

Analyzing the Ringelmann Effect with the Repeated Measures ANOVA

Nikolas Höft

Constantin Meyer-Grant

Joachim Munch

Quang Nguyen Duc

Frederik Schreck

Statistical Programming Languages

Humboldt-Universität zu Berlin



MAGA: The Package to make ANOVA great again

- ▣ The package bundles functionalities around the grand topic repeated measures ANOVA.
- ▣ Some of the functionalities have not been implemented in R yet. This package aims to fill this void.
- ▣ Each core functionality of the package represents a quantlet.
- ▣ After presenting the theory and code examples from the package, we will give a short overview of the technical implementation.

Outline

1. The Ringelmann Effect
2. The Repeated Measures ANOVA
 - 2.1 Based on the ANOVA Model
 - 2.2 An Advantageous Model
 - 2.3 Confidence Intervals
 - 2.4 Effect Size Measures
3. An Important Requirement
4. Orthogonal Polynomial Contrasts
5. Our Package
 - 5.1 Motivation for Making a Package
 - 5.2 Tools to Create a Package in R
 - 5.3 Things to Consider

The Ringelmann Effect

- Maximilian Ringelmann (1861-1931):
 - ▶ French professor of agricultural engineering
- Findings:
 - ▶ Work performance depends of number of group size
 - ▶ Decreasing individual performance with increasing group size
 - ▶ Example: Pulling weights in differently sized groups

The Ringelmann Effect

- The Ringelmann Effect can be investigated with an experimental design
 - ▶ Dependent Variable: Individual performance
 - ▶ Independent Variable / Factor: Group size
 - ▶ Realization of different factor levels
- For our purpose: Data simulation



Quantlet 1: Data Simulation



▣ Simulation function:

```
1 sim_ow_rma_data(n, k, means = c(10, 5, 7),  
2   poly_order = NULL, noise_sd = 10,  
3   between_subject_sd = 40, NAs = 0)
```

▣ Simulate deviation between subjects:

```
1 mean_deviation = rnorm(n, mean = 0,  
2   sd = between_subject_sd)  
3 ow_rma_data[, 2:(k + 1)] = ow_rma_data[, 2:(k + 1)]  
4   + mean_deviation
```

Quantlet 1: Data Simulation



□ Simulate noise:

```
1 noise = matrix(NA, nrow = n, ncol = k)
2   for (i in 1:k) {noise[, i] = rnorm(n,
3     mean = 0, sd = noise_sd[i])}
4 ow_rma_data[, 2:(k + 1)] = ow_rma_data[, 2:(k + 1)]
5   + noise
```

The Ringelmann Effect

Subject	Factor.1	Factor.2	Factor.3	Factor.4	Factor.5
1	218.25	147.13	69.18	74.96	80.11
2	173.77	119.62	114.15	94.04	87.57
3	177.49	116.17	97.97	72.91	69.28
4	126.58	110.36	123.45	90.82	75.07
5	146.61	108.26	86.91	76.62	61.94
6	167.03	95.48	72.13	93.29	102.31

Table 1: The first 6 observations of our simulated data.

The Repeated Measures ANOVA: Based on the ANOVA Model

- ANOVA: Analysis of Variance
- Comparison of the k factor level means
- Hypotheses:

$$H_0 : \mu_1 = \mu_2 = \dots = \mu_k$$

$$H_1 : \exists i \neq j : \mu_i \neq \mu_j$$

- Test is accomplished by decomposition of variance components
- ANOVA is used for independent data
- For dependent data: Repeated Measures ANOVA



The Repeated Measures ANOVA: An Advantageous Model

- Design Requirement: Each subject has to be measured under all factor levels

Subject	Factor.1	Factor.2	Factor.3	Factor.4	Factor.5
1	218.25	147.13	69.18	74.96	80.11
2	173.77	119.62	114.15	94.04	87.57
3	177.49	116.17	97.97	72.91	69.28
4	126.58	110.36	123.45	90.82	75.07
5	146.61	108.26	86.91	76.62	61.94
6	167.03	95.48	72.13	93.29	102.31

Table 2: Our simulated data consists of dependent data.


Quantlet 2: Repeated Measures ANOVA



```
1 rma = function(rma_data, id = 1)
```

- ⇒ Checks the Requirements of the data
- ⇒ Computes the Repeated Measures ANOVA components
- ⇒ Defines the decomposition matrix
- ⇒ Computes F-values and p-values
- ⇒ Constructs an ANOVA table

The Repeated Measures ANOVA: An Advantageous Model

- Problem of ANOVA: In case of large variance between different subjects
⇒ High error variance ⇒ Loss of power in F-Test
- Repeated Measures ANOVA considers the between subject variance separately
⇒ Relatively low error variance ⇒ Gain of power in F-Test 

Quantlet 3: ANOVA and SSE Reduction



```
1 ow_a = function(rma_data, id)
```

```
1 rma_sse_reduct = function(rma_data, id = 1,  
  plot_type = "pie", return_anova_table = FALSE)
```

The Repeated Measures ANOVA: An Advantageous Model

	Source	Sum of squares	Degrees of freedom	Mean squares	F-value	p-value
1	Baseline	1769610.20	1.00	1769610.20	1430.36	0.00
2	Factor	174706.07	4.00	43676.52	142.23	0.00
3	Subject	33403.82	27.00	1237.18		
4	Error	33166.05	108.00	307.09		
5	Total	2010886.14	140.00			
6	Corrected total	241275.94	139.00	1735.80		

Table 3: ANOVA-table for our Repeated Measures ANOVA.

The Repeated Measures ANOVA: An Advantageous Model

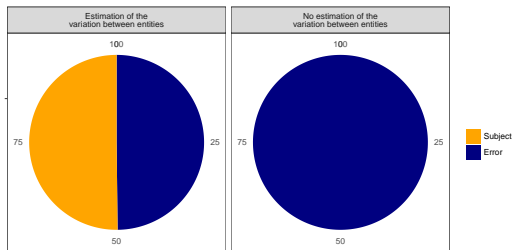


Figure 1: Pie chart on the reduction of sum of squares (SSE) in percentages.

The Repeated Measures ANOVA: Confidence Intervals

- The computation of the confidence intervals has to be adjusted in the Repeated Measures ANOVA

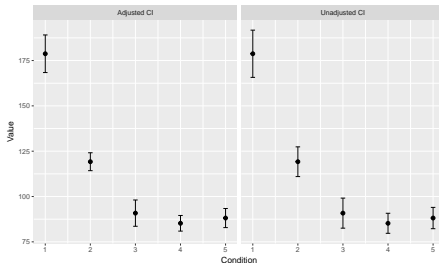


Figure 2: Unadjusted and adjusted confidence intervals.

Analyzing the Ringelmann Effect with the Repeated Measures ANOVA –



Quantlet 4: Confidence Intervals



```
1 rma_ci = function(rma_data, C_level = 0.95, id = 1,  
  print_plot = TRUE)
```

The Repeated Measures ANOVA: Effect Size Measures

□ Two measures of effect size:

- ▶ η^2
- ▶ η_p^2

	Source	eta squared	partial eta squared
1	Factor	0.72	0.84

Table 4: Effect size measures for our simulated data.

Quantlet 5: Effect Size Measures



```
1 rma_eta = function(rma_data, id = 1, append = FALSE)
```



An Important Requirement

- Sphericity: The variance of differences are equal for each pair of factor levels
- Test for sphericity: Mauchly test
- Measurement of sphericity ($\epsilon \in [0, 1]$):
 - ▶ Greenhouse & Geisser: ϵ_{GG}
 - ▶ Box: ϵ_B
 - ▶ Huynh & Feldt: ϵ_{HF}
- These can be used to correct the degrees of freedom and therefore adjust the p-values if sphericity is violated

Quantlet 6: Test and Adjustment for Sphericity



```
1 rma_spheri = function(rma_data, id = 1, append =  
  FALSE)
```

Table

	Source	Mauchly's W	Chi square	df	p
1	Factor	0.20	41.48	9.00	0.00

Table 5: Adjustment for sphericity for our simulated data.

Orthogonal Polynomial Contrasts

- ▣ Further analysis of factor effect
- ▣ Requirement: Level of measurement at least interval
- ▣ Factor effect can be decomposed into polynomial trend components
- ▣ Polynomial trend components can be tested by polynomial contrasts
- ▣ If there shall be no redundant information in each trend component, the contrasts have to be orthogonal
 - ▶ Maximum of orthogonal contrasts: $k - 1$

Quantlet 7: Orthogonal Polynomial Contrasts



```
1 rma_opc = function(rma_data, id = 1, maxpoly = NA,  
  print_plot = TRUE)
```


Orthogonal Polynomial Contrasts

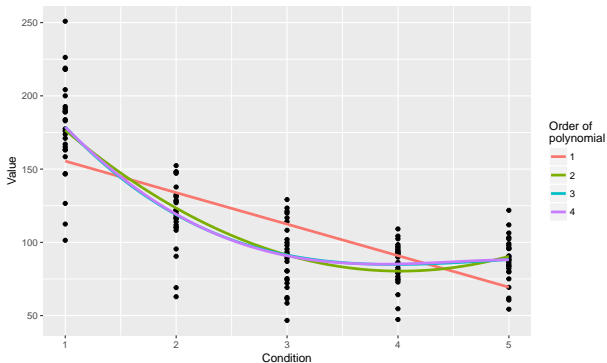


Figure 3: Orthogonal polynomial contrasts.

Our Package: Motivation for Making a Package

- ▣ A package bundles together code, data, documentation, and tests
- ▣ Makes it easy to share and publish code with others (CRAN, Github via Devtools)
- ▣ Loads all relevant functions into the namespace
- ▣ Automatically checks and installs dependency if necessary
- ▣ Packages allow to document functions, so that they easily be used by others (help function, argument list, etc.)

Our Package: Tools to Create a Package in R

- roxygen2
 - ▶ Enables documentation to be written directly into the R script
- devtools
 - ▶ Load packages still under development e.g. from Github
- Github
 - ▶ A package can be handled like a repository, which enables collaboration
- RStudio
 - ▶ Provides many helpful functionalities for creating a package (create, build, check)

Our Package: Things to consider

- Use function names that speak for themselves and use them consistently.
 - ▶ “There are only two hard things in computer science: cache invalidation and naming things.” Phil Karlton
- Error handling
 - ▶ Make sure that functions are robust regarding violation of the required input, e.g. character vector supplied although a numeric vector is needed. Use if-statements or try().
- Custom error and warning messages
 - ▶ stop() interrupts the code and returns an error message
 - ▶ warning() executes the code but returns a warning message

Thank you for your Attention!