Types of Sums of Squares in R

Introduction

In unbalanced factorial designs, there is no clear decompositon of the total sum of square into parts explained by each factor. Commercial software has introduced a taxonomy of four types of possible decomposition methods. Say that one has a model with two factors A and B and the interaction. Let SSE denote the sum of squared errors (the residuals of the model). Then the four types are given as follows.

1. Type I calculates the decrease in SSE as the factors are added in the order the user specified them. In the model `y ~ A * B`, the sum of squares of A is the difference between the model with only an intercept and the model with factor A. The sum of squares of B is the difference between the model with factor A and the model with factors A and B. Finally, the sum of squares of the interaction is the difference in SSE between the model with factors A and B, and the model with A, B, and the interaction.
2. Type II calculates the decrease in SSE while respecting the rule of hierarchy, i.e. a main effect is not compared to a model with an interaction term which includes that main effect. The sum of squares of A is for example the difference between the model with factor B and the model with factors A and B.
3. Type III calculates the decrease in SSE when the model with the factor is compared to the largest smaller model. The sum of square of factor A is for example the difference between the full factorial model and the model with factor B and the interaction between factors A and B.
4. Type IV is allegedly similar to type III, but with a correction for cases where some combination of factors is completely missing. The exact computation is unknown to me.

This piece mainly concerns the difference between type II and type III, as those are the most common choices. We develop the description of the two types using an example.

Numeric example

We create a pseudo-random dataset with two binary factors A and B. A third variable A_B is calculated as the indicator of $A = B$ and the response y is created so that there is an effect of factor B but not of factor A or the interaction.

```r
set.seed(8)
dat <- data.frame(A = sample(0:1, 100, T),
                  B = sample(0:1, 100, T))
dat$A_B <- as.numeric(dat$A == dat$B)
dat$y <- dat$B + rnorm(100)
for(i in 1:3) {dat[, i] <- factor(dat[, i])}
```

```r
knitr::kable(dat[1:10, ], caption =
   "Ten first observations in randomly generated dataset.")
```

Table 1: Ten first observations in randomly generated dataset.

A	B	A_B	y
0	1	0	1.2968513
0	0	1	-1.9005736
1	0	0	-1.6473656
1	0	0	-1.7784054
0	1	0	1.0349434
1	1	1	0.5054536
0	0	1	0.6075449
1	1	1	0.6356352
1	1	1	1.1679343
1	1	1	1.5666227

Next, some different possible models are estimated and the SSE calculated. The anova table with type II sums of squares is calculated using the `Anova` function in the car package.

```r
sum(residuals(lm(y ~ A, dat))^2)
```

```
## [1] 132.1466
```

```r
sum(residuals(lm(y ~ B, dat))^2)
```

```
## [1] 112.7595
```

```r
sum(residuals(lm(y ~ A + B, dat))^2)
```

```
## [1] 111.8651
```

```r
sum(residuals(lm(y ~ A + B + A_B, dat))^2)
```

```
## [1] 110.0759
```

```r
library(car)

Anova(lm(y ~ A * B, dat), type = 2)
```

```
## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value    Pr(>F)
## A          0.894  1  0.7800    0.3793
## B         20.282  1 17.6880 5.848e-05 ***
## A:B        1.789  1  1.5604    0.2146
## Residuals 110.076 96
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can now identify the calculation of the sums of squares: SS_A is the decrease in SSE when the model with factor A and factor B is compared to the model with only factor B; SS_B is similar; SS_{A_B} is the decrease in SSE when the full-factorial model is compared to the model with factor A and factor B.

We make a similar calculation demonstrating sums of squares of type III. Note that specific contrasts must be set in order to get correct results from the Anova function.

```
sum(residuals(lm(y ~ A + A_B, dat))^2)
```

```
## [1] 130.8381
```

```
sum(residuals(lm(y ~ B + A_B, dat))^2)
```

```
## [1] 111.0342
```

```
sum(residuals(lm(y ~ A + B, dat))^2)
```

```
## [1] 111.8651
```

```
sum(residuals(lm(y ~ A + B + A_B, dat))^2)
```

```
## [1] 110.0759
```

```
Anova(lm(y ~ A * B, dat,
         contrasts = list(A = contr.sum,
                          B = contr.sum)),
      type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##              Sum Sq Df F value    Pr(>F)
## (Intercept)  23.424  1 20.4287 1.766e-05 ***
## A             0.958  1  0.8358    0.3629
## B            20.762  1 18.1072 4.857e-05 ***
## A:B           1.789  1  1.5604    0.2146
## Residuals   110.076 96
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

So the SS type III is the decrease in SSE when the interaction model is compared to a model where the factor of interest has been dropped. For example, the sum of squares connected to the factor B is the difference between the SSE of the full factorial model (110.08) and the SSE of the model with factor A and the interaction A_B (130.84).

These examples generalize to models with more factors with more factor levels, but not completely without difficulties as the factors must be codified into numeric variables.

Choice of type

The Case for Type III

Sums of squares of type III is default in most commercial programs (SPSS, Minitab and SAS, the latter being the origin of the formulation) and more common in publications than type II. Type III also has the advantage that the baseline remains the same, regardless of whether one is doing inference on an interaction or on a main effect. This might simplify calculations (?) and interpretation.

A possible disadvantage would be that the smaller model does not always make intuitive sense - the model with an interaction such as A_B, but without the connected main effect, A or B, is seldom used in practice.

Case for Type II

The main advantage of SS type II must be the greater power for the main effects, in comparison to type III. A simple example in an unbalanced design is given below. The p-value of the F-test on the effect of factor B differs greatly between SS type II and SS type III. In the type III case, the main effect is underestimated because of confounding with the interaction effect.

```r
set.seed(6)
dat <- data.frame(A = c(rep(0, 5), rep(0, 5),
                        rep(1, 50), rep(1, 50)),
                  B = c(rep(0, 5), rep(1, 5),
                        rep(0, 50), rep(1, 50)))
dat$y <- dat$B + rnorm(dim(dat)[1], sd = 1)
dat$AB <- dat$A == dat$B + 0
with(dat, table(A,B))
```

```
##    B
## A    0  1
##   0  5  5
##   1 50 50
```

```r
dat$A <- factor(dat$A)
dat$B <- factor(dat$B)

mod <- lm(y ~ A * B, dat,
          contrasts = list(A = contr.sum, B = contr.sum))

Anova(mod, type = 2)
```

```
## Anova Table (Type II tests)
##
## Response: y
##              Sum Sq  Df F value    Pr(>F)
## A           0.221   1  0.2188    0.6409
## B          17.693   1 17.5511 5.804e-05 ***
## A:B         0.675   1  0.6691    0.4152
## Residuals 106.858 106
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
Anova(mod, type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##              Sum Sq  Df F value   Pr(>F)
## (Intercept)  10.117   1 10.0361 0.002006 **
## A             0.221   1  0.2188 0.640911
## B             3.050   1  3.0257 0.084856 .
## A:B           0.675   1  0.6691 0.415193
## Residuals   106.858 106
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Type III in R

So far, we have only briefly touched upon the practicalities of calculating the different types of square sums in R. It turns out that this is an area where it is quite easy to make mistakes. Note that for balanced data, all types of sums of squares should give the same result, and see the following example which calculates the anova table using the `anova` function for type I and the `Anova` function from the `car` package for types II and III.

```r
set.seed(6)
dat <- data.frame(A = rep(0:1, 20),
                  B = rep(0:1, each = 20))
dat$y <- dat$B + rnorm(dim(dat)[1], sd = 1)
dat$AB <- dat$A == dat$B + 0
with(dat, table(A,B))
```

```
##    B
## A    0  1
##   0 10 10
##   1 10 10
```

```r
dat$A <- factor(dat$A)
dat$B <- factor(dat$B)

mod <- lm(y ~ A * B, dat)

anova(mod)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value  Pr(>F)
## A          1  1.941  1.9412  1.5204 0.22555
## B          1  7.433  7.4331  5.8218 0.02105 *
## A:B        1  0.434  0.4342  0.3401 0.56341
## Residuals 36 45.964  1.2768
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova(mod, type = 2)
```

```
## Anova Table (Type II tests)
##
## Response: y
##           Sum Sq Df F value  Pr(>F)
## A          1.941  1  1.5204 0.22555
## B          7.433  1  5.8218 0.02105 *
## A:B        0.434  1  0.3401 0.56341
## Residuals 45.964 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova(mod, type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##             Sum Sq Df F value Pr(>F)
## (Intercept)  0.960  1  0.7517 0.3917
## A            0.270  1  0.2112 0.6486
## B            5.730  1  4.4880 0.0411 *
## A:B          0.434  1  0.3401 0.5634
## Residuals   45.964 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The last tables differs in the calculation of the square sums of the main effects. It also contains an additional line for the intercept. A look at the design matrix for the four possible combinations of factors gives a hint of what has happened.

```
model.matrix(mod)[c(1, 2, 21, 22),]
```

```
##    (Intercept) A1 B1 A1:B1
## 1            1  0  0     0
## 2            1  1  0     0
## 21           1  0  1     0
## 22           1  1  1     1
```

The variable for the interaction has been codified as the product of the main effects, i.e. `A1:B1` is 1 if both `A1` and `B1` are 1. In the full-factorial model, this does not make a difference for the estimated values or the residuals, but if one is estimating a model including the interaction term but excluding a main effect (which is the case when estimating sums of type III), the model with this *product interaction* differs from the model with the *equality interaction* (i.e. where the interaction column is 1 if `A1` equals `B1`).

One way to correct this is to set contrasts. The contrasts specify how the factor levels are codified into numeric variables, and hence impact the form of the interaction term. We continue the example from above.

```r
mod <- lm(y ~ A * B, dat,
          contrasts = list(A = contr.sum, B = contr.sum))

Anova(mod, type = 3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##             Sum Sq Df F value  Pr(>F)
## (Intercept) 15.614  1 12.2291 0.00127 **
## A            1.941  1  1.5204 0.22555
## B            7.433  1  5.8218 0.02105 *
## A:B          0.434  1  0.3401 0.56341
## Residuals   45.964 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova table now corresponds with those of sums of type I and II. In this case the contrasts are specified in the `lm` call, but it can alternatively by done on the data frame:

```r
contrasts(dat$A) <- contr.sum
contrasts(dat$B) <- contr.sum

#mod <- lm(y ~ A * B, dat)
#Anova(mod, type = 3)
```

We take a look at the model matrix to see how the specification has changed.

```r
model.matrix(mod)[c(1, 2, 21, 22),]
```

```
##    (Intercept) A1 B1 A1:B1
## 1            1  1  1     1
## 2            1 -1  1    -1
## 21           1  1 -1    -1
## 22           1 -1 -1     1
```

The interaction variable is the product of the two main effects, but follows the pattern of the *equality interaction*, i.e. it takes one value when the main effects are equal and a different value when the main effects differ.