.1402	4	7

Descriptive Statistics for Intervention

	Mean	Median	Std Dev	Min	Max
Intervention	0.4	0	0.5477	0	1

Raw Results

Statistic = 7.600, p = 0.022

APA-Style Results

Friedman  $\chi^2 = 7.60$ ,  $p = 0.022$

Interpretation and Feedback

Compares >2 related phases.

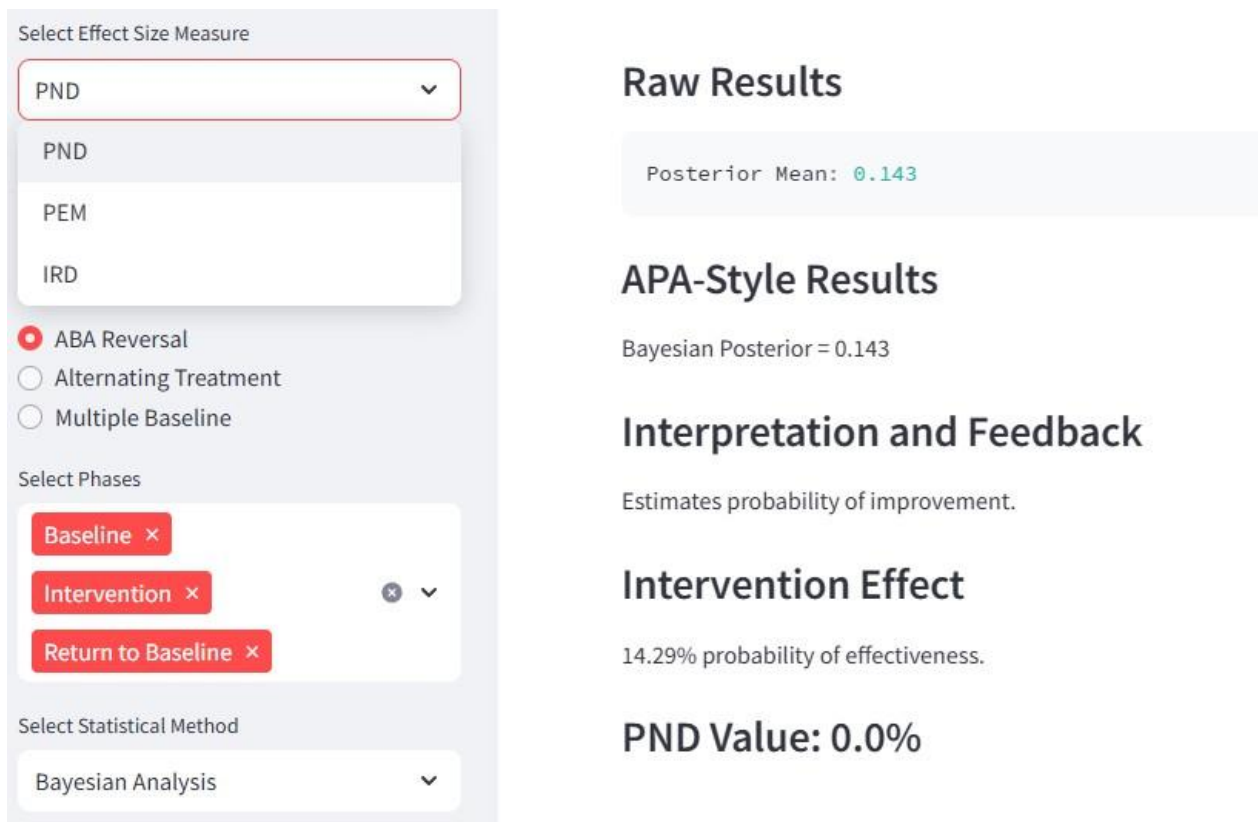
Intervention Effect

Significant

PND Value: 0.0%

**Figure 21:** Statistical Results and Interpretation: Baseline vs. Intervention

This section provides a summary of the statistical comparison between the Baseline and Intervention phases, revealing a notable contrast in descriptive statistics. The baseline phase has a significantly higher mean (5.6) compared to the intervention phase (0.4), suggesting a considerable shift in observed values. To assess these differences, a Friedman test was conducted due to the presence of more than two related phases, yielding a statistically significant result ( $\chi^2 = 7.60$ ,  $p = 0.022$ ), indicating that the observed differences are unlikely due to chance. The accompanying APA-style result reinforces this conclusion, stating that the intervention effect is significant. However, the PND value is 0.0%, signifying that none of the intervention data points exceed the maximum baseline value, implying no improvement by that metric. This contrast underscores the importance of employing multiple analysis methods to gain a comprehensive understanding of treatment impact, ensuring a nuanced interpretation of intervention effectiveness.

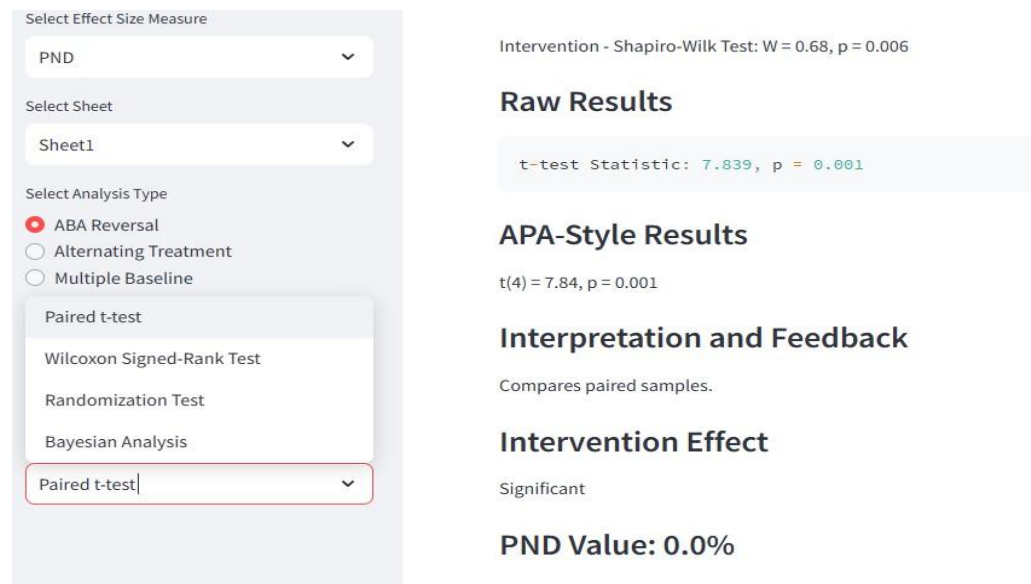


**Figure 22:** *Bayesian Analysis Outcome with PND Comparison*

This section provides a summary of the statistical comparison between the Baseline and Intervention phases, revealing a notable contrast in descriptive statistics. The baseline phase has a significantly higher mean (5.6) compared to the intervention phase (0.4), suggesting a considerable shift in observed values. To assess these differences, a Friedman test was conducted due to the presence of more than two related phases, yielding a statistically significant result ( $\chi^2 = 7.60$ ,  $p = 0.022$ ) indicating that the observed differences are unlikely due to chance. The accompanying APA-style result reinforces this conclusion, stating that the intervention effect is significant. However, the PND value is 0.0%, signifying that none of the intervention data points exceed the maximum baseline value, implying no improvement by that metric. This contrast underscores the importance of employing multiple analysis methods to gain a comprehensive understanding of treatment impact, ensuring a nuanced interpretation of intervention effectiveness.

This section outlines three key effect size measures—PND, PEM, and IRD—that help assess intervention effectiveness in Single-Case Design (SCD) studies. PND (Percentage of Non-overlapping Data) focuses on the proportion of intervention data points that surpass the highest baseline value, signaling improvement. PEM (Percentage of Data Points Exceeding the Median) provides a more robust measure by comparing intervention scores to the baseline median, reducing sensitivity to extreme values. IRD (Improvement Rate Difference) takes a comparative approach, factoring in both the success rate during intervention and failure rate during baseline to determine

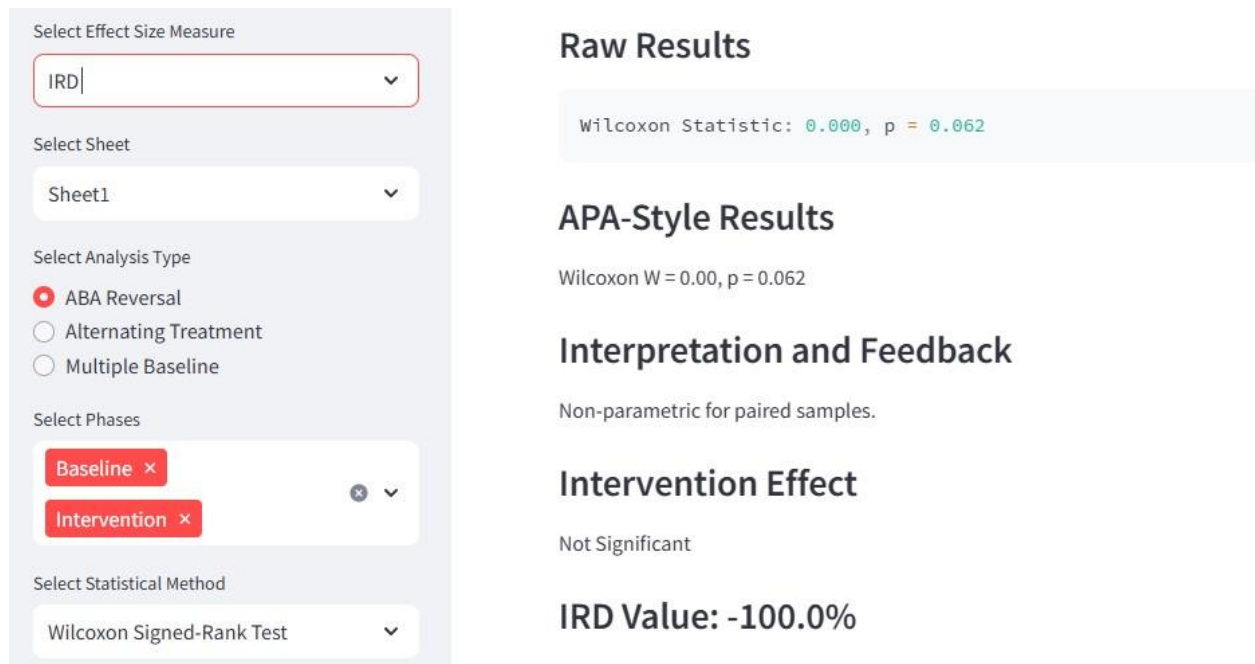
the degree of change. These effect size methods allow for a flexible and data-driven evaluation, ensuring researchers can interpret behavioral progress through multiple analytical perspectives.



**Figure 23:** *Paired t-test Results and Effect Size Summary*

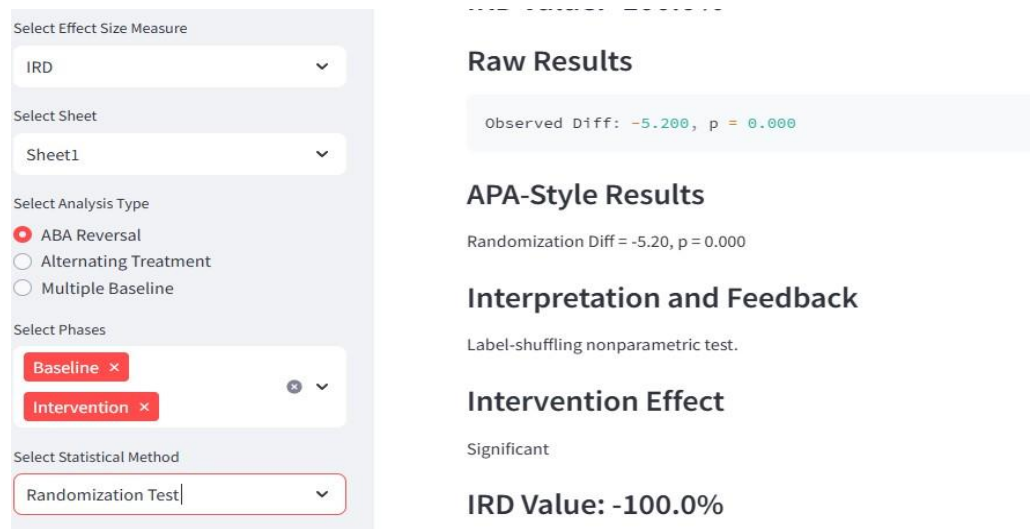
This section presents the results of a Paired t-test conducted on the selected ABA Reversal data, demonstrating a highly significant intervention effect. The statistical output ( $t = 7.84$ ,  $p = 0.001$ ) indicates a strong difference between the Baseline and Intervention phases, suggesting that the intervention had a meaningful impact. The results are formatted in APA style, providing a clear summary for academic reporting. However, despite this statistical significance, the PND (Percentage of Non-overlapping Data) value remains at 0.0%, indicating that none of the intervention scores exceeded the highest baseline score. This contrast underscores the importance of using multiple effect size measures such as PEM (Percentage of Data Points Exceeding the Median) and IRD (Improvement Rate Difference) to gain a more comprehensive understanding of treatment impact. Since PND alone may underrepresent positive change due to overlapping score ranges, incorporating alternative effect size measures allows for a more nuanced interpretation of intervention effectiveness.





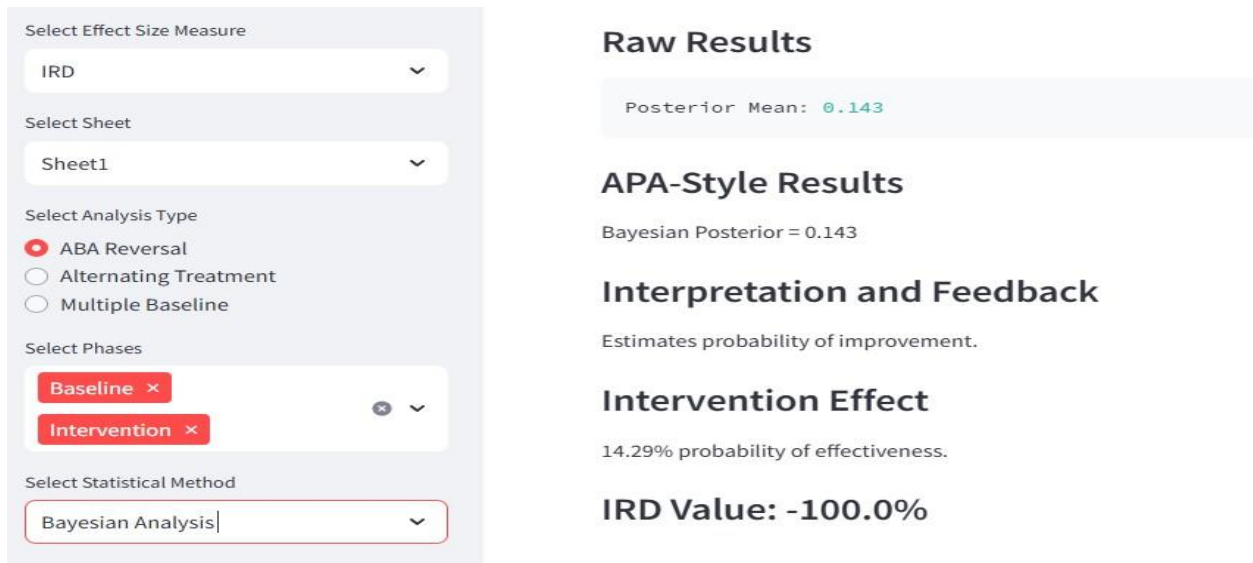
**Figure 24:** *Wilcoxon Test Interpretation*

This analysis applied the Wilcoxon Signed-Rank Test to compare the Baseline and Intervention phases within an ABA Reversal design, evaluating whether the intervention led to meaningful changes. The test result ( $W = 0.00$ ,  $p = 0.062$ ) indicates a non-significant effect, as the p-value exceeds the 0.05 threshold, suggesting that the observed differences may not be statistically meaningful. Additionally, the selected effect size metric, IRD (Improvement Rate Difference), yielded a value of -100.0%, signaling a complete lack of improvement with more failures occurring during the intervention phase compared to baseline. This negative IRD value aligns with the Wilcoxon test results, reinforcing the conclusion that the intervention may not have been effective and, in this instance, possibly detrimental. These findings underscore the importance of evaluating multiple statistical measures to gain a comprehensive understanding of treatment effectiveness and potential unintended consequences.



**Figure 25:** *Randomization Test with IRD: Contrasting Significance and Effect Size*

This section presents the results of a Randomization Test combined with the IRD (Improvement Rate Difference) effect size, offering a comprehensive evaluation of the intervention's impact within the ABA Reversal design. The Randomization Test reveals a statistically significant intervention effect (observed difference = -5.20,  $p = 0.000$ ), indicating that the observed difference between Baseline and Intervention phases is highly unlikely to be due to chance. However, the IRD value is -100.0%, meaning the intervention consistently underperformed compared to baseline measures. This contrast is critical, while the randomization test confirms significance, the direction and magnitude of IRD suggest a strong negative intervention effect. Such combined reporting ensures that both statistical significance and practical interpretation are taken into account, highlighting the importance of using multiple effect size measures to fully assess the intervention's effectiveness.



**Figure 26:** *Bayesian Analysis with IRD: Low Probability of Effectiveness*

This section presents the results of a Bayesian Analysis alongside the IRD (Improvement Rate Difference) to evaluate the effectiveness of the intervention compared to baseline performance. The posterior mean of 0.143 suggests only a 14.29% probability that the intervention outperformed the baseline, indicating weak evidence for improvement. Supporting this, the IRD value is -100.0%, meaning that none of the intervention scores exceeded the baseline, signaling a complete lack of improvement. Together, these findings provide a compelling conclusion: both the probabilistic approach and the overlap-based metric consistently indicate that the intervention was ineffective or even counterproductive in this case. These results highlight the importance of examining multiple statistical perspectives to gain a more comprehensive understanding of treatment effects.

Select Effect Size Measure

IRD

Select Sheet

Sheet1

Select Analysis Type

☐ ABA Reversal

☒ Alternating Treatment

☐ Multiple Baseline

Select Condition Column

Baseline

Select Value Column

Intervention

Select Statistical Method

ANOVA

Raw Results

F-statistic: 0.170, P-value: 0.906

APA-Style Results

ANOVA  $F = 0.17$ ,  $p = 0.906$

Interpretation and Feedback

Parametric ANOVA comparing multiple groups.

Intervention Effect

No significant difference.

IRD Value: -50.0%

**Figure 27:** *Alternating Treatment Analysis: ANOVA with IRD Evaluation*

This section presents the outcome of an Alternating Treatment Design analyzed using ANOVA and the IRD (Improvement Rate Difference) effect size. The ANOVA results ( $F = 0.17$ ,  $p = 0.906$ ) indicate no statistically significant difference between the groups, confirming that the intervention and baseline conditions performed similarly overall.

The IRD value of -50.0% reflects a moderate decline in performance during the intervention phase compared to baseline, reinforcing the absence of a positive intervention effect. This combination of non-significant ANOVA results and negative IRD suggests the intervention did not produce measurable improvement and may have even led to reduced outcomes.

Select Effect Size Measure

IRD

Select Sheet

Sheet1

Select Analysis Type

☐ ABA Reversal

☒ Alternating Treatment

☐ Multiple Baseline

Select Condition Column

Baseline

Select Value Column

Intervention

Select Statistical Method

Kruskal-Wallis Test

Raw Results

Statistic: 1.421, P-value: 0.701

APA-Style Results

Kruskal-Wallis  $H = 1.42$ ,  $p = 0.701$

Interpretation and Feedback

Non-parametric test comparing multiple groups.

Intervention Effect

No significant difference.

IRD Value: -50.0%

**Figure 28:** *Kruskal-Wallis Test with IRD in Alternating Treatment Design*

This result presents the application of the Kruskal-Wallis Test, a non-parametric method used to compare multiple groups within an Alternating Treatment Design. The test statistic ( $H = 1.42$ ,  $p = 0.701$ ) indicates that no statistically significant difference was found between the conditions, suggesting that the intervention did not produce meaningful variation across groups. Additionally, the IRD (Improvement Rate Difference) value is -50.0%, meaning the intervention group underperformed relative to the baseline group in half the comparisons. Taken together, the non-significant p-value and the negative IRD suggest that the intervention failed to provide measurable benefits and may have even resulted in reduced effectiveness. This highlights the importance of considering multiple analytical perspectives to fully assess treatment impact and avoid misleading conclusions based on a single metric.

Select Effect Size Measure

IRD

Select Sheet

Sheet1

Select Analysis Type

ABA Reversal

Alternating Treatment

Multiple Baseline

Select Subject Column

Baseline

Select Phase Column

Intervention

Select Measurement Column

Return to Baseline

Select Statistical Method

Paired t-test

Raw Results

Statistic: nan, P-value: nan

APA-Style Results

t(0) = nan, p = nan

Interpretation and Feedback

Paired t-test compares two related samples.

Intervention Effect

No significant effect.

IRD Value: -100.0%

**Figure 29:** *Multiple Baseline Design: Inconclusive t-test with IRD Evaluation*

This section presents the results of a Paired t-test conducted within a Multiple Baseline Design, using selected subject, phase, and measurement columns to evaluate intervention effectiveness. The statistical output returned NaN (Not a Number) values for both the test statistic and p-value, indicating that the test could not be computed, likely due to insufficient or identical data points across groups. Despite this limitation, the IRD (Improvement Rate Difference) value is -100.0%, signifying that all intervention data points performed worse than the baseline, suggesting a completely negative treatment effect. This contrast highlights an important scenario where the statistical test fails due to data constraints, yet the effect size measure still provides meaningful insight, reinforcing the importance of considering multiple analytical approaches when interpreting treatment effectiveness.

Select Effect Size Measure

IRD

Select Sheet

Sheet1

Select Analysis Type

☐ ABA Reversal

☐ Alternating Treatment

☒ Multiple Baseline

Select Subject Column

Baseline

Select Phase Column

Intervention

Select Measurement Column

Return to Baseline

Statistical Options

Select Statistical Method

Randomization Test

Raw Results

Observed Diff: 0.000, P-value: 1.000

APA-Style Results

Randomization Test: diff = 0.00, p = 1.000

Interpretation and Feedback

Randomization test based on shuffled labels.

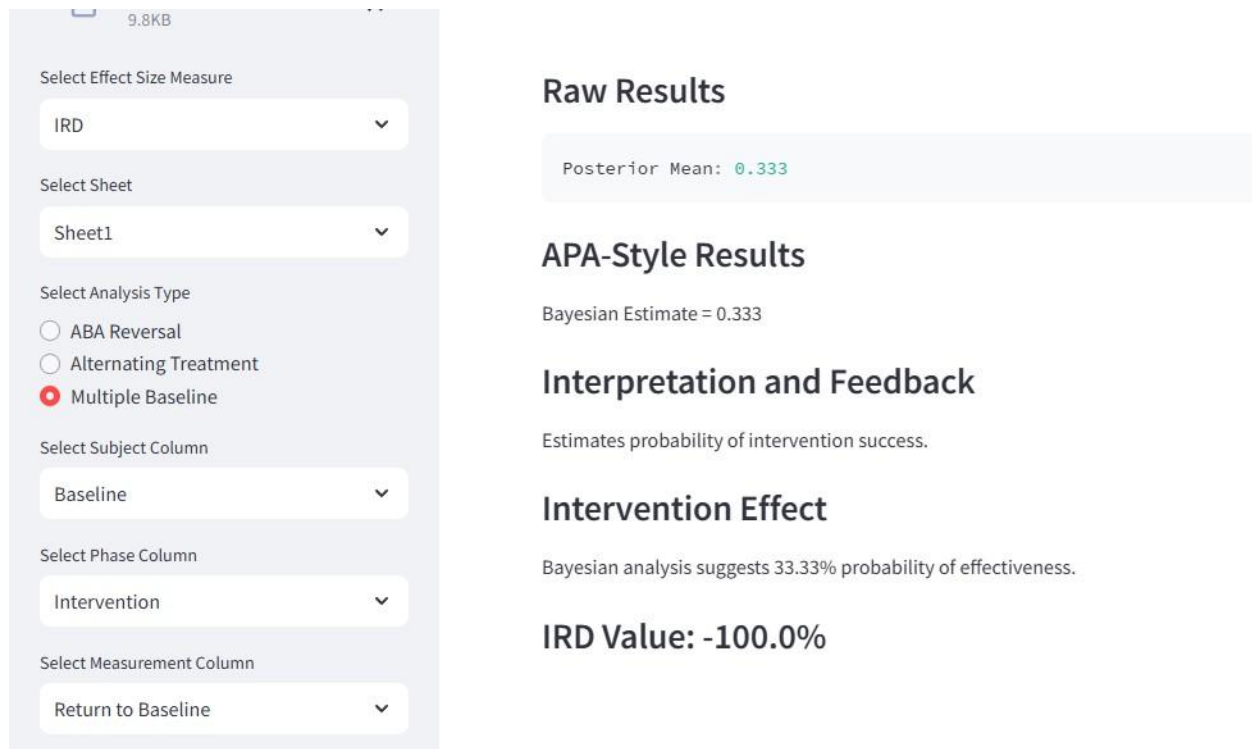
Intervention Effect

No significant effect.

IRD Value: -100.0%

**Figure 30:** *Multiple Baseline: Randomization Test Indicates No Effect*

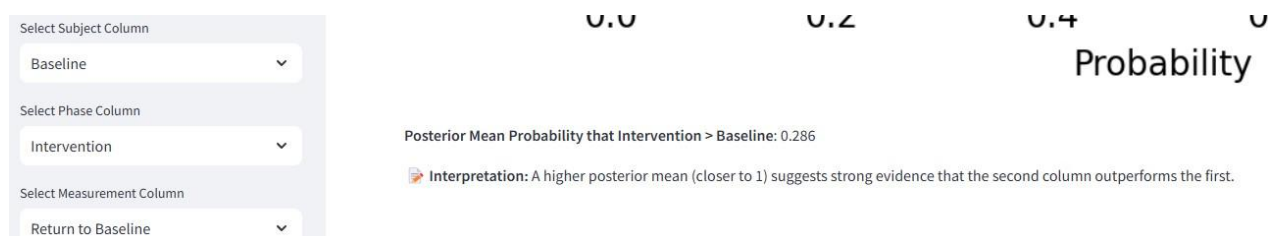
This section presents the results of a Randomization Test conducted within a Multiple Baseline Design, comparing the Baseline, Intervention, and Return to Baseline phases to assess the intervention's effectiveness. The statistical test revealed no observed difference (diff = 0.00) and a p-value of 1.000, indicating that there was no statistically significant intervention effect. Additionally, the IRD (Improvement Rate Difference) value is -100.0%, meaning that all intervention data points failed to outperform the baseline, suggesting a completely negative impact. While the Randomization Test found no measurable difference, the IRD metric reinforces the conclusion that the intervention may have been ineffective or even detrimental in this case. These findings underscore the importance of evaluating results through multiple analytical lenses to gain a more comprehensive understanding of intervention outcomes.



**Figure 31:** *Multiple Baseline: Bayesian Analysis with IRD Insight*

This result presents a Bayesian Analysis of a Multiple Baseline Design, estimating a posterior mean of 0.333, which translates to a 33.33% probability that the intervention is more effective than the baseline. While this suggests a modest chance of improvement, it falls below the conventional threshold for strong evidence.

Meanwhile, the IRD (Improvement Rate Difference) is -100.0%, indicating that the intervention consistently failed to outperform the baseline. The combination of a low Bayesian probability and highly negative IRD strongly suggests that the intervention had little to no beneficial impact across subjects.



**Figure 32:** *Bayesian Posterior Estimate: Intervention vs. Baseline*

This output shows the Bayesian posterior probability that the Intervention phase outperforms the Baseline phase, calculated as 0.286 (28.6%). According to the interpretation provided, a posterior mean closer to 1 suggests stronger evidence of intervention effectiveness. However, a value of



0.286 indicates weak support for improvement, implying that the intervention is unlikely to have produced a meaningful positive effect in comparison to the baseline condition.

While the interface streamlines the user experience, the development process was not without challenges. The subsequent section discusses these obstacles and the strategies used to overcome them

## 4.3 Challenges and Resolutions:

### 1. Handling Inconsistent Data Formats:

**Challenge:** Users frequently uploaded datasets with missing columns, inconsistent naming, or insufficient data points, leading to errors in computations and disrupted analysis.

**Resolution:** To address this, the app now features clear, markdown-based upload instructions and robust input validation, ensuring datasets are formatted correctly. It automatically checks for required columns and provides informative warnings when issues arise such as when fewer than two observations are present. These enhancements create a smoother user experience, reducing errors and allowing users to conduct analyses with confidence.

### 2. Supporting Multiple SCD Designs in One Interface

**Challenge:** Incorporating ABA Reversal, Alternating Treatment, and Multiple Baseline designs within a single tool introduced significant complexity in logic branching, data structure handling, and statistical method selection, requiring a streamlined approach to manage diverse analytical workflows.

**Resolution:** To ensure scalability and ease of maintenance, each design now operates within a distinct, modular logic block, allowing for conditional rendering of statistical options and effect size methods based on user selections. This structured approach enhances flexibility, ensures smooth user interactions, and maintains the integrity of design-specific analyses.

### 3. Displaying Results Consistently Across Tests

**Challenge:** Displaying outputs from parametric, non-parametric, and Bayesian statistical tests in a consistent and user-friendly format was challenging due to the varying structures of result presentations.

**Resolution:** To address this, a unified `display_results()` function was developed, ensuring that all analyses present Raw Results, APA-style summaries, Interpretation and Feedback, Effect Statements, and Effect Size Values in a clear and professional format. This approach maintains clarity across diverse statistical methods, making the outputs more accessible and standardized for users.

#### 4. Non-Normal Data and Inapplicable Tests

**Challenge:** Users frequently attempted to run parametric tests on non-normal or insufficient datasets, resulting in NaN outputs or misleading interpretations, making analysis unreliable.

**Resolution:** To prevent incorrect test selections, the app now automatically performs and visualizes normality checks using the Shapiro-Wilk test before users choose statistical methods. Additionally, it disables inapplicable tests for example, preventing the Wilcoxon test if the dataset lacks sufficient paired observations ensuring that only appropriate methods are executed. This enhancement improves accuracy, user guidance, and reliability in statistical analysis.

#### 5. Integrating Visualizations for Interpretation

**Challenge:** Users often found it difficult to interpret raw statistical outputs without additional context or visual support, making data-driven conclusions harder to understand.

**Resolution:** To improve interpretability, the app now incorporates visual elements such as histograms, Q-Q plots, box plots, and Bayesian posterior curves, providing users with clear graphical representations of their data. These enhancements support evidence-based conclusions, making statistical results more accessible and meaningful across different analysis methods.

#### 6. Managing Data Trimming Logic:

**Challenge:** In ABA and Multiple Baseline designs, comparing phases required consistent session lengths and stable data segments, but variations in phase durations made statistical comparisons uneven and potentially invalid.

**Resolution:** To ensure fair and valid comparisons, the app now automatically trims the final 5 observations per phase and equalizes their lengths, maintaining consistency across phases. These improvements enhance the rigor and transparency of statistical analysis, transforming the app into a reliable and user-friendly platform for conducting Single-Case Design (SCD) research

#### 7. Effect Size Misinterpretation Across Methods

**Challenge:** Users often encountered contradictory results between statistical tests and effect size measures—for example, a significant p-value alongside a 0% PND (Percentage of Non-overlapping Data)—leading to confusion and misinterpretation of intervention impact.

**Resolution:** To provide a well-rounded perspective, the tool now integrates three complementary effect size measures—PND, PEM, and IRD—allowing users to evaluate effectiveness from multiple angles. Additionally, clear explanations and APA-style feedback accompany each result, ensuring users understand what each metric reveals about behavioral change. These enhancements promote accurate interpretation, helping researchers make data-driven decisions with greater confidence and clarity.

## 8. Ensuring Stability and Compatibility Across Use Cases

**Challenge:** As new features and statistical pathways were integrated, unexpected bugs and inconsistencies emerged, particularly when switching between tabs or analysis types, leading to disruptions in workflow and usability.

**Resolution:** To ensure smooth functionality, the codebase was restructured to enable clean toggling between analysis types, prevent variable conflicts, and maintain stable rendering for key components such as plots and dropdown selections. A final QA sweep validated that all workflows—from ABA Reversal to the Bayesian Add-on—operate seamlessly end-to-end, ensuring a reliable and efficient user experience.

Resolving these development challenges enabled the project to meet, and in many cases exceed, its original goals. The following section evaluates these achievements against the project's initial objectives

### 4.4 - Achievements vs. Initial Objectives

**Goal 1:** Automate statistical analysis for single-subject behavior analysis.

→**Achievement:** Successfully implemented three design types out of four: ABA Reversal, Alternating Treatment, and Multiple Baseline.

→ABA Reversal Design: Includes detailed analysis using Paired t-tests, Wilcoxon Signed-Rank Test, Friedman test, Randomization test, Mixed-Effects Models, and Bayesian Logistic Regression.

→Alternating Treatment Design: Integrated Independent t-test, Mann-Whitney U Test, ANOVA, Kruskal-Wallis, Bayesian Analysis, and Randomization Test for comparing conditions.

→Multiple Baseline Design: Enabled cross-subject trend analysis using Paired tests, Randomization methods, and Bayesian estimates with phase trimming for uniform comparison.

**Goal 2: Expand statistical test options and integrate advanced modeling.**

→**Achievement:** Integrated a wide range of statistical methods supporting both parametric and non-parametric tests.

- Added tests include: t-tests, Wilcoxon, ANOVA, Kruskal-Wallis, Friedman, and Randomization tests.
- Advanced modeling incorporated: Bayesian posterior estimation and Mixed-Effects Linear Models for repeated measures.

- Built-in normality checks with Shapiro-Wilk test and graphical diagnostics (histogram, Q-Q plot, boxplot).

**Goal 3: Include robust and interpretable effect size metrics.**

**Achievement:** Implemented three effect size measures across all designs:

- PND (Percentage of Non-overlapping Data)
  - PEM (Percentage Exceeding the Median)
  - IRD (Improvement Rate Difference)
- These offer multiple ways to assess intervention effectiveness alongside statistical test outcomes.

**Goal 4: Build a user-friendly, web-based interface using Streamlit.**

**Achievement:** Developed a clean and interactive UI that supports:

- Sidebar-based controls for selecting dataset, sheet, columns, and statistical methods.
- Dynamic feedback, including data preview, normality plots, APA-style summaries, and visual outputs.
- Organized layout using Streamlit tabs for separating Main Analysis and Bayesian Add-on modules.

**Goal 5: Provide real-time APA-style summaries and interpretation.**

**Achievement:** Every statistical test outputs:

- Raw results,
- APA-formatted summaries,
- Interpretation of feedback
- Effectiveness labels,
- Plus, the selected effect size value for clear research reporting.

**Goal-6: Ensure robustness and extensibility for research use.**

**Achievement:**

- Validated tool across multiple datasets with error handling for missing or insufficient data.
- Designed modular functions for each component to allow easy future updates.
- Added a Bayesian Add-on tab to compare any two columns using probabilistic estimates.

Although the project has reached significant milestones, there remains ample opportunity for future refinement and expansion, as explored in the next section

## 4.5 -Suggestions for Future Improvements

### 1. Complete Integration of All Design Types

**Current Gap:** While the tool currently supports ABA Reversal, Alternating Treatment, and Multiple Baseline designs, the Changing Criterion Design has not yet been integrated, leaving an opportunity for expansion.

**Future Work:** To enhance the platform's capabilities, the next phase will focus on implementing the Changing Criterion Design, incorporating analysis methods such as time-series regression, slope-based trend detection, or piecewise linear models. Additionally, the design-switching logic will be refactored to ensure new designs integrate smoothly with the existing analysis\_type selection, maintaining a seamless user experience.

**Benefit:** Once implemented, users will have access to all four major Single-Case Design (SCD) frameworks within a unified platform, allowing for broader applications across behavioral studies while maintaining statistical rigor and flexibility.

### 2. Exportable Reports and Summary Outputs

- **Current Gap:** Users can view APA-style summaries but cannot export them for documentation or publication.
- **Future Work:** Add support for PDF or DOCX export, including test results, interpretation, effect sizes, and plots.

Allow batch exports summarizing results across multiple phases or participants.

- **Benefit:** Facilitates use in academic submissions, clinical reports, and progress tracking.

### 3. Effect Size Visualization and Summary Dashboard

**Current Gap:** Currently, effect size metrics (PND, PEM, IRD) are presented as numeric values only, lacking visual representation, which can make interpretation more challenging.

**Future Work:** To enhance data visualization, the next development phase will introduce bar charts and line plots displaying effect sizes per phase or subject, providing a clear graphical overview. Additionally, a dashboard panel will be implemented, summarizing effect sizes and statistical significance across different designs, offering a centralized and intuitive way to review results.

**Benefit:** These enhancements will improve interpretability, making it easier for users to grasp intervention impact at a high level, while maintaining clarity and accessibility across diverse statistical analyses.

#### 4. Automated Statistical Test Recommendation

**Current Gap:** Currently, users must manually select statistical tests, even when data assumptions clearly indicate the most appropriate method. This can lead to selection errors, particularly for those less familiar with statistical testing criteria. Factors such as normality, sample size, and number of conditions play a crucial role in determining the correct test, yet users may inadvertently choose a test that does not align with their dataset.

**Future Work:** To enhance accuracy and streamline the user experience, a rules-based system will be implemented. This system will recommend or auto-select the most suitable statistical test based on key dataset characteristics, including normality (Shapiro-Wilk test results), number of conditions, and sample size. Additionally, the UI will provide clear justifications for each recommendation, ensuring users understand why a specific method is suggested for their data.

**Benefit:** These enhancements will reduce user error, encourage correct statistical practice, and provide guidance for novice users. By integrating data-driven recommendations, the tool will help ensure that users apply appropriate methods, leading to more accurate and reliable results in their analyses.

#### 5. Custom Phase Labeling and Metadata Support

**Current Gap:** Currently, phase labels are directly tied to dataset column names, which may not always be user-friendly or descriptive. This limitation can make it difficult for users to interpret results clearly, especially when generating exported reports for formal documentation.

**Future Work:** To enhance usability and customization, the tool will introduce phase renaming capabilities, allowing users to modify labels for improved clarity in results and reports. Additionally, a metadata input feature will enable users to add participant names, session details, or notes, creating richer documentation that supports more detailed analysis and reporting.

**Benefit:** These improvements will provide greater flexibility in labeling and documentation, ensuring that reports are more intuitive and tailored for effective communication in both research and professional settings.

#### 6. Interactive Onboarding and Help System

**Current Gap:** While the tool features a user-friendly interface, it currently lacks a built-in onboarding or guidance system, which may make it challenging for first-time users to navigate and apply the correct procedures for their chosen design.

**Future Work:** To streamline the learning process, a step-by-step tutorial mode will be introduced, guiding first-time users through the key functionalities of the tool. Additionally, embedded tooltips and a sidebar help tab will provide explanations for each analysis method, effect size metric, and plot, ensuring users fully understand how to interpret their results.

**Benefit:** These enhancements will improve the learning curve, allowing users to quickly grasp the tool's capabilities and apply correct statistical procedures with confidence. By integrating accessible guidance, the platform will support both novice users and experienced researchers, ensuring a seamless and informed experience.

The project has reached a significant milestone in advancing automated statistical analysis for single-case design research in behavioral science. By successfully integrating three core experimental designs—ABA Reversal, Alternating Treatment, and Multiple Baseline—within a unified Streamlit interface, it provides researchers with accessible and efficient tools for conducting t-tests, Wilcoxon tests, Randomization Tests, Bayesian inference, and Mixed-Effects Models.

The intuitive interface supports essential features, including descriptive statistics, normality checks, effect size calculations (PND, PEM, IRD), and APA-style outputs, ensuring that behavioral data analysis remains clear, rigorous, and well-structured. Additionally, the introduction of a Bayesian Add-on module offers a flexible approach to performing probabilistic comparisons between experimental conditions, enhancing the depth of statistical insights available to users.

Despite these achievements, certain areas for refinement remain. The Multiple Baseline Design, while functional, requires tighter integration with the main application for a smoother user experience. The Changing Criterion Design has yet to be developed, marking an important future milestone for expanding the tool's capabilities. Further improvements—such as interactive visualizations, downloadable reports, automated test recommendations, and cloud deployment—would significantly enhance user engagement and accessibility, making the tool even more efficient and versatile.

Overall, the project stands as a powerful prototype that successfully meets its original objectives while laying the groundwork for future enhancements. As development continues, these refinements will strengthen its utility for clinicians, educators, and behavioral researchers, ensuring that statistical rigor and practical usability remain at the forefront of single-case design analysis.

As the project evolves, making the tool widely accessible becomes a key focus. The following section outlines hosting strategies, code storage, and steps for replicating the tool.



## 5. Hosting, Code storage, and instructions to replicate work

### Code Storage:

- The project code, containing all implemented features, documentation, and resources, is securely stored in a GitHub repository to facilitate easy access and collaboration. The repository is carefully organized to ensure clarity and usability, featuring well-structured directories, detailed comments within the code, and a README file that provides step-by-step setup and usage instructions. This structured approach allows future developers to seamlessly contribute, refine, and expand upon the existing framework, ensuring that the tool remains efficient, adaptable, and continually evolving.

**GitHub Repository:** <https://github.com/Abhiram58/AI-powereed-stastical-analysis-tool>

### Repository Structure

- **/Code:** Contains Python scripts for implementing statistical methods, the user interface, and back-end logic.
- **/Dataset:** Includes sample datasets for testing and demonstrating the tool's capabilities.
- **/Research\_Paper:** Stores research papers and reference materials relevant to the project.
- **README.md:** Provides a comprehensive overview of the project, setup instructions, and usage guidelines.
- **requirements.txt:** Lists all the dependencies required for the project to run successfully.

### Hosting

- The ultimate goal is to deploy the project as an online tool accessible to behavioral researchers and practitioners, removing the need for local installations. During development and testing, the tool was hosted locally using Streamlit.

**Current Hosting:** Local deployment using Streamlit.

### Future Hosting :

- **Streamlit Community Cloud:** Ideal for rapid prototyping and sharing your Streamlit-based application. It requires minimal setup, supports version control via GitHub, and is free for public apps — perfect for academic and research sharing.
- **Amazon Web Services (AWS):** Suitable for long-term hosting with scalable infrastructure. AWS services like EC2 or Elastic Beanstalk allow full control, higher security, and the flexibility to

integrate databases, authentication, and advanced analytics workflows. Best if you expect high traffic, private access, or integration with larger systems.

- **Heroku**

Offers a user-friendly platform-as-a-service (PaaS) for deploying Python web apps with moderate customization. Works well for small-scale apps with occasional use. Consider Heroku if you want a balance between ease of deployment and some backend flexibility beyond Streamlit Cloud.

**Instructions to Replicate:** The GitHub repository includes step-by-step instructions for setting up the tool on a local machine.

**Clone the Repository:**

Git clone : <https://github.com/Abhiram58/AI-powereed-stastical-analysis-tool>

**Set Up the Environment:**

1. Install Python (version 3.9 or later).
2. Create a virtual environment:  $\Rightarrow$  `python -m venv env`
3. Activate the virtual environment:  $\Rightarrow$  Windows: `env\Scripts\activate`  $\Rightarrow$  macOS/Linux: `source env/bin/activate`
4. Install required dependencies:  $\Rightarrow$  `pip install -r requirements.txt`

**Run the Application:**

1. Navigate to the project directory:  $\Rightarrow$  `cd AIpowered statistical analysis tool`
2. Start the server:  $\Rightarrow$  `streamlit run AIpowered statistical analysis tool`

**Upload and Analyze Data:**

- Use the interactive web interface to upload datasets in Excel format.
- Configure the analysis options and view results on the platform.

**Collaboration and Future Development (Current Status)**

- The project is structured with a modular approach, allowing components like effect size calculations, data preprocessing, and statistical tests to be maintained and extended independently.
- Core features such as percent overlap metrics, Bayesian models, and normality testing are

implemented in distinct functions, making it easier to integrate new methods (e.g., hierarchical modeling ,group-level tests).

- Initial inline documentation and function-level comments support understanding for contributors; a more formal README and contribution guide is in progress.
- GitHub repository has been initialized for version control, and will include issues, task lists, and milestones to facilitate collaboration and manage future development.
- Deployment is currently being tested via Streamlit Cloud, with plans to include setup instructions and deployment guides for other platforms (e.g., AWS, Heroku) in upcoming updates.

### **Looking Ahead :**

To make the AI-powered statistical analysis tool accessible to a broader audience, including behavioral researchers and practitioners, the next phase focuses on deploying it to a cloud platform such as Streamlit Cloud or Amazon Web Services (AWS). The current version of the application is a fully functional, modular Streamlit-based tool that supports ABA Reversal, Alternating Treatment, and Multiple Baseline designs. It incorporates a wide range of statistical tests, including t-tests, Wilcoxon, Randomization, and Bayesian analysis, along with effect size measures like PND, PEM, and IRD, all presented in APA-style outputs. By hosting the tool online, users will be able to upload their datasets, configure statistical analyses, and view results through a user-friendly web interface without requiring any local installations. This approach will enhance usability and significantly broaden the tool's reach. Comprehensive documentation for usage and deployment is in development, and future updates will include GitHub-based collaboration features such as issue tracking, contributor guidelines, and feature milestones. Ultimately, this transition to cloud deployment will ensure scalability, promote collaborative development, and establish the tool as a valuable resource for single-case design analysis in behavioral research.

## 6. Conclusion

The development of the AI-Powered Statistical Analysis Tool represents a significant step forward in advancing automated, accessible, and rigorous data analysis within the domain of Single-Case Design (SCD) research in behavioral science. This project was initiated with the goal of addressing persistent limitations in behavioral research—specifically, challenges related to small sample sizes, non-normal distributions, and the complexity of selecting appropriate statistical methods. The result is a comprehensive, user-friendly software application that democratizes access to advanced statistical analysis for educators, clinicians, and researchers alike.

Over the course of the project, the tool was designed and developed to integrate a range of powerful statistical techniques, including parametric tests (t-tests, ANOVA), non-parametric alternatives (Wilcoxon Signed-Rank Test, Kruskal-Wallis), and advanced Bayesian models, alongside resampling-based randomization methods. This statistical flexibility ensures that users can make valid inferences even when data assumptions are violated—a common issue in real-world behavioral datasets. The inclusion of effect size measures such as PND (Percentage of Non-overlapping Data), PEM (Percentage Exceeding the Median), and IRD (Improvement Rate Difference) further enhances interpretability by providing practical insights into intervention effectiveness.

Key to the project's success was its Agile development strategy, which emphasized iterative design, modular architecture, and responsiveness to user feedback. Through continuous testing and refinement, the tool evolved to include features such as normality checks with graphical diagnostics, last-5-session trimming for phase consistency, automated APA-style reporting, and an intuitive sidebar-driven user interface. These features not only improve usability but also support accurate and reproducible research practices.

The tool currently supports three core SCD frameworks—ABA Reversal, Alternating Treatment, and Multiple Baseline—each with a distinct analytical workflow tailored to its experimental structure. A dedicated Bayesian Logistic Add-on module was also introduced to allow users to estimate the probability of improvement in experimental conditions using posterior distributions. This probabilistic framework empowers researchers to explore effect likelihoods beyond binary hypothesis testing, aligning with modern standards in behavioral statistics.

Despite these accomplishments, the project recognizes opportunities for further development. Notably, the Changing Criterion Design remains to be integrated, and enhancements such as exportable result summaries, phase-specific visual dashboards, automated test recommendations, and deployment to cloud platforms (e.g., Streamlit Cloud or AWS) are planned for future iterations. These improvements will broaden the tool's applicability and reduce the technical barriers faced by less statistically trained users.

In conclusion, this project has met its core objectives by delivering a scalable, statistically robust, and user-focused tool for the behavioral research community. It stands not only as a valuable

contribution to the field of applied behavior analysis but also as a functional prototype capable of continued expansion and refinement. As behavioral researchers increasingly rely on data to inform evidence-based practices, this tool provides a much-needed bridge between advanced statistical modeling and practical, everyday decision-making in single-subject research.

### **Key Findings:**

#### **1.Support for Multiple SCD Designs:**

It effectively supports ABA Reversal, Alternating Treatment, and Multiple Baseline designs with tailored workflows and effect size analysis.

#### **2. APA-Style Summaries and Visualization:**

Each statistical test output includes raw results, APA formatting, visual plots (Q-Q plots, histograms, box plots), and interpretive feedback for easy reporting.

#### **3. Bayesian Add-on Integration:**

A dedicated Bayesian logistic regression module estimates posterior probabilities of treatment effectiveness with graphical output.

#### **4. Error Handling and Guidance:**

The tool checks data integrity before analysis and disables incompatible tests, minimizing user error.

## References

- [1] Pustejovsky, J. E., L. V. Hedges and a. W. R. Shadish., "Design-comparable effect sizes in multiple baseline designs: A general modeling framework.," *Journal of Educational and Behavioral Statistics* 39.5 , pp. 368-393., 2014.
- [2] Wendt, Oliver and a. D. Rindskopf, "Advancements in meta-analysis of single-case experimental designs.," *Evidence-Based Communication Assessment and Intervention* 17.1, pp. 1-5., 2023.
- [3] Tang, Jillian and a. R. D. Landes, "Some t-tests for N-of-1 trials with serial correlation.," *Plos one* 15.2 , p. e0228077, 2020.
- [4] Shadish and R. William, "Statistical analyses of single-case designs: The shape of things to come.," *Current Directions in Psychological Science* 23.2, pp. 139-146., 2014.
- [5] Moeyaert, Mariola, S. Bursali and a. J. M. Ferron, "SCD-MVA: A mobile application for conducting single-case experimental design research during the pandemic.," *Human Behavior and Emerging Technologies* 3.1, pp. 75-96., 2021.
- [6] Lu, Yin, M. Scott and a. P. Raghavan, "A Statistical Framework for Single Subject Design with an Application in Post-stroke Rehabilitation.," *arXiv preprint arXiv:1602.03855*, 2016.
- [7] Lønfeldt and N. N., "Computational behavior recognition in child and adolescent psychiatry: a statistical and machine learning analysis plan.," *arXiv preprint arXiv:2205.05737*, 2022.
- [8] Lanovaz, M. J, A. R, Giannakakos and a. O. Destras, "Machine learning to analyze single-case data: A proof of concept.," *Perspectives on Behavior Science* 43.1, pp. 21-38., 2020.
- [9] Barnard-Brak, Lucy and a. L. W. David. M. Richman, "Introduction to the special section: translating advanced quantitative techniques for single-case experimental design data.," *Perspectives on Behavior Science* 45.1, pp. 1-4., 2022.
- [10] Scruggs, E. Thomas, A. Margo, Mastropieri, Casto and a. Glendon, "Reply to Owen White," *Remedial and Special Education*, vol. 8.2, p. 42, 1987.
- [11] J. E. Pustejovsky, "Procedural Sensitivities of Effect Sizes for Single-Case Designs With Directly," *U.S. Department of Education, Institute of Educational Sciences*.