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(spider$friction~factor(spider$leg)+0) #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)
```

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 this design. The intercept, β_0, which is the mean of friction from L1 leg pair. β_1 is the offset from the intercept to the mean of L2. legL3, β_2, is the effect size of L3 from L1 ($\beta_0 + \beta_2 =$ mean friction of L3). The final parameter is β_3 which is the effect size of L4 (the offset from L1 to L4).

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

---

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 ϵ. 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 ϵ_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 ϵ_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 inlcude this only because students always ask about the adjusted R^2. More generally, R^2 is a slipperly statistic, and it's often times much less useful that 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 σ^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 has 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
```

```
spider.lm <- lm(friction~leg,data=spider)
print("Mean friction coefficient for L1:")
```

```
## [1] "Mean friction coefficient for L1:"
```

```
 mean(spider$friction[spider$leg=="L1"]) #L1
```

```
## [1] 0.6644118
```

```
print("Mean friction coefficient for L2:")
```

```
## [1] "Mean friction coefficient for L2:"
```

```
 mean(spider$friction[spider$leg=="L2"]) #L2
```

```
## [1] 0.8363333
```

```
 spider.lm$coefficients[1]+spider.lm$coefficients[2]
```

```
## (Intercept)
##   0.8363333
```

```
print("Mean friction coefficient for L3:")
```

```
## [1] "Mean friction coefficient for L3:"
```

```
 mean(spider$friction[spider$leg=="L3"]) #L3
```

```
## [1] 0.8249038
```

```
 spider.lm$coefficients[1]+spider.lm$coefficients[3]
```

```
## (Intercept)
##   0.8249038
```

```r
print("Mean friction coefficient for L4:")
```

```
## [1] "Mean friction coefficient for L4:"
```

```r
 mean(spider$friction[spider$leg=="L4"]) #L4
```

```
## [1] 0.94575
```

```r
 spider.lm$coefficients[1]+spider.lm$coefficients[4]
```

```
## (Intercept)
##     0.94575
```

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.
If forcing the intercept to be zero can give the mean instead of the effect size of each in comparison to the first parameter, 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.

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.

```r
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
```

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.
The sample space contains the following 8 rows:

```r
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
```

Pull:L1, Pull:L2,

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.