Data Jujutsu III – Papers from UofC*

Stefano Allesina & Graham Smith *University of Chicago*

Description of the data

I have collected information on the 13,140 papers published between 2011 and 2020 by researchers with a UofC affiliation in the field of biology—including all papers in multidisciplinary journals (as such, some papers from other fields are included). We are going to explore this data set to highlight the variety of research programs and publications produced by our faculty. We will use these data to test the hypothesis that `open access` publishing leads to more citations than paywalled publishing.

Recap of linear regression in R

To be able to complete the challenge, you need to know how `R` approaches linear regressions. Take $y = \{y_1, y_2, \ldots, y_n\}$ to be the variable that we want to model (containing numeric values), and suppose that we want to model y as a function of a numeric covariate $x = \{x_1, x_2, \ldots, x_n\}$. The simplest model we can choose is a linear regression:

$$y_i = \alpha + \beta x_i + \epsilon_i$$

where α is an intercept parameter, β is a slope parameter, and ϵ_i is the error we make when we predict y_i given x_i, α and β. Importantly, when we are writing this model we typically assume that ϵ_i is sampled independently from a Normal distribution with mean 0 and variance σ^2. If this is the case, we can explicitly solve for the "best" values of the parameters α, β and σ (i.e., compute their maximum likelihood estimates).

In `R`, which was born for statistics, coding the linear regression is very easy:

```
y <- rnorm(100, mean = 1, sd = 0.5) # generate random data
x <- y + rnorm(100, mean = 0, sd = 0.1)
# fit the model
my_model <- lm(y ~ x) # this encodes the model above
```

To determine the quality of fit, run

```
summary(my_model)
```

```
##
## Call:
## lm(formula = y ~ x)
##
```

*This document is included as part of the Advanced Computing packet for the U Chicago BSD qBio6 boot camp 2020. **Current version**: August 12, 2020; **Corresponding author**: sallesina@uchicago.edu.

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.229791 -0.073213  0.002596  0.066801  0.216159
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.02446    0.02053   1.191    0.236
## x            0.98086    0.01874  52.336   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09639 on 98 degrees of freedom
## Multiple R-squared:  0.9655, Adjusted R-squared:  0.9651
## F-statistic:  2739 on 1 and 98 DF,  p-value: < 2.2e-16
```

In particular, the `Adjusted R-squared`, ranging between 0 and 1, is a good indication (for the simple case above, it corresponds to the proportion of variance explained by the model).

When covariates are not numeric values, but rather categorical variables (e.g., strings, factors), then the model takes form:

$$y_i = \alpha + \beta_1 z_{i1} + \beta_2 z_{i2} + \ldots + \beta_k z_{ik} + + \epsilon_i$$

where z_{ik} is a variable taking value 1 when the observation i belongs to category k and 0 otherwise. As such, in the model, a certain "slope" β_k is turned on when the variable belongs to category k, and switched off otherwise. In `R`, the syntax is the same.

You can include interaction terms in the model. For example, suppose that you have two numerical covariates, v and w. Then if you write:

```
lm(y ~ v + w)
```

You are fitting a model having form:

$$y_i = \alpha + \beta_1 v_i + \beta_2 w_i + \epsilon_i$$

If you write:

```
lm(y ~ v*w)
```

you are fitting a more complicated model:

$$y_i = \alpha + \beta_1 v_i + \beta_2 w_i + \beta_3 v_i w_i + \epsilon_i$$

Finally, if you write:

```
lm(y ~ v:w)
```

You only retain the interaction:

$$y_i = \alpha + \beta_3 v_i w_i + \epsilon_i$$

Similarly, if you are dealing with categories, writing $v : w$ in the code amounts to constructing a new category, with one value for each combination of v and w.

More advanced: when you have categorical values, one of them serves as a baseline value (technically, the coefficient gets absorbed into the slope). You can set manually the baseline by calling the function `relevel`, specifying a new reference category (`ref`). For example:

```
data("warpbreaks")
# contrast
warpbreaks$tension <- relevel(warpbreaks$tension, ref = "L")
summary(lm(breaks ~ wool + tension, data = warpbreaks))
# with
warpbreaks$tension <- relevel(warpbreaks$tension, ref = "M")
summary(lm(breaks ~ wool + tension, data = warpbreaks))
```

Note that the models have identical fit, but that the values corresponding to the tension categories ("L" for low, "M" for medium and "H" for high) have changed.

The challenge

1. Read the data The data are stored in the file `All_UofC_Bio_2011-20.csv`. Read the file, and rename the columns for easier typing:

```
au = Authors
au_ids = ‘Author(s) ID‘
year = Year
journal = ‘Source title‘
cits = ‘Cited by‘
article = ‘Document Type‘
oa = ‘Access Type‘
```

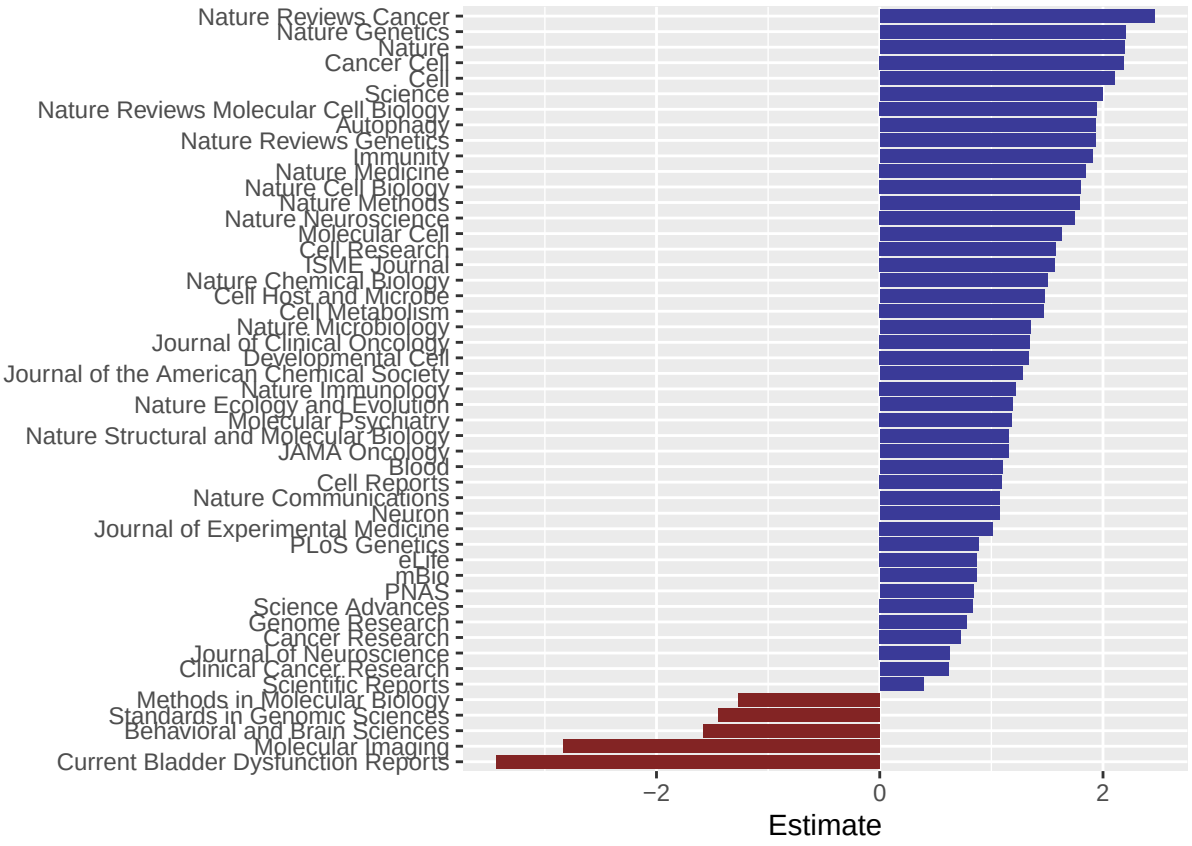
2. Distribution of citations by year Several studies have shown that the number of citations per paper in a given year is approximately lognormal (with better fit for older years).

Remove the papers with 0 citations, and plot the histogram for the log number of citations faceting by year. You should see that older years yield (approximately) a normal distribution. With this transformation at hand, you can attempt modeling the citations as a function of other bibliometric variables.

To start off, fit a linear model `log(cits) ~ I(2010 - year)`. Specifying `I(2010 - year)` builds a new variable containing the year -2010, so that a paper from 2011 would take value 1, from 2012 would take 2, etc. Because we expect the number of citations to grow with the age of the paper, the coefficient associated with this covariate measure the growth in number of citations for each year (to be specific, $e^{\beta} - 1$ is the proportion of increase for papers that are 1 year old).

3. Multi-authored papers The number of authors per paper varies dramatically. Count the number of authors per paper, and show that including a covariate specifying whether the paper has more than 12 authors improves the fit (store this variable in the column `multi`, and run `lm(log(cits) ~ year:multi)`). Similarly, including whether the article is a research article or a review improves the fit (column `article`).

4. Top journals Of course, research papers published in high-visibility journals (such as Nature and Science) tend to receive many more citations. We can model each journal separately as a categorical variable, and look at the distribution of effect sizes on citations: fit the model `log(cits) ~ I(2010 - year) + multi + article + journal`, and plot the effect sizes from most positive to most negative. Include only the journals with a strongly significant effect (e.g., a p-value $< 10^{-6}$ to avoid problems with multiple hypothesis testing), and draw a barplot such as the one below (obtained setting the baseline journal as "PLoS ONE" using `relevel`):



4. Effects of open access Some of the journals have an "open access option", which typically costs top dollars. Will this give your article more visibility (and therefore citations)? To test this effect, find all the journals that have published both open access and paywalled articles in the same year, and contrast the citation counts between the two subsets. To do so, extract all the papers for the journal/year combinations where you have paywalled and open access papers. Test the effect of open access by regressing:

```
log(cits) ~ year:journal:multi:article + open
```

where `year:journal:multi` fits the mean number of log citations for each year, journal and accounting for multiauthored papers and reviews, and `open` tests the effect of having published with the open access option. Is the effect positive?

5. Most productive researchers [Optional] Count how many times does each author ID appears in the data. Find the most productive authors in biology, and try extracting their names from the `au` column.