Orthogonal Polynomial Contrasts

After the conduction of a repeated measures ANOVA we obtain an omnibus F-Value and thus a general idea, whether the factor indeed affects the depended variable. However, it remains unclear what this dependency looks like in particular. To evaluate the impact of the different factor levels on the depended variable in further detail, one might conduct post hock t-tests between each pair of factor levels or even test specific a priori hypothesis by the implementation of complex contrast analyses (see e.g. Howell, 2010; Steavens, 2009). Since the factor levels of a repeated measures ANOVA most commonly possess an interval scale of measurement (or are at least believed to be approximately equidistant ordinal scaled), there is an additional method witch should be considered. This method is known as orthogonal polynomial contrasts or trend analysis. As the name implies, this method detects orthogonal polynomial trend components in the Effects of the factor. If there are **k** factor levels it decomposes the omnibus factor effect into a full set of **k - 1** orthogonal contrasts. These contrast all detect a specific polynomial trend of all orders from **1** to **k - 1** and test if these trends contribute significantly to the overall effect of the factor. This allows testing specific hypothesis regarding separate trends in the depended variable due to manipulations of the factor level.

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

The above function conducts this type of analysis. As input parameter it needs a repeated measures data file which meets the above described requirements. It also contains the **id = 1** and **print_plot = TRUE** Arguments. Since at is always possible to describe the data perfect with a polynom of the order **k - 1** and hence this is the maximal order of a polynomial trend used for this analysis we define the **maxpoly** variable. Certainly it is conceivable that not all **k - 1** orthogonal polynomial trends are of interest, so the function contains the additional argument **maxpoly = NA**. If a Value smaller than **k - 1** is inserted the function only computes and tests the contrasts up to the polynomial trend of the same order.

```
# maximal polynomial degree for orthogonal polynomials
if((maxpoly > k - 1) | (is.na(maxpoly))){
        maxpoly = k - 1}
```

At first it is necessary to calculate for each of the requested orthogonal trend components a so called linear contrast. A linear contrast is basically the sum of the weighted dependent variable values of each individual factor level. To obtain linear contrasts which represent all orthogonal components of **k - 1** possible polynomial trends, the weights are generated by using **contr.poly()**.

```
# Defining Contrast weights for orthogonal polynomial contrasts
contrast_weights = (t(contr.poly(k)))[1:maxpoly, ]
```

It should be noted, that it needs **maxpoly** times **k** contrast weights in total for this type of analysis. In the next step the weighting factors are applied and the linear contrasts for each specific trend are computed individually for each subject

$$L_S = \sum_{i=1}^{k} c_i Y_{i\,S}\,.$$

```
# Applying formula for linear contrasts
weighted_dependend_variables =
        dependent_variable[rep(1:n, each = maxpoly), ] * (contrast_weights)[rep(1:maxpoly, n), ]
linear_subject_contrasts = matrix(rowSums(weighted_dependend_variables), byrow = TRUE,
        ncol = maxpoly)
```

Following this step there are linear contrasts for each individual entity in the study, so it is necessary to construct an aggregated score to draw further conclusions. This is accomplished by taking the mean of al linear contrasts for one specific trend component to obtain the **contrast_estimator**

$$\bar{L} = \sum_{k=1}^{n} L_k \, .$$

```
# Computing contrast estimators for each orthogonal polynomial contrast as well as standard
errors for thees estimators
contrast_estimator = colMeans(linear_subject_contrasts)
contrast_se = sqrt(apply(linear_subject_contrasts,2,var)) / sqrt(n)
```

The standard error of this contrast estimator **contrast_se** is computed as well for inference reasons. This allows to test whether this contrast score deviates from 0 significantly, which is accomplished via a t-test.

```
# Computing t-values for each contrast
contrast_t_values = contrast_estimator / contrast_se
#contrast_F_values = contrast_t_values^2
```

```
# Computing the corresponding p-values
contrast_p_values = 1 - pt(abs(contrast_t_values), n-1)
```

If the individual trend components contribute significantly to the factor effect, it might be of interests how much a certain trend contributed to the effect of the factor. Therefore the sum of squares component which is explained by the factor is decomposed into **maxpoly** trend component sums of squares by computing the individual trend sum of squares

$$SS_L = \frac{n \times \bar{L}^2}{\sum_{i=1}^{k} c_i^2} \, .$$

This is possible due to the fact, that the contrasts are orthogonal to each other, so they don't have redundant information. The proportion of a specific trend sum of square to the factor sum of square marks the percentage in contribution of this particular trend component to the whole factor effect

$$\sum_{j=1}^{k-1} SS_{L(orth.)_j} = SS_{Factor} \, .$$

```
# Computing sums of squares for each contrast
contrast_ss = n * contrast_estimator^2 / rowSums(contrast_weights^2)

# Computing amount of the variance in the dependent variable explained by the factor
# which in turn can be explained by a cerain orthogonal polynomial trend (ss_trend / ss_factor)
proportional_trend_contribution = contrast_ss / rep(sum(rep((Flm - Gm)^2, each = n)), maxpoly)
```

The results of the computational steps described above are summarized in a table, which is later returned by the function.

```
# define source variable
source = rownames(contrast_weights)
contrast_table = data.frame(check.names = FALSE, "Source" = source, "Sum of squares" =
        contrast_ss, "Proportional contribution to the factor effect" =
        proportional_trend_contribution, "Contrast estimator" = contrast_estimator,
        "Standard error" = contrast_se, "Degrees of freedom" = rep((n - 1), maxpoly), "t-value" =
        contrast_t_values, "p-value" = contrast_p_values)
rownames(contrast_table) = NULL
```

To display the polynomial trends in the data, polynomial regressions were used. The results actually don’t represent orthogonal trends but rather each trend line of a certain polynomial degree also includes all lower order polynomial trend contributions.

… Niko’s Part

The function will eventually create the plot and return the table with the results of the contrast analysis.

```
if (print_plot == TRUE){
        print(poly_plot)}

return(list(contrast_table = contrast_table, poly_plot = poly_plot))
```

Literature

Howell, D. C. (2010). *Statistical Methods for Psychology* (5th ed.). Belmont: Wads worth, Cengage Learning.

Stevens, J. P. (2009). *Applied multivariate statistics for the social sciences* (5th ed.). New York: Routledge.