

## Lab 8 - design matrix

This lab offers practice with design matrices and an introduction to summaries of the `lm()` function. Let's work with the spider data, described in 5.8

```
spider <- read.csv("P:/My Documents/BDA_Spring2018/spider_wolff_gorb_2013.csv")
```

**Q1.** Create the design matrix for modeling the effect of different leg sets on the observed friction coefficient. Provide only the R code for doing this (it's a long matrix), and write a few sentences to describe the components of the matrix (i.e. say what comprise the rows, columns, and values in each cell).

```
spider.dm <- model.matrix(~factor(spider$leg) -1) #as a boolean  
spider.dm[c(67:70,99:100,240:241),] #subset of the design matrix
```

```
##      factor(spider$leg)L1 factor(spider$leg)L2 factor(spider$leg)L3  
## 67          1          0          0  
## 68          1          0          0  
## 69          0          1          0  
## 70          0          1          0  
## 99          0          0          1  
## 100         0          0          1  
## 240         0          0          0  
## 241         0          0          0  
##      factor(spider$leg)L4  
## 67          0  
## 68          0  
## 69          0  
## 70          0  
## 99          0  
## 100         0  
## 240         1  
## 241         1
```

This design matrix acts as a boolean matrix setting the value in a cell to 1 when that individual (row)

If you are asking for the effect of different leg sets and whether the legs are pushing or pulling, then the design matrix becomes:

```
spider.full.dm <- model.matrix(friction ~ factor(leg) + factor(type), data=spider)  
spider.full.dm[c(1:2,67:70,80:85,99:100,240:241,nrow(spider)),] #subset of the design matrix
```

```
##      (Intercept) factor(leg)L2 factor(leg)L3 factor(leg)L4 factor(type)push  
## 1              1          0          0          0          0  
## 2              1          0          0          0          0  
## 67             1          0          0          0          1  
## 68             1          0          0          0          1  
## 69             1          1          0          0          0  
## 70             1          1          0          0          0  
## 80             1          1          0          0          0  
## 81             1          1          0          0          0  
## 82             1          1          0          0          0  
## 83             1          1          0          0          0  
## 84             1          1          0          0          1  
## 85             1          1          0          0          1
```

## 99	1	0	1	0	0
## 100	1	0	1	0	0
## 240	1	0	0	1	0
## 241	1	0	0	1	0
## 282	1	0	0	1	1

**Q2.** How many parameters in this design? Explain what they are and what information they provide, referencing the design matrix you produced in Q1. Then provide the parameter matrix.

There are four parameters in the leg design. The intercept,  $\beta_1$ , which is the mean of friction from L1 leg pair.  $\beta_2$  is the offset from the intercept to the mean of L2. legL3,  $\beta_3$ , is the effect size of L3 from L1 ( $\beta_1 + \beta_3 =$  mean friction of L3). The final parameter is  $\beta_4$  which is the effect size of L4 (the offset from L1 to L4).

$$\begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}$$

This full design matrix is not boolean (missing the “+0”). In this case the Intercept,  $\beta_0$ , is the mean friction of L1 when pulling. The last column,  $\beta_4$ , is the effect size of pushing so mean pushing friction of L1 is  $\beta_0 + \beta_4$ . The parameters  $\beta_j$  ( $j = 1-3$ ) are the offset of pulling friction for each leg (L2 to L4). Additionally adding  $\beta_4$  to any of these will give the effect size of pushing to each leg pair.

$$\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}$$

lm() handles all that linear algebra we did in lab last week and in class this week. It outputs a lot of information. Let’s take a look at a model for the effect of pushing vs. pulling (movement type) on the friction coefficient.

```
modelfit <- lm(friction~type, data=spider)
str(modelfit)

## List of 13
## $ coefficients : Named num [1:2] 1.211 -0.779
##   .. attr(*, "names")= chr [1:2] "(Intercept)" "typepush"
## $ residuals    : Named num [1:282] -0.311 -0.301 -0.351 -0.361 -0.411 ...
##   .. attr(*, "names")= chr [1:282] "1" "2" "3" "4" ...
## $ effects      : Named num [1:282] -13.799 -6.541 -0.319 -0.329 -0.379 ...
##   .. attr(*, "names")= chr [1:282] "(Intercept)" "typepush" "" "" ...
## $ rank         : int 2
## $ fitted.values: Named num [1:282] 1.21 1.21 1.21 1.21 1.21 ...
##   .. attr(*, "names")= chr [1:282] "1" "2" "3" "4" ...
## $ assign       : int [1:2] 0 1
## $ qr           :List of 5
##   ..$ qr      : num [1:282, 1:2] -16.7929 0.0595 0.0595 0.0595 0.0595 ...
##   .. .. attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:282] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:2] "(Intercept)" "typepush"
##   .. .. attr(*, "assign")= int [1:2] 0 1
##   .. .. attr(*, "contrasts")=List of 1
##   .. ..$ type: chr "contr.treatment"
##   ..$ qraux: num [1:2] 1.06 1.06
```

```
## ..$ pivot: int [1:2] 1 2
## ..$ tol : num 1e-07
## ..$ rank : int 2
## ..- attr(*, "class")= chr "qr"
## $ df.residual : int 280
## $ contrasts :List of 1
## ..$ type: chr "contr.treatment"
## $ xlevels :List of 1
## ..$ type: chr [1:2] "pull" "push"
## $ call : language lm(formula = friction ~ type, data = spider)
## $ terms :Classes 'terms', 'formula' language friction ~ type
## .. ..- attr(*, "variables")= language list(friction, type)
## .. ..- attr(*, "factors")= int [1:2, 1] 0 1
## .. .. ..- attr(*, "dimnames")=List of 2
## .. .. ..$ : chr [1:2] "friction" "type"
## .. .. ..$ : chr "type"
## .. ..- attr(*, "term.labels")= chr "type"
## .. ..- attr(*, "order")= int 1
## .. ..- attr(*, "intercept")= int 1
## .. ..- attr(*, "response")= int 1
## .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
## .. ..- attr(*, "predvars")= language list(friction, type)
## .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "factor"
## .. .. ..- attr(*, "names")= chr [1:2] "friction" "type"
## $ model :'data.frame': 282 obs. of 2 variables:
## ..$ friction: num [1:282] 0.9 0.91 0.86 0.85 0.8 0.87 0.92 0.83 0.88 0.87 ...
## ..$ type : Factor w/ 2 levels "pull","push": 1 1 1 1 1 1 1 1 1 1 ...
## ..- attr(*, "terms")=Classes 'terms', 'formula' language friction ~ type
## .. .. ..- attr(*, "variables")= language list(friction, type)
## .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
## .. .. .. ..- attr(*, "dimnames")=List of 2
## .. .. .. ..$ : chr [1:2] "friction" "type"
## .. .. .. ..$ : chr "type"
## .. .. ..- attr(*, "term.labels")= chr "type"
## .. .. ..- attr(*, "order")= int 1
## .. .. ..- attr(*, "intercept")= int 1
## .. .. ..- attr(*, "response")= int 1
## .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
## .. .. ..- attr(*, "predvars")= language list(friction, type)
## .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "factor"
## .. .. .. ..- attr(*, "names")= chr [1:2] "friction" "type"
## - attr(*, "class")= chr "lm"
```

Lots of results. The first few are the most commonly used. Coefficients are the  $\hat{\beta}$  of course. Residuals are the  $\epsilon$ . The data, of course, provide the X and Y. So why all that other stuff? Much of it is beyond the scope of this course; we'll just look at summaries using the `summary.lm()` function. Note: can also just use `summary()` and R will recognize that it's an `lm` object...

```
summary(modelfit)
```

```
##
## Call:
## lm(formula = friction ~ type, data = spider)
##
## Residuals:
```

```
##      Min      1Q  Median      3Q      Max
## -0.6212 -0.1290 -0.0222  0.1288  0.6288
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.21121    0.01946   62.23  <2e-16 ***
## typepush    -0.77901    0.02752  -28.30  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2311 on 280 degrees of freedom
## Multiple R-squared:  0.741, Adjusted R-squared:  0.7401
## F-statistic: 801 on 1 and 280 DF, p-value: < 2.2e-16
```

Nice and tidy. First, R reminds us which model we fit. Next it provides the 5-point summary for the distribution of the  $\epsilon_i$ . Next, it gives the  $\hat{\beta}_i$ , associated uncertainties, and the results for a t-test of the null hypothesis that the given coefficient value is equal to zero. It next provides the standard error of the distribution of  $\epsilon_i$  (and the denominator value so that a person could compute the standard deviation instead). Next is the  $R^2$  and adjusted  $R^2$ . We can peek at the summary.lm function to figure out how  $R^2$  and adjusted  $R^2$  are computed:

- 1)  $R^2 = \frac{MSS}{(MSS+RSS)}$
- 2)  $adj.R^2 = 1 - (1 - r.squared) * \frac{(n-df.int)}{rdf}$

MSS is the mean sum of squares. RSS is the residual sum of squares. df.int is 1 if an intercept term is estimated, 0 if not. rdf is the residual degrees of freedom (N-p).

I include this only because students always ask about the adjusted  $R^2$ . More generally,  $R^2$  is a slippery statistic, and it's often times much less useful than it's touted as being. We'll discuss the computations and interpretations in class. Some will benefit from this, others not so much. And that's OK.

Finally, the F statistic is the test statistic for all linear models. It is used to test the null hypothesis that the data were generated by a random process centered on a single value for the mean, and variance  $\sigma^2$ . We'll cover that in more detail later.

---

**Q3.** Given the output for the linear model presented above - for the effect of movement type (push v. pull) on the friction coefficient - what is the mean value of the friction coefficient for the spider legs that were pushing?

```
mean(spider$friction[spider$type=="push"])
```

```
## [1] 0.4321986
```

```
1.21121+(-0.77901)
```

```
## [1] 0.4322
```

The legs that were pushing have a mean value of 0.43. This is the intercept plus the effect size of the type push estimate.

**Q4.** Fit the linear model for the design you provided in Q1 and write down the value for the mean friction coefficient for each of the 4 leg sets.

```
lm(friction~leg,data=spider)
```

```
##
```

```
## Call:
```

```
## lm(formula = friction ~ leg, data = spider)
```

```
##
## Coefficients:
## (Intercept)      legL2      legL3      legL4
##      0.6644      0.1719      0.1605      0.2813

lm(friction~leg+0,data=spider) # My first design matrix

##
## Call:
## lm(formula = friction ~ leg + 0, data = spider)
##
## Coefficients:
##  legL1  legL2  legL3  legL4
## 0.6644 0.8363 0.8249 0.9457

spider.lm <- lm(friction~leg,data=spider)
spider.lm.forceintzero <- lm(friction~leg+0,data=spider)

coefs <- c("L1","L2","L3","L4")
coefs.full <- expand.grid(c("pull","push"),c("L1","L2","L3","L4"))

for(i in 1:4){
  print(paste("Mean friction coefficient for ", coefs[i], ": mean = ",
    round(mean(spider$friction[spider$leg==coefs[i]]),2), #L1
    ", beta_", i-1," = ",
    round(spider.lm.forceintzero$coefficients[i],2), sep=""))
  if(i>1){
    print(paste(" *Effect size for coefficient ", coefs[i], ", beta_",
      i-1, " = ",
      round(spider.lm$coefficients[i], 2), sep=""))
  }
}

## [1] "Mean friction coefficient for L1: mean = 0.66, beta_0 = 0.66"
## [1] "Mean friction coefficient for L2: mean = 0.84, beta_1 = 0.84"
## [1] " *Effect size for coefficient L2, beta_1 = 0.17"
## [1] "Mean friction coefficient for L3: mean = 0.82, beta_2 = 0.82"
## [1] " *Effect size for coefficient L3, beta_2 = 0.16"
## [1] "Mean friction coefficient for L4: mean = 0.95, beta_3 = 0.95"
## [1] " *Effect size for coefficient L4, beta_3 = 0.28"

#for(i in 1:5)
```

**Q5.** Given what you understand of the data, does the way we've been fitting the model seem to make intuitive sense, or should we be forcing the intercept to be zero? Explain.

Forcing the intercept to be zero makes the  $\hat{\beta}$  be the offset from 0, the mean instead of the effect size in comparison to the baseline value of each in comparison to the first parameter. When there is not a level that is the control, then forcing the intercept makes sense. However, in some cases we are interested in the effect size or difference from a baseline value for treatments compared to a control mean.

---

Finally, let's consider a multi-variate model (two-way ANOVA) for the additive effects of leg set and movement type on the friction coefficient. We'll fit the simple main effects model without and interaction.

```
twowayFit <- lm(friction ~ type + leg, data = spider)
summary(twowayFit)
```

```
##
## Call:
## lm(formula = friction ~ type + leg, data = spider)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46392 -0.13441 -0.00525  0.10547  0.69509
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.05392    0.02816  37.426 < 2e-16 ***
## typepush    -0.77901    0.02482 -31.380 < 2e-16 ***
## legL2        0.17192    0.04569   3.763 0.000205 ***
## legL3        0.16049    0.03251   4.937 1.37e-06 ***
## legL4        0.28134    0.03438   8.183 1.01e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2084 on 277 degrees of freedom
## Multiple R-squared:  0.7916, Adjusted R-squared:  0.7886
## F-statistic: 263 on 4 and 277 DF, p-value: < 2.2e-16
```

---

```
##      spider$legL1 spider$legL2 spider$legL3 spider$legL4 spider$typepush
## 233              0              0              0              1              0
## 234              0              0              0              1              0
## 235              0              0              0              1              0
## 236              0              0              0              1              0
## 237              0              0              0              1              0
## 238              0              0              0              1              0
## 239              0              0              0              1              0
## 240              0              0              0              1              0
## 241              0              0              0              1              0
## 242              0              0              0              1              0
## 243              0              0              0              1              1
## 244              0              0              0              1              1
## 245              0              0              0              1              1
## 246              0              0              0              1              1
## 247              0              0              0              1              1
## 248              0              0              0              1              1
## 249              0              0              0              1              1
## 250              0              0              0              1              1
## 251              0              0              0              1              1
## 252              0              0              0              1              1
## 253              0              0              0              1              1
## 254              0              0              0              1              1
## 255              0              0              0              1              1
## 256              0              0              0              1              1
## 257              0              0              0              1              1
## 258              0              0              0              1              1
## 259              0              0              0              1              1
## 260              0              0              0              1              1
```

## 261	0	0	0	1	1
## 262	0	0	0	1	1
## 263	0	0	0	1	1
## 264	0	0	0	1	1
## 265	0	0	0	1	1
## 266	0	0	0	1	1
## 267	0	0	0	1	1
## 268	0	0	0	1	1
## 269	0	0	0	1	1
## 270	0	0	0	1	1
## 271	0	0	0	1	1
## 272	0	0	0	1	1
## 273	0	0	0	1	1
## 274	0	0	0	1	1
## 275	0	0	0	1	1
## 276	0	0	0	1	1
## 277	0	0	0	1	1
## 278	0	0	0	1	1
## 279	0	0	0	1	1
## 280	0	0	0	1	1
## 281	0	0	0	1	1
## 282	0	0	0	1	1

**Q6.** Write the sample space for this experimental design. Hint: this will be the set of all possible “states” for leg set and movement type that can be observed in the experiment. It may help to look at the actual design matrix you developed in Q1.

- In question one you asked for the states for leg set, not for movement type. I will go back and add the full set of leg sets and movement type in question 1.

The sample space of both leg set and movement type contains the following 8 rows:

```
expand.grid(c("pull","push"),c("L1","L2","L3","L4"))
```

```
##   Var1 Var2
## 1 pull  L1
## 2 push  L1
## 3 pull  L2
## 4 push  L2
## 5 pull  L3
## 6 push  L3
## 7 pull  L4
## 8 push  L4
```

**Q7.** Use that sample space to interpret each of the  $\hat{\beta}_i$ . That is, write down the experimental condition for the intercept, for typepush, legL2, etc.

```
leg<-factor(c(rep("L1",2),rep("L2",2),rep("L3",2),rep("L4",2)))
type<-factor(rep(c("pull","push"),4))

model.matrix(~ type + leg -1)
```

```
##   typepull typepush legL2 legL3 legL4
## 1      1      0      0      0      0
## 2      0      1      0      0      0
## 3      1      0      1      0      0
## 4      0      1      1      0      0
## 5      1      0      0      1      0
```

```
## 6      0      1      0      1      0
## 7      1      0      0      0      1
## 8      0      1      0      0      1
## attr("assign")
## [1] 1 1 2 2 2
## attr("contrasts")
## attr("contrasts")$type
## [1] "contr.treatment"
##
## attr("contrasts")$leg
## [1] "contr.treatment"
```

The experimental condition is:

- 1 - L1 pull is  $\beta_1$
- 2 - L1 push is  $\beta_2$
- 3 - L2 pull is  $\beta_1 + \beta_3$
- 4 - L2 push is  $\beta_2 + \beta_3$
- 5 - L3 pull is  $\beta_1 + \beta_4$
- 6 - L3 push is  $\beta_2 + \beta_4$
- 7 - L4 pull is  $\beta_1 + \beta_5$
- 8 - L4 push is  $\beta_2 + \beta_5$