oneR: Automated Statistical Testing with Normality Assessment

Author: Bright Boamah

2025-06-17

Introduction

The oneR package provides a streamlined approach to statistical testing by automatically assessing data normality and selecting appropriate parametric or non-parametric tests. This vignette demonstrates the package's functionality and provides comprehensive examples for various use cases.

Package Philosophy

Statistical testing often requires researchers to make decisions about which test to use based on data characteristics, particularly normality assumptions. The oneR package automates this decision-making process while maintaining statistical rigor and providing comprehensive documentation of the analysis process.

Key Features

- **Automated test selection**: Chooses between parametric and non-parametric tests based on normality assessment
- **Comprehensive visualization**: Generates plots for normality assessment and test results
- **PDF reporting**: Creates detailed reports with all analysis components
- **Flexible interface**: Supports one-sample, two-sample, and paired tests
- **Statistical rigor**: Uses established tests (Shapiro-Wilk, t-test, Wilcoxon) with proper implementation

Installation and Setup

```r
# Install required packages if not already installed
install.packages(c("ggplot2", "gridExtra", "knitr", "rmarkdown"))

# Load the oneR package
library(oneR)
```

Basic Usage

One-Sample Testing

The most basic use case involves testing whether a single sample comes from a population with a specific mean.

```r
# Generate sample data
set.seed(123)
sample_data <- rnorm(30,
mean = 100, sd = 15)

# Perform automated testing
```

```r
result <- oneR_test(sample_data, mu = 100)

# View results
print(result)
```

Understanding the Output

The oneR_test function returns a comprehensive object containing:

- **Normality assessment**: Results from Shapiro-Wilk tests
- **Test selection**: Which statistical test was chosen and why
- **Test results**: Complete statistical output including p-values and confidence intervals
- **Recommendations**: Interpretation of results in plain language

Two-Sample Testing

Comparing two independent groups is equally straightforward:

```r
# Generate two groups
set.seed(456)
group_a <- rnorm(25, mean = 85, sd = 12)
group_b <- rnorm(25, mean = 90, sd = 12)

# Perform comparison
comparison <- oneR_test(group_a, group_b)

# View summary
summary(comparison)
```

Paired Testing

For paired or repeated measures data:

```r
# Generate paired data
set.seed(789)
before <- rnorm(20, mean = 75, sd = 10)
after <- before + rnorm(20, mean = 5, sd = 3) # Treatment effect

# Perform paired test paired_result <-
oneR_test(before, after, paired = TRUE)
print(paired_result)
```

Advanced Features

Customizing Analysis Parameters

The oneR_test function provides several parameters for customizing the analysis:

```r
# Custom significance levels and
alternatives result <- oneR_test( x =
sample_data, mu = 95,
  alternative = "greater", # One-tailed test
  alpha = 0.01,    # Stricter significance level
  conf.level = 0.99      # Higher confidence
  level
)
result
```

Parameter Guide

Parameter	Description	Default	Options
x	Primary data vector	Required	Numeric vector
y	Second group (optional)	NULL	Numeric vector
mu	Hypothesized mean	0	Numeric value

`alternative`	Alternative hypothesis	"two.sided"	"two.sided", "less", "greater"
`alpha`	Significance level for normality	0.05	0 < < 1
`conf.level`	Confidence level for test	0.95	0 < level < 1
`paired`	Paired test indicator	FALSE	TRUE/FALSE

Visualization Capabilities

Normality Assessment Plots

The package generates comprehensive plots to assess data normality:

```
# Generate normality assessment plots
plot_normality(result)
```

These plots include:

- **Histograms** with normal distribution overlay
- **Q-Q plots** with reference lines
- **Boxplots** with Shapiro-Wilk p-values

Test Results Visualization

Results can be visualized to understand the data and test outcomes:

```
# Generate test results
plots
plot_results(comparison)
```

For two-sample tests, this includes:

- **Side-by-side boxplots** for group comparison
- **Density plots** showing distribution overlap
- **Summary panels** with key statistics

Integrated Plotting

The plot method provides convenient access to all visualizations:

```
# Show all plots
plot(result, type =
"both")

# Show only normality plots
plot(result, type = "normality")

# Show only results plots
plot(result, type = "results")
```

PDF Report Generation

Basic Report Generation

Creating comprehensive PDF reports is straightforward:

```
# Generate standard report
library(latexpdf)
oneR_report(result,
"analysis_report.pdf")
```

Customized Reports

Reports can be customized with various options:

```
# Customized report with raw data
oneR_report( result,
  output_file = "detailed_analysis.pdf",
  title = "Clinical Trial Statistical
  Analysis", author = "Research Team",
  include_data = TRUE
)
```

Report Contents

Generated reports include:

1. **Executive Summary**: Key findings and conclusions
2. **Data Overview**: Descriptive statistics for all groups
3. **Normality Assessment**: Detailed Shapiro-Wilk test results
4. **Statistical Test Results**: Complete test output with interpretation
5. **Visualizations**: All relevant plots embedded in the document
6. **Conclusions and Recommendations**: Statistical interpretation and guidance
7. **Technical Details**: Software information and methodology
8. **Raw Data** (optional): Complete dataset tables

Real-World Examples

Example 1: Clinical Trial Analysis

```


# Simulate clinical trial data
set.seed(2024)
placebo_group <- rnorm(50, mean = 120, sd = 15) # Blood pressure
treatment_group <- rnorm(50, mean = 110, sd = 15)

# Perform analysis
clinical_analysis <- oneR_test(placebo_group, treatment_group)

# Generate comprehensive report
oneR_report( clinical_analysis,
  "clinical_trial_report.pdf", title = "Blood
Pressure Treatment Efficacy Analysis", author =
"Clinical Research Team" )
```

Example 2: Quality Control Testing

```
# Manufacturing quality control data
set.seed(2024)
production_measurements <- c(
  rnorm(30, mean = 50.2, sd = 0.8), # Normal production runif(5,
  min = 48, max = 52)    # Some outliers
)

# Test against specification
qc_result <-
oneR_test( production_measurements,
mu = 50.0,
  alternative = "two.sided"
)

# Quick assessment print(qc_result)
```

Example 3: Educational Assessment

```
# Pre/post training scores
set.seed(2024) pre_scores <-
rbeta(25, 2, 3) * 100   #
Skewed distribution post_scores
<- pre_scores + rnorm(25, mean
= 8, sd = 5)

# Paired analysis
training_effect <-
oneR_test(pre_scores,
post_scores, paired = TRUE)

# Visualization
plot(training_effect, type =
"both")
```

Statistical Methodology

Normality Testing

The package uses the Shapiro-Wilk test for normality assessment, which is appropriate for sample sizes up to 5000. The test evaluates the null hypothesis that the data comes from a normal distribution.

Test Selection Logic

The automated test selection follows this decision tree:

1. **Assess normality** for all groups using Shapiro-Wilk test 2. **If all groups are normal** (p >):
   - Use parametric tests (t-tests)
3. **If any group is non-normal** (p):
- Use non-parametric tests (Wilcoxon tests)

Parametric Tests Used

- **One-sample t-test**: Compares sample mean to hypothesized value
- **Two-sample t-test**: Compares means of two independent groups
- **Paired t-test**: Compares paired observations

Non-Parametric Tests Used

- **Wilcoxon signed-rank test**: Non-parametric alternative to one-sample and paired t-tests
- **Wilcoxon rank-sum test**: Non-parametric alternative to two-sample t-test

Advantages of Automated Selection

1. **Reduces user error** in test selection
2. **Ensures appropriate assumptions** are met
3. **Provides consistent methodology** across analyses
4. **Documents decision rationale** in reports

Best Practices

Data Preparation

Before using oneR, ensure your data is properly prepared:

```r
# Remove missing values
clean_data <-
data[!is.na(data)]

# Check for outliers
boxplot(clean_data)

# Verify data types
str(clean_data)
```

Sample Size Considerations

- **Minimum sample size**: 3 observations (required for Shapiro-Wilk test)
- **Recommended minimum**: 10-15 observations for reliable normality assessment
- **Maximum for Shapiro-Wilk**: 5000 observations

Interpretation Guidelines

P-value Interpretation

- **p <** : Reject null hypothesis (statistically significant)
- **p** : Fail to reject null hypothesis (not statistically significant)

Effect Size Considerations

While oneR focuses on statistical significance, consider practical significance:

- **Confidence intervals**: Indicate range of plausible effect sizes
- **Mean differences**: Assess practical importance
- **Context**: Consider domain-specific meaningful differences

Common Pitfalls to Avoid

1. **Multiple comparisons**: Adjust significance levels when performing multiple tests
2. **Sample size**: Ensure adequate power for detecting meaningful effects
3. **Assumptions**: Verify that chosen tests are appropriate for your data
4. **Interpretation**: Distinguish between statistical and practical significance

Troubleshooting

Common Issues and Solutions

Issue: “Data must have at least 3 observations”

Solution: Ensure your data vectors have sufficient observations for normality testing.

```r
# Check data length
length(your_data)

# Remove missing values
clean_data <- your_data[!is.na(your_data)]
```

Issue: PDF generation fails

Solution: The package will automatically generate HTML reports if PDF creation fails.

```r
# Check if required packages are installed
install.packages(c("rmarkdown", "knitr"))

# Try HTML output explicitly oneR_report(result,
"report.html")
```

Issue: Plots not displaying correctly

Solution: Ensure graphics devices are properly configured.

```r
# Reset graphics parameters
dev.off()

# Try different plot types
plot(result, type =
"normality")
```

Extending oneR

Custom Plotting

You can create custom visualizations using the data from oneR objects:

```
# Extract data from oneR
result x_data <-
result$data_x y_data <-
result$data_y

# Create custom plots
library(ggplot2)
custom_plot <- ggplot(data.frame(x = x_data), aes(x = x)) +
  geom_histogram(bins = 20, alpha = 0.7) +
  theme_minimal() +
  labs(title = "Custom Data Visualization")
print(custom_plot)
```

Integration with Other Packages

oneR results can be easily integrated with other statistical packages:

```
# Extract results for further analysis
test_results <- extract_results(result)

# Use with other packages
library(broom) tidy_results <-
broom::tidy(result$test_result)
```

Conclusion

The oneR package provides a comprehensive solution for automated statistical testing with proper normality assessment. By combining rigorous statistical methodology with user-friendly interfaces and comprehensive reporting, oneR enables researchers to conduct reliable statistical analyses efficiently.

Key benefits include:

- **Automated decision-making** reduces errors in test selection
- **Comprehensive visualization** aids in data understanding
- **Professional reporting** facilitates communication of results
- **Statistical rigor** ensures appropriate methodology

For additional support and examples, consult the package documentation and help files.

References

1. Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4), 591-611.

2. Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6), 80-83.

3. Student. (1908). The probable error of a mean. *Biometrika*, 6(1), 1-25.

4. R Core Team (2024). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.