M21_202303_aov_rt

Joanna Morris

2023-03-20

Compute PCA

This script computes separate ANOVAs for simple and complex non-words.

Following Andrews and Lo (2013) this script computes a PCA for our spelling and vocabulary measures. Because the standardised spelling and vocabulary scores were correlated, to facilitate interpretation, two orthogonal measures of individual differences were derived from a principal components analysis. Analysis based on this tutorial

```
library(readr)
library(dplyr)
library(datawizard)
sv_202303 <- read_csv("m21_spell_vocab_raw.csv")
sv_202303.na <- na.omit(sv_202303)
sv_202303.na <- mutate(sv_202303.na,
                       z_ART = standardise(ART_correct),
                       z_vocab = standardise(vocab_correct),
                       z_spell = standardise(spell_correct))

cor.test(sv_202303.na$z_vocab, sv_202303.na$z_spell)
```

```

    Pearson's product-moment correlation

data:  sv_202303.na$z_vocab and sv_202303.na$z_spell
t = 1.9352, df = 61, p-value = 0.05761
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.007724833  0.460807138
sample estimates:
      cor
0.2405005
```

By default, the function `PCA()` in `FactoMineR`, standardizes the data automatically during the PCA; so you don't need do this transformation before the PCA.

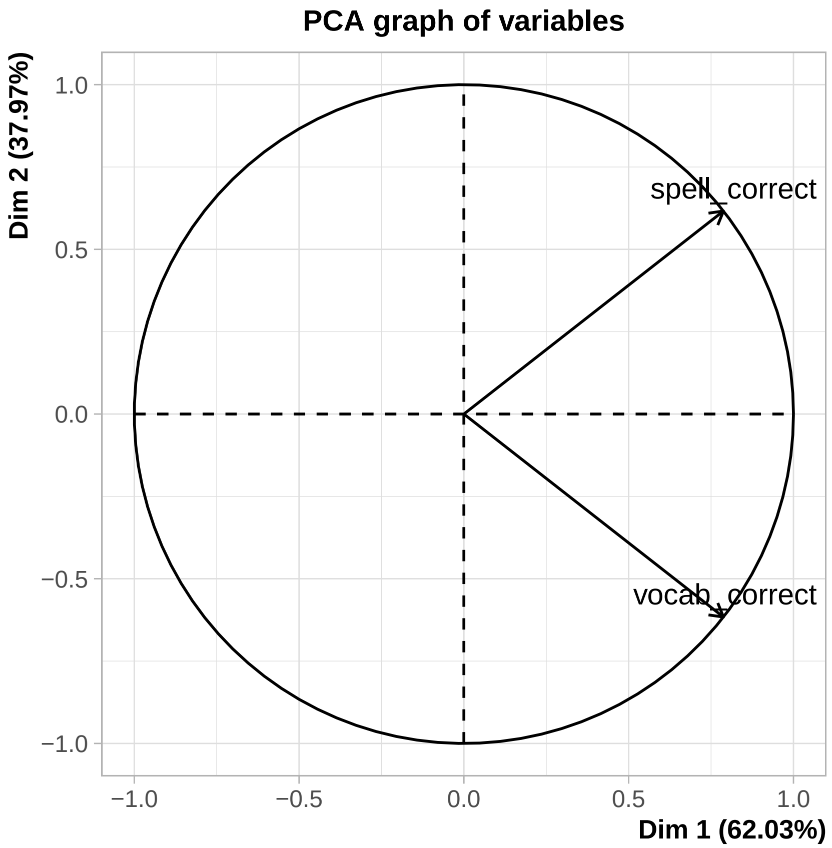
- `X`: a data frame. Rows are individuals and columns are numeric variables
- `scale.unit`: a logical value. If TRUE, the data are scaled to unit variance before the analysis. This standardization to the same scale avoids some variables to become dominant just because of their large measurement units. It makes variable comparable.

- `ncp`: number of dimensions kept in the final results.
- `graph`: a logical value. If TRUE a graph is displayed.

The plot shows the relationships between all variables. It can be interpreted as follow:

- Positively correlated variables are grouped together.
- Negatively correlated variables are positioned on opposite sides of the plot origin (opposed quadrants).
- The distance between variables and the origin measures the quality of the variables on the factor map. Variables that are away from the origin are well represented on the factor map.

```
library(FactoMineR)
library(factoextra)

res.pca <- PCA(sv_202303.na[,3:4], scale.unit = TRUE, ncp = 2, graph = FALSE)
plot(res.pca, choix = "varcor", graph.type = c("ggplot"))
```



The eigenvalues measure the amount of variation retained by each principal component. Eigenvalues are large for the first PCs and small for the subsequent PCs. That is, the first PCs corresponds to the directions with the maximum amount of variation in the data set.

We examine the eigenvalues to determine the number of principal components to be considered

```
(eig.val <- get_eigenvalue(res.pca))
```

```
      eigenvalue variance.percent cumulative.variance.percent
Dim.1  1.2405005         62.02503                    62.02503
Dim.2  0.7594995         37.97497                   100.00000
```

The quality of representation of the variables on factor map is called cos2 (square cosine, squared coordinates). A high cos2 indicates a good representation of the variable on the principal component. In this case the variable is positioned close to the circumference of the correlation circle. A low cos2 indicates that the variable is not perfectly represented by the PCs. In this case the variable is close to the center of the circle. For a given variable, the sum of the cos2 on all the principal components is equal to one. If a variable is perfectly represented by only two principal components (Dim.1 & Dim.2), the sum of the cos2 on these two PCs is equal to one. In this case the variables will be positioned on the circle of correlations.

```
res.pca$var$cos2
```

```
                  Dim.1     Dim.2
vocab_correct 0.6202503 0.3797497
spell_correct 0.6202503 0.3797497
```

The contributions of variables in accounting for the variability in a given principal component are expressed in percentage. Variables that are correlated with PC1 (i.e., Dim.1) and PC2 (i.e., Dim.2) are the most important in explaining the variability in the data set. Variables that do not correlated with any PC or correlated with the last dimensions are variables with low contribution and might be removed to simplify the overall analysis.

```
res.pca$var$contrib
```

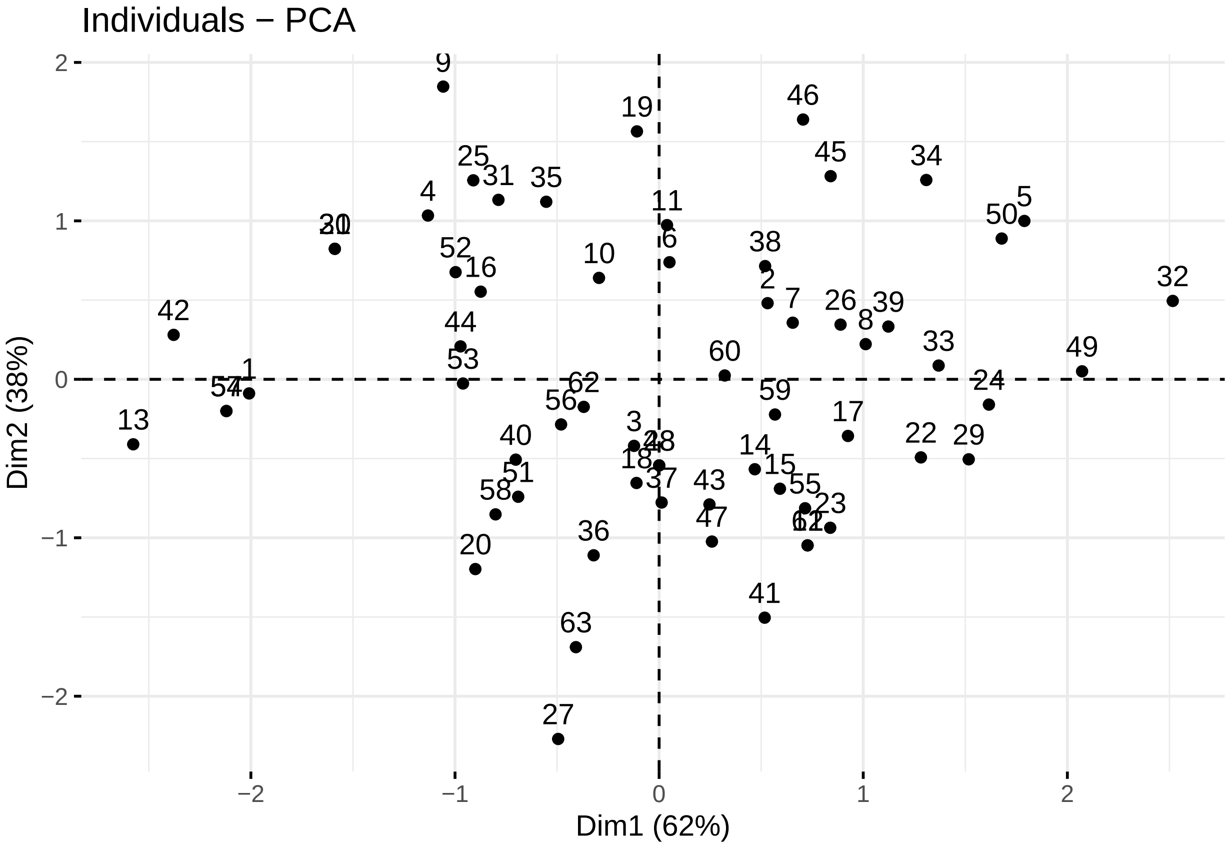
```
              Dim.1 Dim.2
vocab_correct    50    50
spell_correct    50    50
```

```
(res.desc <- dimdesc(res.pca, axes = c(1,2), proba = 0.05))
```

```
$Dim.1

Link between the variable and the continuous variables (R-square)
=================================================================================
              correlation      p.value
vocab_correct   0.7875597 1.913196e-14
spell_correct   0.7875597 1.913196e-14

$Dim.2

Link between the variable and the continuous variables (R-square)
=================================================================================
              correlation      p.value
spell_correct   0.6162384 7.592937e-08
vocab_correct  -0.6162384 7.592937e-08
```

The fviz_pca_ind() is used to produce the graph of individuals.

```
fviz_pca_ind(res.pca)
```



```
sv_202303.na<-bind_cols(sv_202303.na,res.pca$ind$coord)
```

Loads RT data and join to PCA dataset

```
cw_frq <- read_csv("CW_frq.csv")
nw_frq <- read_csv("NW_frq.csv")


CW_rt <- read_csv("CW_rt_2.csv")
CW_rt$cw_target <- NULL
CW_rt <- rename(CW_rt, cw_target = target_lower)

NW_rt <- read_csv("NW_rt_2.csv")
NW_rt$nw_target <- NULL
NW_rt <- rename(NW_rt, nw_target = target_lower)

cw_rt_pca <- inner_join(sv_202303.na, CW_rt, by = "SubjID")  #join subject PCA data
nw_rt_pca <- inner_join(sv_202303.na, NW_rt, by = "SubjID")

cw <- left_join(cw_rt_pca, cw_frq, by = c("cw_target")) #join word frequency data
nw <- left_join(nw_rt_pca, nw_frq, by = c("nw_target"))
```

Divide participants based on median split of Dim2. Higher values on this factor indicate that spelling scores were relatively higher than vocabulary,

```
cw.median <- median(cw$Dim.2)
cw <- cw |>
    mutate(lang_type = case_when(
      Dim.2 < cw.median ~ "Semantic",
      Dim.2 >= cw.median ~ "Orthographic"
    ))

nw.median <- median(nw$Dim.2)
nw <- nw |>
    mutate(lang_type = case_when(
      Dim.2 < nw.median ~ "Semantic",
      Dim.2 >= nw.median ~ "Orthographic"
    ))
```

Export list of subjects with Reading type classification

```
df <- distinct(select(cw, SubjID, lang_type ))
df_o <- filter(df, lang_type == "Orthographic")
df_s <- filter(df, lang_type != "Orthographic")
write_csv(df, "m21_subjlist_reading_type.csv")
write_csv(df_o, "m21_subjlist_ortho.csv")
write_csv(df_s, "m21_subjlist_semant.csv")
```

```
cols <- c( "cw_famsize", "lang_type")  # recode ind variable columns as factors
cw <- cw |> mutate_at(cols, factor)
cw$cw_famsize <- recode_factor(cw$cw_famsize, S = "Small", L = "Large")


cols <- c( "cw_famsize", "lang_type")
cw <- cw |> mutate_at(cols, factor)
cw$cw_famsize <- recode_factor(cw$cw_famsize, S = "Small", L = "Large")


cols <- c( "nw_famsize", "lang_type", "complexity")
nw <- nw |> mutate_at(cols, factor)
nw$nw_famsize <- recode_factor(nw$nw_famsize, S = "Small", L = "Large")
nw$complexity <- recode_factor(nw$complexity, SIMP = "Simple", COMP = "Complex")


nw_smpl <- filter(nw, complexity == "Simple")
nw_smpl$complexity <- NULL

nw_cplx <- filter(nw, complexity == "Complex")
nw_cplx$complexity <- NULL

rm(CW_rt)  #remove original rt file after joining neuropsych data
rm(NW_rt)
```

Removes rts for errors (column rt.err) and then imputes missing values with the mean for the dataset (column “rt.err.imp”) then creates a new column with inverse RTs

```
library(tidyr)
cw <- cw |> mutate(rt.err = response_time * correct)  # convert error rts to 0
cw <- cw |> mutate(rt.err = na_if(rt.err, 0))         # convert 0 rts to NA
cw.mean <- mean(cw$rt.err, na.rm = TRUE)  # get mean rt excluding errors
cw <- cw |> mutate(rt.err.imp = ifelse(is.na(rt.err),
                                       cw.mean,
                                       rt.err))  # replace missing values with mean
cw <- cw |> mutate(inv.rt = 1/rt.err.imp)  # creates new column with inverse RTs


nw_smpl <- nw_smpl |> mutate(rt.err = response_time * correct)  # convert error rts to 0
nw_smpl <- nw_smpl |> mutate(rt.err = na_if(rt.err, 0))         # convert 0 rts to NA
nw_smpl.mean <- mean(nw_smpl$rt.err, na.rm = TRUE)  # get mean rt excluding errors
nw_smpl <- nw_smpl |> mutate(rt.err.imp = ifelse(is.na(rt.err),
                                                 nw_smpl.mean,
                                                 rt.err))  # replace missing values with mean
nw_smpl <- nw_smpl |> mutate(inv.rt = 1/rt.err.imp)  # creates new column with inverse RTs

nw_cplx <- nw_cplx |> mutate(rt.err = response_time * correct)  # convert error rts to 0
nw_cplx <- nw_cplx |> mutate(rt.err = na_if(rt.err, 0))         # convert 0 rts to NA
nw_cplx.mean <- mean(nw_cplx$rt.err, na.rm = TRUE)  # get mean rt excluding errors
nw_cplx <- nw_cplx |> mutate(rt.err.imp = ifelse(is.na(rt.err),
                                                 nw_smpl.mean,
                                                 rt.err))  # replace missing values with mean
nw_cplx <- nw_cplx |> mutate(inv.rt = 1/rt.err.imp)  # creates new column with inverse RTs
```

Determines how much missing data there is. Creates new dataframe with just the non-missing data

```
cw_missing_data<- filter(cw, is.na(cw$rt.err))
(xtab.missing.data <- xtabs(~cw_famsize+lang_type, data=cw_missing_data))
```

```
          lang_type
cw_famsize Orthographic Semantic
     Small          213      169
     Large          225      163
```

```
nw.smpl_missing_data<- filter(nw_smpl, is.na(nw_smpl$rt.err))
(xtab.missing.data <- xtabs(~nw_famsize+lang_type, data=nw.smpl_missing_data))
```

```
          lang_type
nw_famsize Orthographic Semantic
     Small          207      212
     Large          215      257
```

```
nw.cplx_missing_data<- filter(nw_cplx, is.na(nw_cplx$rt.err))
(xtab.missing.data <- xtabs(~nw_famsize+lang_type, data=nw.cplx_missing_data))
```

```
          lang_type
nw_famsize Orthographic Semantic
     Small          223      229
     Large          245      252
```

Get condition means

```
#Define standard error of the mean function

sem <- function(x) sd(x)/sqrt(length(x))

(cw.cond.means <- cw |>
   group_by(cw_famsize, lang_type) |>
   summarise(mean = mean(rt.err.imp),
             se = sem(rt.err.imp),
             num_stim = n()))
```

```
# A tibble: 4 x 5
# Groups:   cw_famsize [2]
  cw_famsize lang_type     mean    se num_stim
  <fct>      <fct>        <dbl> <dbl>    <int>
1 Small      Orthographic  591.  3.20     1500
2 Small      Semantic      635.  3.19     1456
3 Large      Orthographic  578.  3.21     1500
4 Large      Semantic      616.  3.04     1465
```

```
(nw_smpl.cond.means <- nw_smpl |>
    group_by(nw_famsize, lang_type) |>
    summarise(mean = mean(rt.err.imp),
              se = sem(rt.err.imp),
              num_stim = n()))
```

```
# A tibble: 4 x 5
# Groups:   nw_famsize [2]
  nw_famsize lang_type     mean    se num_stim
  <fct>      <fct>        <dbl> <dbl>    <int>
1 Small      Orthographic  686.  4.02      750
2 Small      Semantic      720.  4.06      730
3 Large      Orthographic  690.  3.95      750
4 Large      Semantic      726.  3.66      732
```

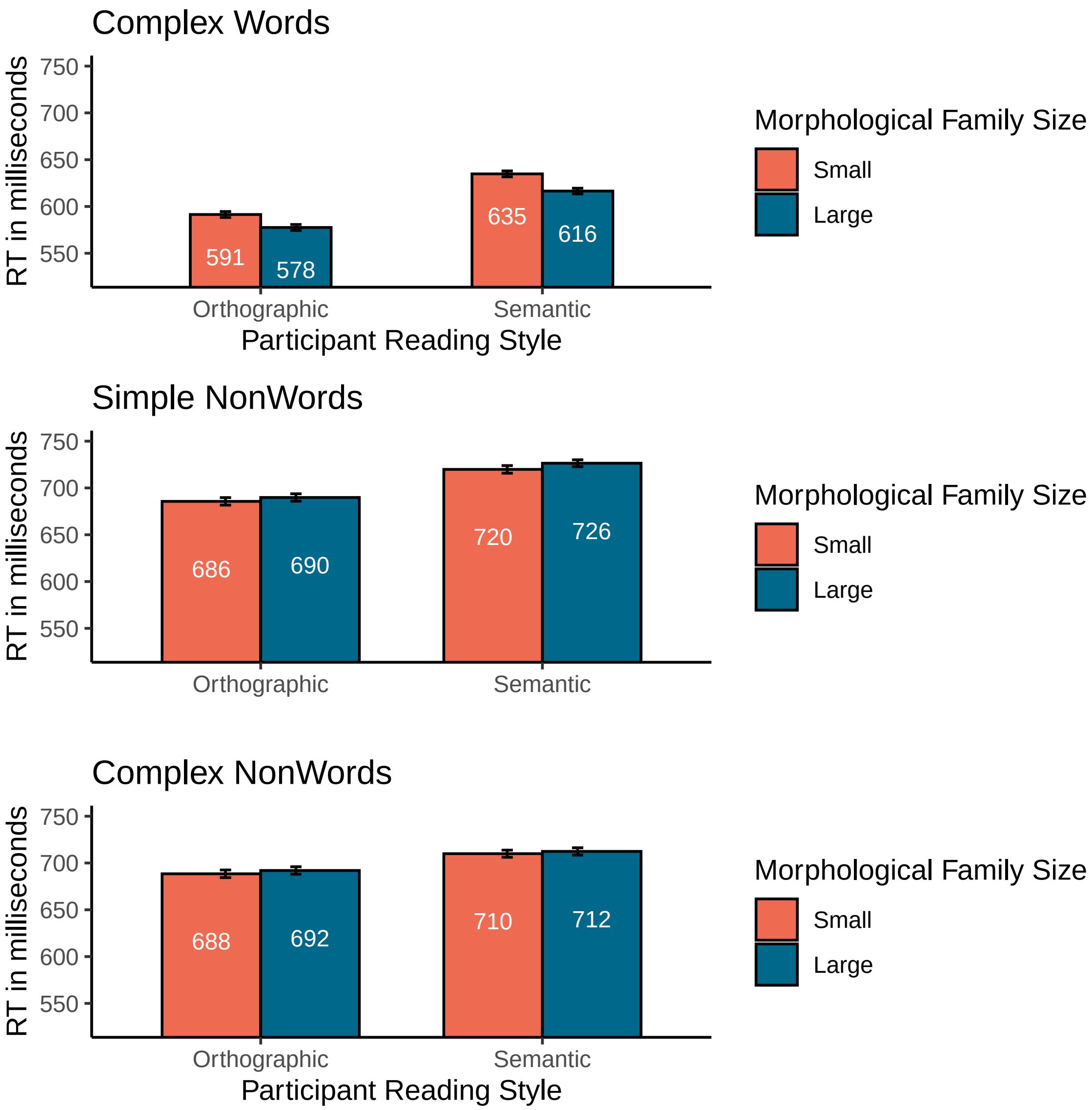
```
(nw_cplx.cond.means <- nw_cplx |>
    group_by(nw_famsize, lang_type) |>
    summarise(mean = mean(rt.err.imp),
              se = sem(rt.err.imp),
              num_stim = n()))
```

```
# A tibble: 4 x 5
# Groups:   nw_famsize [2]
  nw_famsize lang_type     mean    se num_stim
  <fct>      <fct>        <dbl> <dbl>    <int>
1 Small      Orthographic  688.  4.09      750
2 Small      Semantic      710.  3.83      736
3 Large      Orthographic  692.  4.01      750
4 Large      Semantic      712.  3.90      737
```

Barplots

```
library(gridExtra)
p1 <-  cw.cond.means %>% ggplot(aes(x=lang_type,
                                    y=mean,
                                    fill = cw_famsize,
                                    ymin = mean - se,
                                    ymax = mean + se)) +
  coord_cartesian(xlim = NULL,
                  ylim = c(525, 750),
                  expand = TRUE,
                  default = FALSE,
                  clip = "on") +
  geom_col(position = "dodge", width = 0.5, color = "black")  +
  ylab("RT in milliseconds")  +
  xlab("Participant Reading Style")  +
  ggtitle("Complex Words") +
  scale_fill_manual(values = c("coral2", "deepskyblue4"))+
  geom_errorbar(width = .08, position = position_dodge(0.5)) +
  theme_classic() +
   geom_text(aes(label = round(mean, digits = 0)),
             colour = "white",
             size = 3,
             vjust = 3,
             position = position_dodge(.5))+
  guides(fill=guide_legend(title="Morphological Family Size"))

p2 <-  nw_smpl.cond.means %>% ggplot(aes(x=lang_type,
                                         y=mean,
                                         fill = nw_famsize,
                                         ymin = mean - se,
                                         ymax = mean + se)) +
  coord_cartesian(xlim = NULL, ylim = c(525, 750),
                  expand = TRUE,
                  default = FALSE,
                  clip = "on") +
  geom_col(position = "dodge", width = .7, color = "black")  +
  xlab(" ")  +
  ylab("RT in milliseconds")  +
  ggtitle("Simple NonWords") +
  scale_fill_manual(values = c("coral2", "deepskyblue4"))+
  geom_errorbar(width = .08, position = position_dodge(0.5)) +
  theme_classic() +
  geom_text(aes(label = round(mean, digits = 0)),
             colour = "white",
             size = 3,
             vjust = 4.5,
            position = position_dodge(.7)) +
  guides(fill=guide_legend(title="Morphological Family Size"))

p3 <-  nw_cplx.cond.means %>% ggplot(aes(x=lang_type,
                                         y=mean,
                                         fill = nw_famsize,
                                         ymin = mean - se,
                                         ymax = mean + se)) +
```

```r
  coord_cartesian(xlim = NULL, ylim = c(525, 750),
                  expand = TRUE,
                  default = FALSE,
                  clip = "on") +
  geom_col(position = "dodge", width = .7, color = "black")  +
  xlab("Participant Reading Style")  +
  ylab("RT in milliseconds")  +
  ggtitle("Complex NonWords") +
  scale_fill_manual(values = c("coral2", "deepskyblue4"))+
  geom_errorbar(width = .08, position = position_dodge(0.5)) +
  theme_classic() +
  geom_text(aes(label = round(mean, digits = 0)),
             colour = "white",
             size = 3,
             vjust = 4.5,
            position = position_dodge(.7)) +
  guides(fill=guide_legend(title="Morphological Family Size"))

grid.arrange(p1, p2, p3)
```



```
# grid.arrange(p2, p3)
```

```
library(ez)

ezANOVA(data = cw,
        dv = rt.err.imp,
        wid = SubjID,
        within = .(cw_famsize),
        within_full = .(cw_famsize, cw_target),
        between = lang_type,
        type = 3)
```

```
Warning: Converting "SubjID" to factor for ANOVA.

Warning: Collapsing data to cell means first using variables supplied to
"within_full", then collapsing the resulting means to means for the cells
supplied to "within".
```

```
$ANOVA
                Effect DFn DFd          F            p p<.05          ges
2            lang_type   1  58  6.8551395 1.125713e-02     * 0.1024825208
3           cw_famsize   1  58 29.1533842 1.299937e-06     * 0.0167579493
4 lang_type:cw_famsize   1  58  0.5478038 4.622020e-01       0.0003201529
```

```
ezANOVA(data = nw_cplx,
        dv = rt.err.imp,
        wid = SubjID,
        within = .(nw_famsize),
        within_full = .(nw_famsize, nw_target),
        between = lang_type,
        type = 3)
```

```
Warning: Converting "SubjID" to factor for ANOVA.

Warning: Collapsing data to cell means first using variables supplied to
"within_full", then collapsing the resulting means to means for the cells
supplied to "within".

$ANOVA
                Effect DFn DFd          F         p p<.05          ges
2            lang_type   1  58 2.14786827 0.1481685       0.0337835603
3           nw_famsize   1  58 0.57142520 0.4527524       0.0005497191
4 lang_type:nw_famsize   1  58 0.07260026 0.7885440       0.0000698760
```

```
ezANOVA(data = nw_smpl,
        dv = rt.err.imp,
        wid = SubjID,
        within = .(nw_famsize),
        within_full = .(nw_famsize, nw_target),
        between = lang_type,
        type = 3)
```

```
Warning: Converting "SubjID" to factor for ANOVA.

Warning: Collapsing data to cell means first using variables supplied to
"within_full", then collapsing the resulting means to means for the cells
supplied to "within".

$ANOVA
                Effect DFn DFd        F           p p<.05          ges
2            lang_type   1  58 7.197773 0.009494787     * 0.1018639868
3           nw_famsize   1  58 2.105327 0.152174207       0.0031148459
4 lang_type:nw_famsize   1  58 0.171281 0.680502319       0.0002541385
```