Contrasting Contrasts

Livio Finos

Contents

Abstract I recall the importance of the use of zero-sum contrasts for categorical variables compared to the usual coding in dummy variables. The problem remains the same for quantitative variables. I tackle the problem with a synthetic dataset and a linear model with a factor (= categorical variable), a quantitative variable and their interaction.

1 The data + EDA

The Challenge:
Let's build a dataset where the effects are known, can you analyze it adequately?

The proposed model is a simple ANCOVA model: - normal response (lm) with errors with 0 mean and variance equal to 1 - linear model with predictors: * two groups (`A` and`B`), * a continuous variable e * their interaction - Effects: Intercept and group. The continuous variable has no relation to the answer for the group `A`, it has it instead in the group`B` (interaction).

These are the data created and their representation.

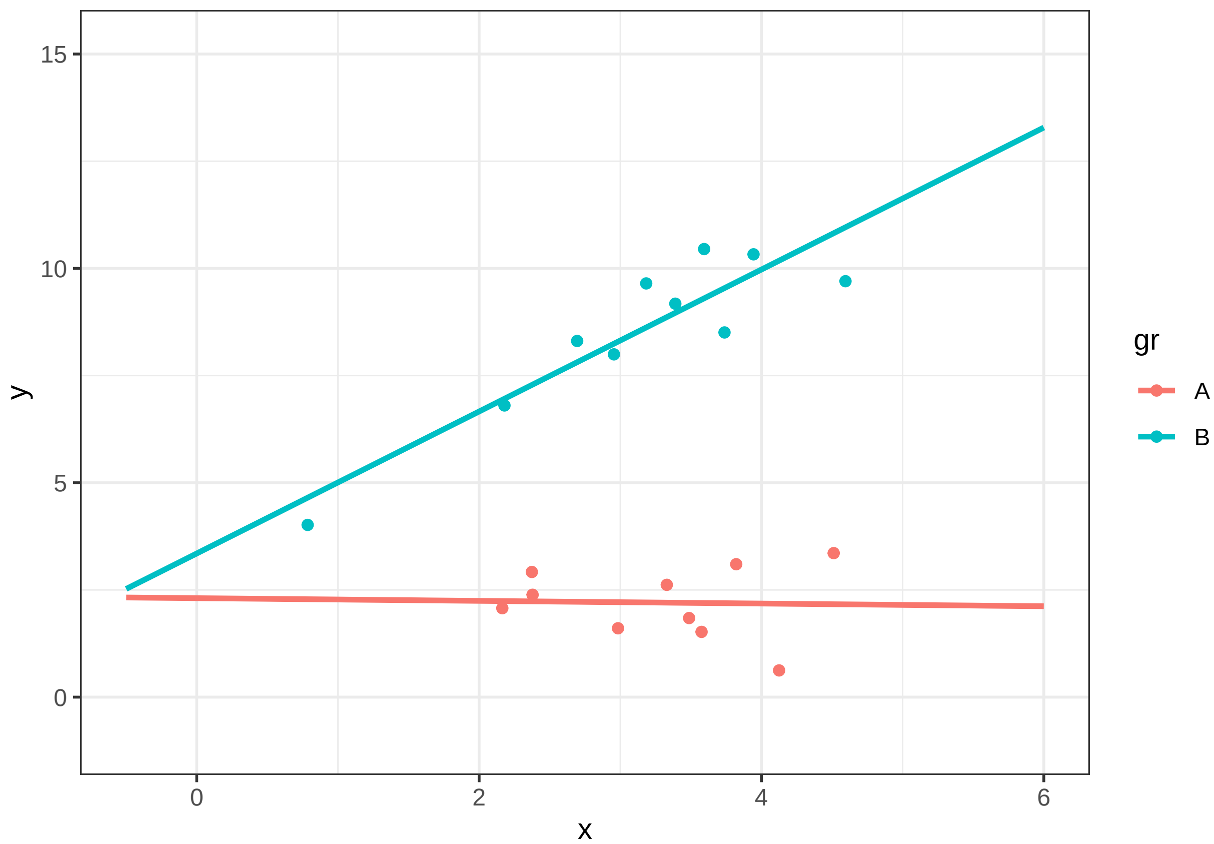
```
set.seed(1)
n0=10
D=data.frame(gr=as.factor(rep(LETTERS[1:2],n0)),
             x=rnorm(n0*2)+3)
mu=2+(D$gr=="B")*.5+D$x*(D$gr=="B")*2
D$y= mu+rnorm(n0*2)

D
```

```
##    gr         x          y
## 1   A 2.3735462  2.9189774
## 2   B 3.1836433  9.6494229
## 3   A 2.1643714  2.0745650
## 4   B 4.5952808  9.7012099
## 5   A 3.3295078  2.6198257
## 6   B 2.1795316  6.8029345
## 7   A 3.4874291  1.8442045
## 8   B 3.7383247  8.5058970
## 9   A 3.5757814  1.5218499
## 10  B 2.6946116  8.3071648
## 11  A 4.5117812  3.3586796
## 12  B 3.3898432  9.1768987
## 13  A 2.3787594  2.3876716
## 14  B 0.7853001  4.0167952
## 15  A 4.1249309  0.6229404
## 16  B 2.9550664  7.9951382
## 17  A 2.9838097  1.6057100
## 18  B 3.9438362 10.3283590
## 19  A 3.8212212  3.1000254
## 20  B 3.5939013 10.4509784
```

```r
library(ggplot2)
ggplot(D,aes(x=x,y=y,color=gr))+geom_point()+
  geom_smooth(method = "lm", fill = NA,fullrange = TRUE)+xlim(-.5, 6)+theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



2 Linear Models

2.1 A linear model?

```
modDU=lm(y~gr*x,data=D)
summary(modDU)
```

```
##
## Call:
## lm(formula = y ~ gr * x, data = D)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.55585 -0.61528 -0.00131  0.54234  1.19203
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.30818    1.24368   1.856  0.08198 .
## grB          1.04292    1.53659   0.679  0.50701
## x           -0.03137    0.37016  -0.085  0.93352
## grB:x        1.68703    0.46200   3.652  0.00215 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8778 on 16 degrees of freedom
## Multiple R-squared:  0.9481, Adjusted R-squared:  0.9384
## F-statistic: 97.49 on 3 and 16 DF,  p-value: 1.708e-10
```

matrix of predictors (i.e. independent variables)

```r
(mm <- model.matrix(~gr*x,data=D))
```

```
##    (Intercept) grB         x     grB:x
## 1            1   0 2.3735462 0.0000000
## 2            1   1 3.1836433 3.1836433
## 3            1   0 2.1643714 0.0000000
## 4            1   1 4.5952808 4.5952808
## 5            1   0 3.3295078 0.0000000
## 6            1   1 2.1795316 2.1795316
## 7            1   0 3.4874291 0.0000000
## 8            1   1 3.7383247 3.7383247
## 9            1   0 3.5757814 0.0000000
## 10           1   1 2.6946116 2.6946116
## 11           1   0 4.5117812 0.0000000
## 12           1   1 3.3898432 3.3898432
## 13           1   0 2.3787594 0.0000000
## 14           1   1 0.7853001 0.7853001
## 15           1   0 4.1249309 0.0000000
## 16           1   1 2.9550664 2.9550664
## 17           1   0 2.9838097 0.0000000
## 18           1   1 3.9438362 3.9438362
## 19           1   0 3.8212212 0.0000000
## 20           1   1 3.5939013 3.5939013
## attr(,"assign")
## [1] 0 1 2 3
## attr(,"contrasts")
## attr(,"contrasts")$gr
## [1] "contr.treatment"
```

Have a look to the mixed moments (i.e. covariance without centering around the mean) between predictors. mixed moments equal to 0 means orthogonal predictors:

```r
t(mm)%*%mm/nrow(mm)
```

```
##             (Intercept)      grB         x    grB:x
## (Intercept)    1.000000 0.500000  3.190524 1.552967
## grB            0.500000 0.500000  1.552967 1.552967
## x              3.190524 1.552967 10.971773 5.327434
## grB:x          1.552967 1.552967  5.327434 5.327434
```

and the Multiple R-squared of the first three columns to explain the interaction column:

```r
summary(lm(mm[,2]~mm[,-2]+0))
```

```
##
## Call:
## lm(formula = mm[, 2] ~ mm[, -2] + 0)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.243736 -0.060709  0.005904  0.071867  0.265952
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## mm[, -2](Intercept)  0.65509    0.11529   5.682 2.70e-05 ***
## mm[, -2]x           -0.19006    0.03590  -5.294 5.95e-05 ***
## mm[, -2]grB:x        0.29060    0.01871  15.528 1.79e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1385 on 17 degrees of freedom
## Multiple R-squared:  0.9674, Adjusted R-squared:  0.9616
## F-statistic:   168 on 3 and 17 DF,  p-value: 7.853e-13
```

2.2 Another linear model? (x 0-centered)

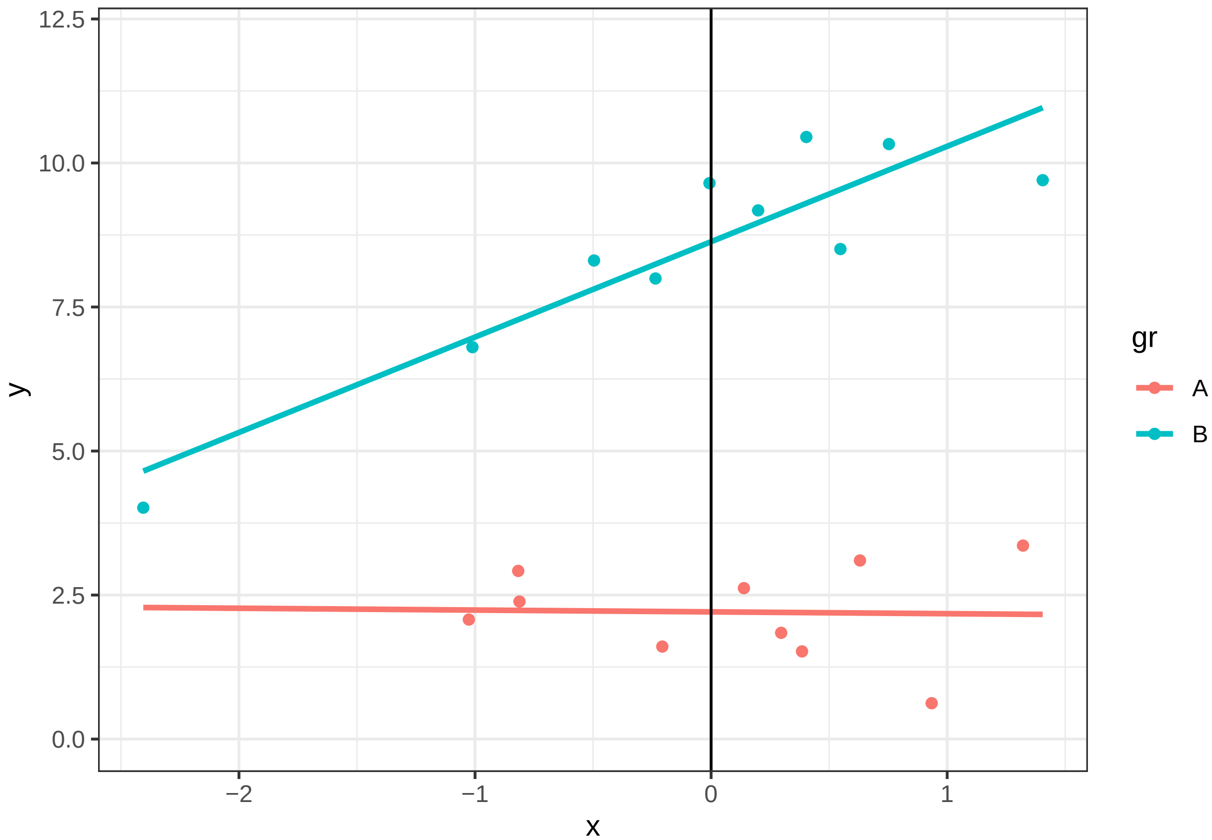
```r
D2=D
D2$x=D$x-mean(D$x)
modDUC=lm(y~gr*x,data=D2)
summary(modDUC)
```

```
##
## Call:
## lm(formula = y ~ gr * x, data = D2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.55585 -0.61528 -0.00131  0.54234  1.19203
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.20810    0.27933   7.905 6.47e-07 ***
## grB          6.42543    0.39449  16.288 2.21e-11 ***
## x           -0.03137    0.37016  -0.085  0.93352
## grB:x        1.68703    0.46200   3.652  0.00215 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8778 on 16 degrees of freedom
## Multiple R-squared:  0.9481, Adjusted R-squared:  0.9384
## F-statistic: 97.49 on 3 and 16 DF,  p-value: 1.708e-10
```

Observe the 0 on the abscise: there is a clear difference between groups A and B.

```r
ggplot(D2,aes(x=x,y=y,color=gr))+geom_point()+
  geom_smooth(method = "lm", fill = NA,fullrange = TRUE)+ geom_vline(xintercept = 0)+theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Correlations between predictors

It is also useful to evaluate the correlations between predictors.

```r
mm <- model.matrix(~gr*x,data=D2)
head(mm)
```

```
##   (Intercept) grB            x        grB:x
## 1           1   0 -0.816977687  0.000000000
## 2           1   1 -0.006880552 -0.006880552
## 3           1   0 -1.026152489  0.000000000
## 4           1   1  1.404756926  1.404756926
## 5           1   0  0.138983896  0.000000000
## 6           1   1 -1.010992260 -1.010992260
```

Have a look to the mixed moments (i.e. covariance without centering around the mean) between predictors. mixed moments equal to 0 means orthogonal predictors:

```r
t(mm)%*%mm/nrow(mm)
```

```
##               (Intercept)         grB             x       grB:x
## (Intercept)  1.000000e+00  0.50000000  1.332268e-16 -0.04229497
## grB          5.000000e-01  0.50000000 -4.229497e-02 -0.04229497
## x            1.332268e-16 -0.04229497  7.923307e-01  0.50759860
## grB:x       -4.229497e-02 -0.04229497  5.075986e-01  0.50759860
```

and the Multiple R-squared of the first three columns to explain the interaction column:

```r
summary(lm(mm[,2]~mm[,-2]+0))
```

```
##
## Call:
## lm(formula = mm[, 2] ~ mm[, -2] + 0)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.57782 -0.48222 -0.00215  0.50046  0.55697
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## mm[, -2](Intercept)  0.50139    0.12127   4.135 0.000693 ***
## mm[, -2]x           -0.07448    0.22686  -0.328 0.746694
## mm[, -2]grB:x        0.03293    0.28393   0.116 0.909022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5397 on 17 degrees of freedom
## Multiple R-squared:  0.5049, Adjusted R-squared:  0.4175
## F-statistic: 5.779 on 3 and 17 DF,  p-value: 0.006501
```

2.3 a third linear model?? (x 0-centered + gr 0-centered)

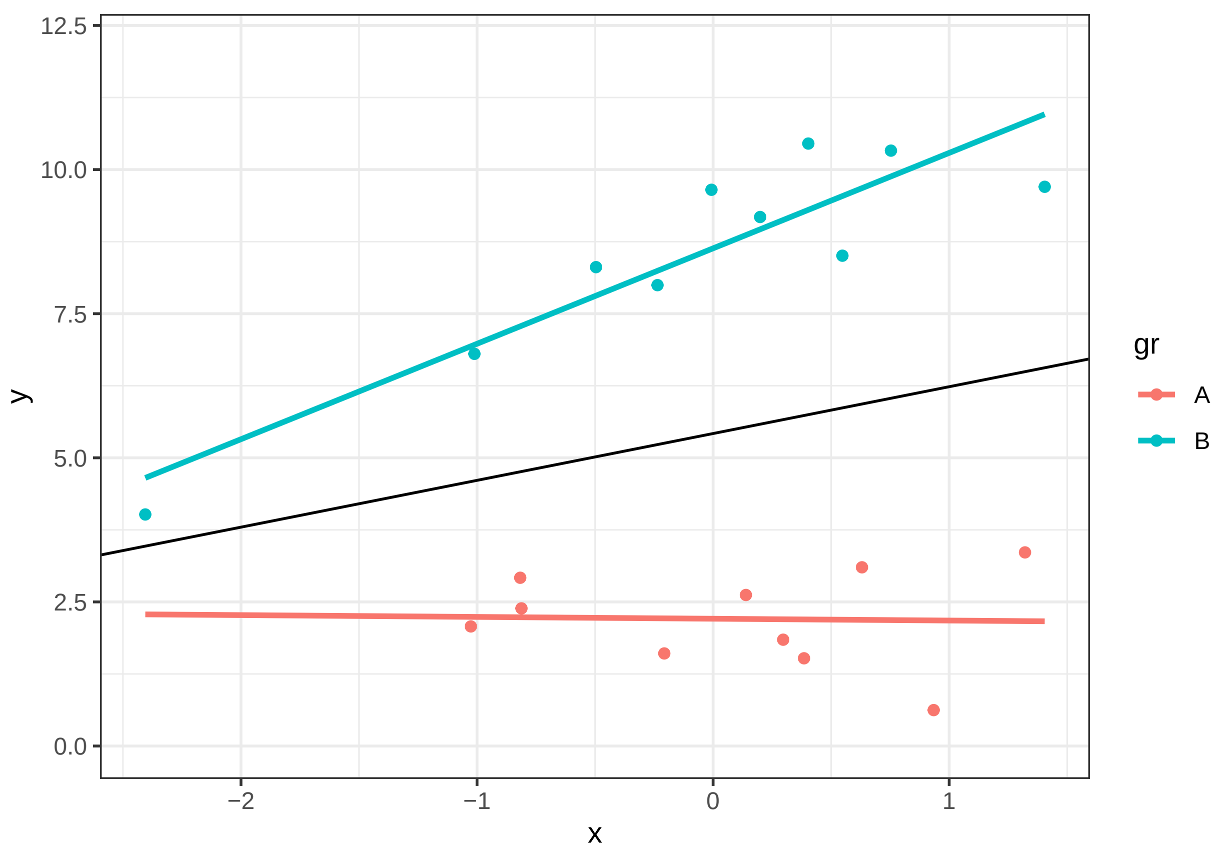
```r
D3=D2
contrasts(D3$gr)=contr.sum(2)
modS0=lm(y~gr*x,data=D3)
summary(modS0)
```

```
##
## Call:
## lm(formula = y ~ gr * x, data = D3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.55585 -0.61528 -0.00131  0.54234  1.19203
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.4208     0.1972  27.483 6.80e-15 ***
```

```
## gr1          -3.2127     0.1972 -16.288 2.21e-11 ***
## x             0.8121     0.2310   3.516  0.00287 **
## gr1:x        -0.8435     0.2310  -3.652  0.00215 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8778 on 16 degrees of freedom
## Multiple R-squared:  0.9481, Adjusted R-squared:  0.9384
## F-statistic: 97.49 on 3 and 16 DF,  p-value: 1.708e-10
```

```
ggplot(D3,aes(x=x,y=y,color=gr))+geom_point()+
  geom_smooth(method = "lm", fill = NA,fullrange = TRUE)+ geom_abline(intercept = coef(modS0)[1], slope
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



The black line in the graph represents the equation: `Y = ` 5.4208153 `+` 0.8121474 `X` as estimated by the model. Actually, this is the effect for subjects with `gr == 0` which do not exist in reality, but represent an intermediate (somewhat null) value; are therefore an estimate *net of* the effects of `gr`.

Let's now observe how the predictor matrix has changed (in particular the interaction):

```
mm <- model.matrix(~gr*x,data=D3)
head(mm)
```

```
##   (Intercept) gr1           x        gr1:x
```

```
## 1           1   1 -0.816977687 -0.816977687
## 2           1  -1 -0.006880552  0.006880552
## 3           1   1 -1.026152489 -1.026152489
## 4           1  -1  1.404756926 -1.404756926
## 5           1   1  0.138983896  0.138983896
## 6           1  -1 -1.010992260  1.010992260
```

And have a look to the mixed moments (i.e. covariance without centering around the mean) between predictors. mixed moments equal to 0 means orthogonal predictors:

```
t(mm)%*%mm/nrow(mm)
```

```
##              (Intercept)          gr1             x         gr1:x
## (Intercept) 1.000000e+00 0.000000e+00  1.332268e-16  8.458994e-02
## gr1         0.000000e+00 1.000000e+00  8.458994e-02  1.332268e-16
## x           1.332268e-16 8.458994e-02  7.923307e-01 -2.228665e-01
## gr1:x       8.458994e-02 1.332268e-16 -2.228665e-01  7.923307e-01
```

and:

```
summary(lm(mm[,2]~mm[,-2]+0))
```

```
##
## Call:
## lm(formula = mm[, 2] ~ mm[, -2] + 0)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.11394 -1.00093  0.00431  0.96444  1.15564
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## mm[, -2](Intercept) -0.002786   0.242535  -0.011    0.991
## mm[, -2]x            0.116024   0.282650   0.410    0.687
## mm[, -2]gr1:x        0.032933   0.283935   0.116    0.909
##
## Residual standard error: 1.079 on 17 degrees of freedom
## Multiple R-squared:  0.009814,   Adjusted R-squared:  -0.1649
## F-statistic: 0.05617 on 3 and 17 DF,  p-value: 0.9819
```

3 A simulation

What would the three models tell me if I could repeat the experiment many times?

```
res_sim=replicate(1000,
          {
            D$y <- D2$y <- D3$y <- mu+rnorm(n0*2)
            modDU=lm(y~gr*x,data=D)
            p_DU=summary(modDU)$coeff[,4]
            modDUC=lm(y~gr*x,data=D2)
```

```
            p_DUC=summary(modDUC)$coeff[,4]
            modS0=lm(y~gr*x,data=D3)
            p_S0=summary(modS0)$coeff[,4]
            c(DU=p_DU,DUC=p_DUC,S0=p_S0)
          })

res_sim=t(res_sim)
```

(Empirical) Power:

```
library(r41sqrt10)
```

```
##
## Attaching package: 'r41sqrt10'

## The following object is masked from 'package:base':
##
##     mode
```

```
## model Dummy
summaryResSim(res_sim[,1:4])
```

```
##                     <=0.01     <=0.05      <=0.1     <=0.5    <=0.75
## LowerLim       0.003707147 0.03621595 0.08102633 0.4683772 0.7226139
## UpperLim       0.016292853 0.06378405 0.11897367 0.5316228 0.7773861
## DU.(Intercept) 0.095000000 0.24400000 0.36400000 0.7730000 0.9040000
## DU.grB         0.012000000 0.04600000 0.09800000 0.5170000 0.7420000
## DU.x           0.006000000 0.04300000 0.10000000 0.4960000 0.7470000
## DU.grB:x       0.783000000 0.94000000 0.97400000 0.9990000 1.0000000
```

```
## model Dummy +  Centered X
summaryResSim(res_sim[,5:8])
```

```
##                      <=0.01     <=0.05      <=0.1     <=0.5    <=0.75
## LowerLim        0.003707147 0.03621595 0.08102633 0.4683772 0.7226139
## UpperLim        0.016292853 0.06378405 0.11897367 0.5316228 0.7773861
## DUC.(Intercept) 0.997000000 1.00000000 1.00000000 1.0000000 1.0000000
## DUC.grB         1.000000000 1.00000000 1.00000000 1.0000000 1.0000000
## DUC.x           0.006000000 0.04300000 0.10000000 0.4960000 0.7470000
## DUC.grB:x       0.783000000 0.94000000 0.97400000 0.9990000 1.0000000
```

```
## model S0 +  Centered X
summaryResSim(res_sim[,9:12])
```

```
##                     <=0.01     <=0.05      <=0.1     <=0.5    <=0.75
## LowerLim       0.003707147 0.03621595 0.08102633 0.4683772 0.7226139
## UpperLim       0.016292853 0.06378405 0.11897367 0.5316228 0.7773861
## S0.(Intercept) 1.000000000 1.00000000 1.00000000 1.0000000 1.0000000
## S0.gr1         1.000000000 1.00000000 1.00000000 1.0000000 1.0000000
## S0.x           0.781000000 0.93400000 0.97100000 0.9970000 1.0000000
## S0.gr1:x       0.783000000 0.94000000 0.97400000 0.9990000 1.0000000
```

```
# prova anche con
# D3=D2
# contrasts(D3$gr)=contr.sum(2)
# modS0=lm(y~gr*x,data=D3)
# X non centrata e contr.sum(2)
```

4 Conclusion

The definition of an effect depends crucially on the way we encode variables (whether they are factors or continuous). The interpretation changes depending on the encoding.

Furthermore, the power to test these effects dependes on the dependendence among predictors (i.e. cross product, mixed moments). This point becomes salient in interactions, where correlations are often high by nature (interactions are defined as a product of the columns of the experimental design).

When possible (and sensible), the recommendation is to center the contrasts around 0 (see using `contr.sum ()`).

4.1 PS: What if I use ANOVA (i.e. LRT) tests?

Even the results of the Anova test depend of the contrasts (and linear transformations of the original variables) used to fit the model. (In this case) one get exactly the same results:

```
car::Anova(modDU, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##              Sum Sq Df F value   Pr(>F)
## (Intercept)  2.6538  1  3.4445 0.081978 .
## gr           0.3549  1  0.4607 0.507012
## x            0.0055  1  0.0072 0.933518
## gr:x        10.2731  1 13.3338 0.002152 **
## Residuals   12.3273 16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
car::Anova(modDUC, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##              Sum Sq Df  F value    Pr(>F)
## (Intercept)  48.144  1  62.4878 6.474e-07 ***
## gr          204.405  1 265.3028 2.208e-11 ***
## x             0.006  1   0.0072  0.933518
## gr:x         10.273  1  13.3338  0.002152 **
## Residuals    12.327 16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
car::Anova(modS0, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##             Sum Sq Df F value    Pr(>F)
## (Intercept) 581.94  1 755.311 6.796e-15 ***
## gr          204.40  1 265.303 2.208e-11 ***
## x             9.52  1  12.361  0.002867 **
## gr:x         10.27  1  13.334  0.002152 **
## Residuals    12.33 16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```