Implementation of the Repeated Measurement ANOVA

Statistical Programming Languages

Vorname, Name
Matrikelnummer:

January 29, 2017

Contents

1 Theorie and Motivation **3**

2 Simulating and Preparing the Data **3**

2.1 Simulation 3

2.2 Listwise Deletion 3

3 Estimating the Repeated Measures ANOVA **4**

3.1 Based on the ANOVA model 4

3.2 Estimation of our Repeated Measures ANOVA model 5

4 SSE Reduction with the Repeated Measures ANOVA **8**

4.1 Estimation of our ANOVA model 8

4.2 Comparing the Error Terms 10

5 Confidence Intervals(CI) **12**

5.1 Unadjusted CIs 13

5.2 Adjusted CIs 13

5.3 Plotting the CIs 13

6 Sphericity **13**

6.1 Test for Sphericity 13

6.2 Adjustment for Sphericity 13

7 Orthogonal polynomial contrasts **13**

7.1 Computing the Orthogonal Polynomial Contrasts 13

7.2 Plotting the Contrasts 13

1 Theorie and Motivation

2 Simulating and Preparing the Data

Short theoretical introduction

2.1 Simulation

In order to demonstrate and evaluate the functions presented later in this report, we have developed a function to simulate data, which can then be used to estimate repeated measurement ANOVA models. First we will shortly present the functionalities and the implementation of the simulation function.

```
# Run the data simulation
rma_data = sim_rma_data(n = 1000, k = 4, means = NULL, poly_order = 5,
    noice_sd = c(10, 20, 30, 20), between_subject_sd = 40, NAs = 0)
```

The data can be simulated by running the function shown above. The function includes functionalities for simulating orthogonal polynomial contrast and sphericity, which can be specified by passing arguments. In the following the implementation and the functionality will be explained.

The first two arguments of the function n and k are obligatory. n defines the number of observation and k the number of factors to be simulated. The output of the function will therefore be an matrix of the size n x (k + 1). The first column contains the subject ids to identify each simulated observation and the following columns represents the factors.

The first step, when simulating the data is to simulate the means of each factors. Thereby each factor columns is filled with the mean for the corresponding factor. This results in all observations having the same value for each factor, in the next step we will therefor simulate the differences between the subjects. Additionally by passing an integer not larger than k to the argument poly_order, the means will be simulated so that they create a polynomial contrast in the data.

Instead of letting the function simulate the means, a vector of the lengths k containing the means that should be used for each factor can be passed to the function.

2.2 Listwise Deletion

When computing a Repeated Measures ANOVA, the way to deal with missing values is listwise deletion. As we measure the same subjects over different factor levels, a missing value in one factor level leads to a dropout of the whole observation from our analysis.

Since we use simulated data, we could easily avoid having missing values. However, for demonstration purposes as well as for applicability to other data, we integrated a condition for listwise

deletion. In the above mentioned data simulation function rma_data, the number of NAs is specified, that then randomly replace values of factor levels in the simulated data. Subsequently, the listwise deletion condition checks for existing NAs and drops out the corresponding subject(s), while displaying a message informing about the id(s) of the subject(s) that has/have been deleted.

```
# Listwise deletion
deletionvector = vector(mode = "numeric", length = 0)

for (i in 1:nrow(rma_data)) {
 if (any(is.na(rma_data[i, ])) == TRUE) {
  deletionvector = union(deletionvector, i)
  print(paste("Listwise deletion due to missing value(s) for subject", i))
 }
}

rma_data = rma_data[-deletionvector, ]
```

3 Estimating the Repeated Measures ANOVA

Center of our analyzes is the Repeated Measures ANOVA model. Since the Repeated Measures ANOVA is a variation of the ANOVA model, we will first give the necessary background information on the ANOVA model. Subsequently, we will present the computation of our Repeated Measures ANOVA model.

3.1 Based on the ANOVA model

The (one-way) ANOVA, an abbreviation for "Analysis of Variance", is used to analyze the dependency between an interval scaled dependent variable and a categorical independent variable. The independent variable must therefore have at least two different categories, which are called factor levels in the ANOVA context. It is then tested, whether differences in the means of the dependent variable broken down by the factor levels of the independent variable can be found. The hypotheses for k factor levels are therefore given by:

$$H_0 : \mu_1 = \mu_2 = ... = \mu_k$$
$$H_1 : \exists i \neq j : \mu_i \neq \mu_j$$

The test is accomplished by a decomposition of variance of the dependent variable. From the decomposed elements, the sum of squares of the unrestricted model and the restricted model are computed. In the former, different means for each factor level are assumed, whereas in the latter no such difference is assumed. By use of the F-test, the unrestricted and the restricted model are

then compared in order to make a decision on the hypotheses.

An important requirement in order to compute an ANOVA is independent data. This requirement is fulfilled, when different subjects are measured in each of the factors. Complimentary, in case of dependent data, the Repeated Measures ANOVA is an appropriate model. The Repeated Measures ANOVA is taking use of the dependency structure by additionally calculating the variance between subjects. This subject variance can be separated from the error variance. As a consequence, a high subject variation (with regard to our simulated data: large differences in pulling weights between our subjects) does not lead to a loss of power in the F-test. We will take a closer look at this property of the Repeated Measures ANOVA in section 4.2.

3.2 Estimation of our Repeated Measures ANOVA model

The computational steps of our Repeated Measures ANOVA will be explained in the following by means of our code. All of these are part of the following funtion:

```
# Repeated Measures ANOVA function
rma(rma_data, id = 1)
```

In the first argument the data used to compute the Repeated Measures ANOVA is specified. Four our purposes, we use our simulated data. The argument id specifies the number of independent variables. In general, our code can be applied to every data-set. Hence, we first define the needed constants and check for the requirements of the data. If any of the requirements is not fulfilled, we stop the computation and display the corresponding requirement in a warning message, as can be seen in the following.

```
# Number of entities
n = nrow(rma_data)

# Number of factor levels
k = ncol(dependent_variable)

# Check data requirements
# all variables must be numeric
if (all(sapply(rma_data, is.numeric)) == FALSE | any(sapply(rma_data,
    is.factor))) {stop("All variables in rma_data must be numeric")}

# n > k (i.e. more entities than factor levels)
if (n <= k) {stop("Number of entities must exceed number of factor levels")}

# k >= 2 (i.e. at least two or more factor levels)
if (k < 2) {stop("At least two factor factor levels required")}
```

In order to compute the Repeated Measures ANOVA, we need to decompose the variance of the dependent variable. For that, we define the basic ANOVA components. Each of the components is given by a matrix of size of the used data (n*k). The baseline component is hereby just the overall mean of the dependent variable. The factor level component is then given by the difference between the means of the dependent variable conditional to the factors of the independent variable. It has already been mentioned, that, in contrast to the ANOVA model, the variance between subjects is calculated separately for the Repeated Measures ANOVA model. Hence, we specify a subject component, which is given by the difference between the individual means of each subject and the overall mean of the dependent variable. By subtracting the three components from the values of the dependent variable, we then get the error component, containing the individual errors for each subject under each factor level.

```
# Define basic anova components
grand_mean = mean(dependent_variable)
baseline_components = matrix(grand_mean, nrow = n, ncol = k)

conditional_means = colMeans(dependent_variable)
factor_level_components = matrix(conditional_means - grand_mean, nrow = n,
    ncol = k, byrow = TRUE)

subject_means = rowMeans(dependent_variable)
subject_components = matrix(subject_means - grand_mean, nrow = n, ncol = k)

error_components = dependent_variable - baseline_components -
    factor_level_components - subject_components
```

After having computed these basic ANOVA components, we construct the so called decomposition matrix. The decomposition matrix stacks together the vectorized ANOVA components and is used for further computation. Hence it consists of k*n rows and five columns; one column for the original values, one for the baseline component, one for the factor level component, one for the subject component and one for the error component. By summing up the columns of the decomposition matrix and square the result, we obtain the sum of squares of each ANOVA component.

```
# Construct decomposition matrix
decomposition_matrix = data.frame(dependent_variable =
    as.vector(dependent_variable), baseline = as.vector(baseline_components),
    factor_level = as.vector(factor_level_components), subject_level =
    as.vector(subject_components), error = as.vector(error_components))

# Compute sums of squares
ss = as.data.frame(t(colSums(decomposition_matrix^2)))
rownames(ss) = "sums_of_squares"
```

Subsequently, we define the degrees of freedom. These are then used to compute the mean squares of the decomposed components. Further we correct the total sum of squares, which gives the variance in the dependent variable.

```
# Set degrees of freedom
dof = data.frame(dependent_variable = n * k, baseline = 1, factor_level = k -
    1, subject_level = n - 1, error = (n * k) - 1 - (k - 1) - (n - 1))

# Compute mean squares
ms = ss/dof
rownames(ms) = "mean_squares"

# Compute corrected total sum of squares
corrected_sst = ss$dependent_variable - ss$baseline
variance = corrected_sst/(dof$dependent_variable - dof$baseline)
```

Having defined all components we need, we compute the F-test and the corresponding p-value.

```
# Compute F-values
F_value_factor = ms$factor_level/ms$error
F_value_baseline = ms$baseline/ms$subject_level

# Set p-values of F distribution
p_factor = 1 - pf(F_value_factor, dof$factor_level, dof$error)
p_baseline = 1 - pf(F_value_baseline, dof$baseline, dof$subject_level)
```

Last, we create the output table for the Repeated Measures ANOVA. The table contains the sum of squares, the degrees of freedom, the mean squares, the F-value as well as the p-value for each of the components. By running the funtion rma, this output table is always printed. The Repeated Measures ANOVA-table for the simulated data used in our analysis is given by table 1 on page 8. It can be seen, that the F-tests are highly significant. The baseline-F-test hereby tests, wheter the overall mean is equal to zero, which can be rejected in our case. The Factor-F-test then tests the hypotheses outlined in Section 3.1. Hence, we can reject the H_0, stating that the factor level means are all equal.

```
# Create the output table for rmANOVA

# Specify source variable
source = c("Baseline", "Factor", "Subject", "Error", "Total", "Corrected
    total")

# Create table
```

```
ANOVA_table = data.frame(check.names = FALSE, Source = source, `Sum of
    squares` = c(ss %>% select(2:5, 1) %>% unlist(), corrected_sst), `Degrees
    of freedom` = c(dof %>% select(2:5, 1) %>% unlist(), (n * k) - 1), `Mean
    squares` = c(ms %>% select(2:5) %>% unlist(), NA, variance), `F-value` =
    c(F_value_baseline, F_value_factor, NA, NA, NA, NA), `p-value` =
    c(p_baseline, p_factor, NA, NA, NA, NA))

rownames(ANOVA_table) = NULL
```

	Source	Sum of squares	Degrees of freedom	Mean squares	F-value	p-value
1	Baseline	1769610.20	1	1769610.20	1430.36	<0.001
2	Factor	174706.07	4	43676.52	142.23	<0.001
3	Subject	33403.82	27	1237.18		
4	Error	33166.05	108	307.09		
5	Total	2010886.14	140			
6	Corrected total	241275.94	139	1735.80		

Table 1: Repeated Measures ANOVA-table for our simulated data.

4 SSE Reduction with the Repeated Measures ANOVA

The Repeated Measures ANOVA is closely related to the ANOVA, however used for dependent data. Hence, in the following section, we want to take a closer look at the differences between both models. In order to do so, we compute an ANOVA model and compare its error terms with the ones of the computed Repeated Measures ANOVA model (section 3.2). Such comparison is only made for the purpose of demonstrating the model differences. Since ANOVA requires independent data, it would give us misleading results when computed for of our simulated data.

4.1 Estimation of our ANOVA model

In order to enable a comparison of the error terms between ANOVA and Repeated Measures ANOVA, we computed the (one-way) ANOVA model. The computational steps are mainly the same as for the Repeated Measures ANOVA (cp. section 3.2). Thus, only the main differences shall be discussed in more detail in this section. The ANOVA computation is part of the following function:

```
ow_a(rma_data, id)
```

Again, the first argument specifies the data, whereby we use our simulated data. The second argument then specifies the number of independent variables. When running the function, first the needed variables are defined, then the basic ANOVA components are computed almost similarly to the ones of the Repeated Measures ANOVA. However, no subject component is computed. As a consequence, we calculate the error component by subtracting only the baseline component and

the factor level component from the values of the dependent variable. Large differences between subjects, therefore lead to a larger error component.

```
# Define basic ANOVA components
grand_mean = mean(dependent_variable)
baseline_components = matrix(grand_mean, nrow = n_group, ncol = k)

conditional_means = colMeans(dependent_variable)
factor_level_components = matrix(conditional_means - grand_mean, nrow =
    n_group, ncol = k, byrow = TRUE)

error_components_ANOVA = dependent_variable - baseline_components -
    factor_level_components
```

In the next steps, we construct the decomposition matrix, a matrix of k*n rows and four columns, and compute the sums of squares for the components. Next, we define the degrees of freedom. Importantly, the degrees of freedom for the error component is calculated differently than for the error component in the Repeated Measures ANOVA:

```
# Set degrees of freedom
dof_ANOVA = data.frame(dependent_variable = n, baseline = 1, factor_level =
    (k - 1), error = (n - k))
```

Similarly to the computation of the Repeated Measures ANOVA, we then compute the mean squares, calculate the F-values, set the p-values of the F-distribution and compute the corrected total sum of squares, which gives the variance of the dependent variable. Lastly, we create the ANOVA output table. The output table for our simulated data is given by table 2 on page 9.

In comparison with the Repeated Measures Anova table on page 8, it can be seen, that no subject component is defined. The sum of squares error for the ANOVA is therefore larger by the amount of the sum of squares subject in the Repeated Measures ANOVA. The subsequent subsection will take a closer look at this aspect.

	Source	Sum of squares	Degrees of freedom	Mean squares	F-value	p-value
1	Baseline	1769610.20	1.00	1769610.20	3588.67	0.00
2	Factor	174706.07	4.00	43676.52	88.57	0.00
3	Error	66569.87	135.00	493.11		
4	Total	2010886.14	140.00			
5	Corrected total	241275.94	139.00	1735.80		

Table 2: ANOVA-table for our simulated data.

4.2 Comparing the Error Terms

In order to compare the error terms of the ANOVA model and the Repeated Measures ANOVA model, it is helpful to quantify and visualize the difference. Both is made by the function rma_sse_reduct. The function ow_a, containing the computation of the ANOVA model, is also integrated into this function for model comparison:

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

The first argument of the function refers to the data that is used. Like in the foregoing functions, the argument id specifies the number of independent variables. Furthermore, the function rma_sse_reduct specifies a pie-chart and a bar-chart, that graphically illustrate the reduction of sum of squares error. By the argument plot_type, a user can decide for the plot that shall be printed. The last argument return_anova_table specifies whether a full ANOVA table similar to table 2 on page 9 shall be printed. A small comparison table, stating the percentage in sum of squares reduction, is always printed when running the function.

For preparing the model comparison, the function rma_sse_reduct applies the ANOVA function as well as the Repeated Measures ANOVA function to our simulated data. It then cuts out the sum of squares error and the sum of squares subject from the output tables for each model. Importantly, the sum of squares subject is always set to zero for the ANOVA, since it is not considered in the model calculation.

```
# Preparing the components for model comparison
ow_a_results = ow_a(rma_data, id)[[1]]
rma_results = rma(rma_data, id)[[1]]

sse_anova = ow_a_results[3, 2]
ss_subject_anova = 0

sse_rma = rma_results[4, 2]
ss_subject_rma = rma_results[3, 2]
```

Furthermore, we define variables, that we need the comparison plot displaying the reduction of sum of squares error. The variable var contains the sum of squares error by subject and error. The variable model is defined to assign the values in the plots. The variable source is required for color and legend label assignment in the plots. All three variables are then merged into one data frame. Also, we define a variable var_percent giving the percentage of sum of squares error reduction for better readability in the piechart.

```
# Define variables for the comparison plot
var = c(sse_anova, ss_subject_anova, sse_rma, ss_subject_rma)
```

```
model = rep(c("No estimation of the\nvariation between subjects", "Estimation
    of the\nvariation between subjects"), each = 2)
source = factor(rep(c("Error", "Entity"), times = 2), levels = c("Entity",
    "Error"))
comparison_data = data.frame(var, model, source)

comparison_data$var_percent = comparison_data$var *
    100/(max(comparison_data\$var))
```

Two plots are created, visualizing the reduction of sum of squares error. For that, we take use of the package ggplot2, which allows us to create more visually appealing plots.

```
# Create stacked barplot
comp_plot_bar = ggplot(comparison_data, aes(model, var_percent, fill =
    source))
        + geom_bar(stat = "identity")
        + labs(x = "Model", y = "Sum of squares (error)", title = "Reduction
            of sum of squared errors (SSE)")
        + guides(fill = guide_legend(title = NULL))
        + scale_fill_manual(values = c("orange", "navyblue"))
        + theme_bw()
        + theme(panel.grid.major = element_blank(), panel.grid.minor =
            element_blank(), panel.background = element_blank())
        + theme_bw()
        + theme(panel.grid.major = element_blank(), panel.grid.minor =
            element_blank(), legend.key = element_rect(colour = "black"),
            plot.title = element_text(face="bold", hjust = .5))

# Create pie chart
comp_plot_pie = ggplot(comparison_data, aes(x = "", y = var_percent, fill =
    source))
        + geom_bar(width = 1, stat = "identity")
        + labs(x = "", y = "", title = "Reduction of sum of squared errors
            (SSE) in percent")
        + guides(fill = guide_legend(title = NULL)) + scale_fill_manual(values
            = c("orange", "navyblue"))
        + coord_polar(theta = "y")
        + facet_grid(~model)
        + theme_bw()
        + theme(panel.grid.major = element_blank(), panel.grid.minor =
            element_blank(), panel.background = element_blank())
        + theme_bw()
        + theme(panel.grid.major = element_blank(), panel.grid.minor =
            element_blank(), legend.key = element_rect(colour = "black"),
            plot.title = element_text(face="bold", hjust = .5))
```

Subsequently, we define a condition choosing the plot, that is specified in the function rma_sse_-reduct. Furthermore, we create a comparison table that gives the percentage share of sum of squares by subject and error. This table is always printed, when running the function.

```
# Selection of plot design
if (plot_type == "pie") {
       final_plot = comp_plot_pie
} else {
       final_plot = comp_plot_bar
}

# Create comparison table
%error_ss_comparison_table = data.frame(check.names = FALSE, ‘ ‘ = c("Error",
    "Entity"), ‘Sum of squares‘ = c(sse_rma, ss_subject_rma), ‘Percentage
    share‘ =
    paste(as.character(round(comparison_data$var_percent[3:4])),c("%", "%"),
    sep = ""))
rownames(error_ss_comparison_table) = NULL
```

Finally, we specify a warning message, that is printed when an ANOVA output table is ordered by setting the argument return_anova_table. Computing the ANOVA neglects the dependency in the data and gives misleading results. The warning message shall therefore remind the user, that the computation of the ANOVA is only made for illustration purposes.

In the case of our simulated data, the piechart is given by Figure **??** on page **??**.[Grafik standardisiert einfÃijgen]. The piechart on the right hand side shows the percentage share of the error term, when no sum of squares subject is calculated. In the case of an ANOVA, this would be our error term. The pie chart on the left hand side shows that the percentage share of the sum of squares subject is almost as large as the percentage share of the sum of squares error. This means, that when calculating the sum of squares subject separately, as we do in the Repeated Measures ANOVA model, we obtain an error term, that is reduced by its half. Such a reduction allows us to compute a much more powerful F-Test, which can be seen by comparing the F-values given in the output tables (table 1 on page 8 and table 2 on page **??**). The reduction of the error term can obviously be of different sizes for different data-sets. The more variance between subjects in the data, the larger the reduction. Since we use a larger degree of freedom for the error component in the Repeated Measures ANOVA than in the ANOVA, there can be extreme cases, where we obtain a larger error component by the Repeated Measures ANOVA model. However, this will usually not be the case.

5 Confidence Intervals(CI)

Short theoretical introduction

5.1 Unadjusted CIs

Hier dann noch mal ein anderes Zitat (**?**).

5.2 Adjusted CIs

5.3 Plotting the CIs

6 Sphericity

Short theoretical introduction

6.1 Test for Sphericity

Hier dann noch mal ein anderes Zitat (**?**).

6.2 Adjustment for Sphericity

7 Orthogonal polynomial contrasts

Short theoretical introduction

7.1 Computing the Orthogonal Polynomial Contrasts

Hier dann noch mal ein anderes Zitat (**?**).

7.2 Plotting the Contrasts

Appendix

Literaturverzeichnis