Exercise 1: Number of parameters in a GLM

Suppose that you have three possible predictors to use in constructing a GLM to predict a response $Y$: $X_1$, a 4-level categorical predictor; $X_2$, a 3-level categorical predictor; and $X_3$, a continuous predictor.

1. How many total parameters does a model with purely additive effects of $X_1$, $X_2$, and $X_3$ have?

   **Answer:** 8: the intercept is one, each $k$-level categorical predictor contributes $k - 1$ new parameters, each continuous predictor contributes 1 parameter, and there is one residual noise parameter ($1 + 3 + 2 + 1 + 1$).

2. How many total parameters does a model with an interaction between $X_1$ and $X_2$, plus an additive effect of $X_3$, have?

   **Answer:** 14: now the interaction of $X_1$ and $X_2$ behaves like a single 12 (= $4 \times 3$)-level parameter so we have $1 + 11 + 1 + 1$.

3. How many total parameters does a model with an interaction between $X_2$ and $X_3$, plus an additive effect of $X_1$, have?

   **Answer:** 10: 1 for the intercept, 3 for $X_1$, 2 for $X_2$, one $X_3$ slope for each level of $X_2$, and one residual noise parameter ($1 + 3 + 2 + 3 + 1$).

Exercise 2: Linear regression: a practical example

The `elp` dataset contains naming-time and lexical-decision time data by college-age native speakers for 2197 English words from a dataset collected by Balota and Spieler (1998), along with a number of properties of each word. (This dataset is a slightly cleaned-up version of the `english` dataset provided by the `languageR` package; Baayen, 2008.) Use linear regression to assess the relationship between reaction time and NEIGHBORHOOD DENSITY (defined as the number of words of English differing from the target word by only a single-letter edit). Is higher neighborhood density associated with faster or slower reaction times? Introduce
written word (log-)frequency as a control variable. Does the direction of the neighborhood-density effect change? Is it a reliable effect (that is, what is its level of statistical significance)? Finally, is there an interaction between neighborhood density and word frequency in their effects on reaction time?

Carry out this analysis for both word-naming and lexical-decision recognition times. In both cases, write a careful interpretation of your findings, describing not only what you found but what it might imply regarding how word recognition works. Construct any visualizations you may need to help illustrate what's going on in the data. If you find any qualitative differences in the way that the two predictors (and their interaction, if any) affect reaction times, describe them carefully, and speculate why these differences might exist.

**Answer:** For both naming and lexical-decision data, higher neighborhood density is strongly associated with *faster* reaction time:

```r
> elp <- read.csv("~/ling/prob_lx_book/data/english_lexicon_project/elp.csv")
> summary(naming.m1 <- lm(exp(RTnaming) ~ Ncount,elp))
```

### Call:
```
lm(formula = exp(RTnaming) ~ Ncount, data = elp)
```

### Residuals:
```
    Min     1Q  Median     3Q    Max
```

### Coefficients:
```
                       Estimate  Std. Error   t value Pr(>|t|)
(Intercept)          480.00683    0.68539    700.34   <2e-16 ***
Ncount              -1.60141     0.08668    -18.48   <2e-16 ***
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Residual standard error: 19.83 on 2195 degrees of freedom
Multiple R-squared: 0.1346, Adjusted R-squared: 0.1342
F-statistic: 341.3 on 1 and 2195 DF,  p-value: < 2.2e-16

```r
> summary(lexdec.m1 <- lm(exp(RTlexdec) ~ Ncount,elp))
```

### Call:
```
lm(formula = exp(RTlexdec) ~ Ncount, data = elp)
```

### Residuals:
```
    Min     1Q  Median     3Q    Max
-139.51  -52.05  -12.71  42.86  343.22
```

### Coefficients:

2
However, when written frequency is added in as a control, the picture changes. For naming, neighborhood density remains a very clearly significant predictor of reaction time, with higher neighborhood density leading to faster naming times.

```r
> summary(naming.m.additive <- lm(exp(RTnaming) ~ Ncount + WrittenFrequencyLog2, elp))
```

Call:
`lm(formula = exp(RTnaming) ~ Ncount + WrittenFrequencyLog2, data = elp)`

Residuals:
```
       Min     1Q   Median     3Q    Max
-54.99  -13.11   -1.25   11.66   86.81
```

Coefficients:

|                     | Estimate | Std. Error | t value | Pr(>|t|) |
|---------------------|----------|------------|---------|----------|
| (Intercept)         | 494.14733| 1.24427    | 397.1   | <2e-16 ***|
| Ncount              | -1.48325 | 0.08382    | -17.7   | <2e-16 ***|
| WrittenFrequencyLog2| -2.05376 | 0.15328    | -13.4   | <2e-16 ***|

Residual standard error: 19.07 on 2194 degrees of freedom
Multiple R-squared: 0.2, Adjusted R-squared: 0.1993
F-statistic: 274.3 on 2 and 2194 DF, p-value: < 2.2e-16

For lexical decision, on the other hand, the effect of neighborhood density almost entirely disappears:

```r
> summary(lexdec.m.additive <- lm(exp(RTlexdec) ~ Ncount + WrittenFrequencyLog2, elp))
```

Call:
`lm(formula = exp(RTlexdec) ~ Ncount + WrittenFrequencyLog2, data = elp)`

Residual standard error: 19.07 on 2194 degrees of freedom
Multiple R-squared: 0.2, Adjusted R-squared: 0.1993
F-statistic: 274.3 on 2 and 2194 DF, p-value: < 2.2e-16
Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-212.768</td>
<td>-36.580</td>
<td>-5.132</td>
<td>32.099</td>
<td>284.216</td>
</tr>
</tbody>
</table>

Coefficients:

|                     | Estimate | Std. Error | t value | Pr(>|t|) |
|---------------------|----------|------------|---------|---------|
| (Intercept)         | 751.5315 | 3.4937     | 215.109 | <2e-16  *** |
| Ncount              | -0.4412  | 0.2354     | -1.875  | 0.061   . |
| WrittenFrequencyLog2| -16.4464 | 0.4304     | -38.213 | <2e-16  *** |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 53.56 on 2194 degrees of freedom
Multiple R-squared: 0.4053, Adjusted R-squared: 0.4048
F-statistic: 747.7 on 2 and 2194 DF, p-value: < 2.2e-16

(Note that frequency remains a highly significant predictor in all cases, with faster reaction times for more frequent words.)

Including interactions between neighborhood density and frequency adds nuance to the picture. We begin by centering both predictors so that the main effects remain reasonably interpretable despite the presence of interactions:

```r
> elp$cWrittenFrequencyLog2 <- scale(elp$WrittenFrequencyLog2, scale=F)
> elp$cNcount <- scale(elp$Ncount, scale=F)
```

Using the centered predictors, we see that for naming, higher neighborhood density and frequency both speed reaction time, and that there is also a small interaction:

```r
> summary(naming.m.interactive <- +   lm(exp(RTnaming) ~ cNcount * cWrittenFrequencyLog2, elp))
```

Call:

```
lm(formula = exp(RTnaming) ~ cNcount * cWrittenFrequencyLog2, elp)
```

Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-54.70</td>
<td>-13.09</td>
<td>-1.15</td>
<td>11.73</td>
<td>84.70</td>
</tr>
</tbody>
</table>

Coefficients:

|                     | Estimate | Std. Error | t value | Pr(>|t|) |
|---------------------|----------|------------|---------|---------|
| (Intercept)         | 469.85671 | 0.40744     | 1153.194 | <2e-16  *** |
| cNcount             | -1.49926  | 0.08354     | -17.946  | <2e-16  *** |
| cWrittenFrequencyLog2| -2.09424 | 0.15290     | -13.697  | <2e-16  *** |
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 18.99 on 2193 degrees of freedom
Multiple R-squared: 0.2072, Adjusted R-squared: 0.2061
F-statistic: 191 on 3 and 2193 DF, p-value: < 2.2e-16

> plot.ndensity.effect <- function(dat,m,ylim=c(-5,1),...) {
+   offset.down <- ylim[1] ## this is for putting the histogram we'll plot at the bottom
+   d <- density(dat$WrittenFrequencyLog2, from=min(dat$WrittenFrequencyLog2),
+                 to=max(dat$WrittenFrequencyLog2), n=1000)
+   x <- d$x
+   y <- coef(m)[2] + coef(m)[4]*x
+   plot(d$x,y,type="l",ylim=ylim,
+         xlab="Word frequency (log 2)",ylab="Neighborhood density effect size",...)
+   ## Overlay on top of this a density plot illustrating
+   ## distribution of word frequencies
+   lines(d$x,4*d$y+offset.down) ##
+   rug(dat$WrittenFrequencyLog2)
+   mu.x <- mean(dat$WrittenFrequencyLog2)
+   d1 <- density(dat$WrittenFrequencyLog2, from=mu.x, to=mu.x, n=1)
+   text(d1$x,4*d1$y+offset.down,expression(mu[Freq]), pos=3)
+   points(d1$x,4*d1$y+offset.down,pch=19,col="red")
+   ## get standard error of ncount effect
+   f <- function(freq) {
+     return(sqrt(vcov(m)[2,2] + 2*freq*vcov(m)[2,4] + freq^2*vcov(m)[4,4]))
+   }
+   cisize <- sapply(x,f) * qt(0.975,m$df.resid)
+   lines(x,y+cisize,lty=2)
+   lines(x,y-cisize,lty=2)
+ }

> naming.m.interactive.base <-
+ lm(exp(RTnaming) ~ Ncount * WrittenFrequencyLog2, elp)

> plot.ndensity.effect(elp,naming.m.interactive.base)
The curve at the bottom of the graph is a kernel density estimate of word frequency in this dataset, with the red dot indicating the “average” word frequency. Although the right edge of this graph shows that for the most frequent words in the dataset the neighborhood density effect seems to disappear, there are extremely few words with such high frequency (and inferences about the size of the neighborhood density effect are correspondingly weak).

For lexical decision, the picture is rather different. The “average”-frequency word has no neighborhood density effect:

```r
> summary(lexdec.m.interactive <-
+     lm(exp(RTlexdec) ~ cNcount * cWrittenFrequencyLog2, e1p))
Call:
lm(formula = exp(RTlexdec) ~ cNcount * cWrittenFrequencyLog2,
    data = e1p)
Residuals:
    Min     1Q Median     3Q    Max
```

6
Coefficients:

|                   | Estimate | Std. Error | t value | Pr(>|t|) |
|-------------------|----------|------------|---------|----------|
| (Intercept)       | 629.1796 | 1.1449     | 549.569 | <2e-16   *** |
| cNcount           | -0.4824  | 0.2347     | -2.055  | 0.04     *  |
| cWrittenFrequencyLog2 | -16.5504 | 0.4296     | -38.522 | <2e-16   *** |
| cNcount:cWrittenFrequencyLog2 | 0.3540  | 0.0871     | 38.522  | <2e-16   *** |

---

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 53.37 on 2193 degrees of freedom
Multiple R-squared: 0.4098, Adjusted R-squared: 0.409
F-statistic: 507.5 on 3 and 2193 DF, p-value: < 2.2e-16

> lexdec.m.interactive.base <-
+  lm(exp(RTlexdec ~ Ncount * WrittenFrequencyLog2, elp)
> plot.ndensity.effect(elp, lexdec.m.interactive.base, ylim=c(-5,4))
but there is a much larger interaction of word frequency and neighborhood density! When this interaction is taken into account (see figure), there is a very clear effect of neighborhood density for lexical decision, on low-frequency words, but for high frequency words there even seems to be a slight reverse effect of neighborhood density, such that high-frequency words with more neighbors are slower to be decided upon (though the error bars on this graph indicate that the effect in this region of the graph is far from clear).

Although both lexical decision and naming showed significant interactions between neighborhood density and frequency, examining the details of the frequency effect revealed that there is an important difference between how these predictors affect reaction time in the two different tasks. I speculate on a possible reason for this result as follows: in naming, insofar as words have regular pronunciations (i.e., their orthography-phonology mapping is transparent), neighborhood density can only help: one can start planning on how to pronounce the word without recognizing its exact identity, and thus start to name the word more quickly. In lexical decision, on the other hand, perhaps there are two routes by which that one can solve the task: either recognize the exact word being presented, or decide that the string is “wordlike” enough without necessarily being 100% certain of what word it is. A
high neighborhood density should slow the first route, but speed the second route; and since we know that higher-frequency words are faster overall to recognize, the first route should dominate more the higher-frequency the word. This would lead to the type of interaction that we see in the data.

References
