

formulas and contrasts for linear models

Ben Bolker

September 18, 2018



Licensed under the Creative Commons attribution-noncommercial license

(<http://creativecommons.org/licenses/by-nc/3.0/>). Please share & remix

noncommercially, mentioning its origin.

Version: 2018-09-18 23:48:10

Design matrices and model parameterization

Basics

Linear problem: $X\beta$ (solution)

What is X ? **Design matrix** or **parameterization**.

Setting up a parameterization is the same as setting up a hypothesis, unless your question is just “does this variable have some overall effect”?

Wilkinson and Rogers (1973) syntax (R version): (the response variable and tilde are implicit throughout, e.g. $y \sim f1$): see ?formula.

In the following table, f denotes a categorical (factor) variables and x denotes continuous (numeric) variables

formula	interpretation
$f1$	1-way ANOVA (single categorical predictor)
$x1$	linear regression (single continuous predictor: implicit intercept term)
$x1-1$ or $x1+0$	regression through the origin (eliminate intercept) [dangerous]
$f1-1$	ANOVA, without intercepts (see below)
$x1+x2$	multiple regression
$f1+f2$	2-way ANOVA (additive model)
$f1*f2$	2-way ANOVA (with interactions)
$\equiv f1+f2+f1:f2$	
$f1:f2$	interaction model (equivalent to above, but parameterized differently)
$(f1+f2)^2$	crossing to a specified degree
$\equiv f1+f2+f1:f2$	
$(f1+f2)^2-f1$	minus sign (-) removes a term (example)
$\equiv f2+f1:f2$	
$I(x1^2)$	interpret as literal (arithmetic) operator rather than formula
$\text{poly}(x1, x2, n)$	n^{th} order orthogonal polynomial in $x1$ and $x2$
$\text{ns}(x1, n)$	natural spline basis with n knots (needs spline package) [the mgcv package is more powerful for fitting GAMs]

- Wilkinson-Rogers notation has been extended in various (not

always completely compatible) ways by add-on packages

- The bar (|) is used as a grouping variable in various contexts (lattice: specify conditioning (subplots); lme4, nlme: specify grouping variables; pscl: separate models for binary and count models)
- The : shortcut for specifying an interaction works most of the time, but not always (sometimes R interprets it as a shortcut for the seq function): if it fails use interaction() instead

Continuous predictors

Correlation of slope, intercept: interpretation and numerical problem (especially with interactions). Solution: **centering**¹.

Scaling variables (e.g. by standard deviation) also helps with interpretability, and numerics²; ?scale, ?sweep in R; by_2sd in dotwhisker::dwplot

- **Pro:** Simple, sensible, prevents most common misinterpretations, allows interpretation of main effects
- **Con:** Data set-specific. Choice between real units and scaled units. Alternatives to mean-centering and standard-deviation-scaling?

¹ Schielzeth, H. (2010). Simple means to improve the interpretability of regression coefficients. *Methods in Ecology and Evolution* 1, 103–113

² Gelman, A. (2008, July). Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine* 27(15), 2865–2873

Categorical predictors: contrasts

Independent contrasts.

The *contrast matrix* determines what a given row of the design matrix (for level i of a categorical variable) looks like.

If we have a vector of predicted values \bar{y} , the contrast matrix is essentially defined as

$$\bar{y} = C\beta$$

Set contrasts in general via options() or per-factor via contrasts(), or within the model statement, e.g.

```
d <- data.frame(f=c("a", "a", "a", "b", "b", "b"), y=c(1, 1, 1, 3, 3, 3))
coef(lm(y~f, data=d))

## (Intercept)      fb
##           1         2

coef(lm(y~f, data=d, contrasts=list(f="contr.sum")) )

## (Intercept)      f1
##           2       -1
```

Or:

```
contrasts(d$f) <- "contr.sum"
## or (global)
options(contrasts=c("contr.sum", "contr.poly"))
```

Reordering factors: levels, reorder, relevel

```
levels(relevel(d$f, "b"))

## [1] "b" "a"

levels(with(d, reorder(f, y, mean))) ## with() function saves keystrokes

## [1] "a" "b"
```

In general requesting a contrast for an n -level factor gets us only an $n \times (n - 1)$ matrix: the first column is an implicit intercept (all-1) column.

Treatment contrasts (default: “dummy”, “corner-point”) First level of factor (often alphabetical!) is default. `contr.treatment` (default) vs. `contr.SAS` vs. `contr.treatment(n, base=b)`. Full contrast matrix is not orthogonal (i.e. $C^T C$ is not diagonal: we want $C_i^T C_j = 0$ for all $i \neq j$).

```
(cc <- contr.treatment(4))

##      2 3 4
## 1 0 0 0
## 2 1 0 0
## 3 0 1 0
## 4 0 0 1
```

```
t(cc) %*% cc      ## orthogonal
cc <- cbind(1, cc) ## add intercept column
t(cc) %*% cc      ## NOT orthogonal
```

If we want to know the *meaning* of β , it's easiest to invert, or use the *generalized inverse* (`MASS::ginv()`) function ³:

$$\beta = C^{-1} \bar{y}$$

³ Schad, D. J., S. Hohenstein, S. Vasisht, and R. Kliegl (2018, July). How to capitalize on a priori contrasts in linear (mixed) models: A tutorial. *arXiv:1807.10451 [stat]*. *arXiv:1807.10451*

```
solve(cc)

##      1 2 3 4
##      1 0 0 0
## 2 -1 1 0 0
## 3 -1 0 1 0
## 4 -1 0 0 1
```

Also see [this \(simpler\) piece](#) on contrast and inverse-contrast matrices.

Example (from ⁴)

⁴ Gotelli, N. J. and A. M. Ellison (2004). *A Primer of Ecological Statistics*. Sunderland, MA: Sinauer

```
ants <- data.frame(
  place=rep(c("field", "forest"), c(6,4)),
  colonies=c(12, 9, 12, 10,
             9, 6, 4, 6, 7, 10))
```

```
mean(ants$colonies[ants$place=="field"])

## [1] 9.666667

mean(ants$colonies[ants$place=="forest"])

## [1] 6.75

pr <- function(m) printCoefmat(coef(summary(m)), digits=3, signif.stars=FALSE)
pr(lm1 <- lm(colonies~place, data=ants))

##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.667      0.958   10.09   8e-06
## placeforest   -2.917      1.515   -1.92    0.09
```

The (Intercept) row refers to β_1 , which is the mean density in the "field" sites ("field" comes before "forest"). The placeforest row tells us we are looking at the effect of the place variable on the forest level, i.e. the difference between the "forest" and "field" sites. (The only ways we could know that "field" is the baseline site are (1) to remember, or look at `levels(ants$place)` or (2) to notice which level is *missing* from the list of parameter estimates.)

Helmert Orthogonal but less intuitive.

```
(cc <- cbind(1, contr.helmert(4)))

##      [,1] [,2] [,3] [,4]
## 1      1   -1   -1   -1
```

```
## 2    1    1   -1   -1
## 3    1    0    2   -1
## 4    1    0    0    3
```

The contrast matrix is scaled so that $\sum_j |c_{ij}| = N$, where N is the number of columns ...

```
t(cc) %*% cc ## orthogonal
```

Inverse contrast matrix:

```
MASS::fractions(solve(cc))

##      1      2      3      4
## [1,] 1/4    1/4    1/4    1/4
## [2,] -1/2   1/2     0     0
## [3,] -1/6  -1/6    1/3     0
## [4,] -1/12 -1/12 -1/12    1/4
```

β_1 =mean; β_2 =contrast between levels 1 and 2; β_3 =contrast between levels 1&2 and level 3; β_4 =contrast between levels 1-3 and level 4.

```
cfun <- function(contr) {
  pr(update(lm1, contrasts=list(place=contr)))
}
cfun("contr.helmert")

##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.208      0.758   10.83 4.7e-06
## place1         -1.458      0.758   -1.92  0.09
```

Sum-to-zero What if I want to compare the values with the mean (Schielzeth, 2010) ?

```
(cc <- cbind(1, contr.sum(4)))

##      [,1] [,2] [,3] [,4]
## 1      1      1      0      0
## 2      1      0      1      0
## 3      1      0      0      1
## 4      1     -1     -1     -1
```

```
t(cc) %*% cc ## NOT orthogonal (??)
```

```
MASS::fractions(solve(cc))

##      1      2      3      4
## [1,] 1/4 1/4 1/4 1/4
## [2,] 3/4 -1/4 -1/4 -1/4
## [3,] -1/4 3/4 -1/4 -1/4
## [4,] -1/4 -1/4 3/4 -1/4
```

β_1 =mean; β_2 =level 1 vs levels 2–4; β_3 =level 2 vs. levels 1,3, 4;
 β_4 =level 3 vs. levels 1,2, 4
 Note that we don't have level 4.

```
cfun("contr.sum")

##           Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.208      0.758   10.83 4.7e-06
## place1        1.458      0.758    1.92  0.09
```

Same as Helmert contrasts in this example, except for the sign of place1.

No-intercept When we specify a formula with -1 or +0 (with default treatment contrasts) we get an identity matrix for the contrasts: each level has its own parameter.

```
pr(update(lm1, .~. - 1))

##           Estimate Std. Error t value Pr(>|t|)
## placefield    9.667      0.958   10.09 8e-06
## placeforest   6.750      1.174    5.75 0.00043
```

Sometimes clearer (and we get confidence intervals etc. on the predictions for each level), but the hypotheses tested are rarely interesting (is the mean of each level equal to zero?)

More generally, if you want to compute the group means, you can

- Use the predict function:

```
predict(lm1, newdata=data.frame(place=c("field", "forest")), interval="confidence")
```

- Use the effects package:

```
library("effects")
summary(allEffects(lm1))
```

- Use the emmeans package:

```
library("emmeans")
emmeans(lm1, spec=~place)
```

Custom contrasts Can specify contrasts “by hand” (Crawley (2002) gives an example too.)

Example:

```
c_inv <- matrix(
  c(
    ## intercept: mean of all levels
    1/4, 1/4, 1/4, 1/4,
    ## compare first level against the average of the other three
    ## (no symbionts vs {crabs, shrimp, or crabs+shrimp})
    1, -1/3, -1/3, -1/3,
    ## compare second level (crabs) against the third (shrimp)
    0, 1, -1, 0,
    ## compare the average of the second and third levels {crabs, shrimp}
    ## against the fourth level (crabs+shrimp)
    0, 1/2, 1/2, -1),
  nrow=4,
  byrow=TRUE)
## row/column names, for clarity
dimnames(c_inv) <- list(c("intercept", "symb", "C.vs.S", "twosymb"),
  c("none", "C", "S", "CS"))
print(c_inv)

##           none           C           S           CS
## intercept 0.25  0.2500000  0.2500000  0.2500000
## symb      1.00 -0.3333333 -0.3333333 -0.3333333
## C.vs.S    0.00  1.0000000 -1.0000000  0.0000000
## twosymb   0.00  0.5000000  0.5000000 -1.0000000

## test orthogonality
## (zapsmall() gets rid of tiny non-zero values due to rounding error
zapsmall(c_inv %*% t(c_inv))

##           intercept      symb C.vs.S twosymb
## intercept      0.25  0.000000      0      0.0
## symb           0.00  1.333333      0      0.0
## C.vs.S         0.00  0.000000      2      0.0
## twosymb        0.00  0.000000      0      1.5
```

What contrast matrix should we use?

```

cmat <- MASS::fractions(solve(c_inv)) ## fractions() for prettiness
print(cmat)

##      intercept symb C.vs.S twosymb
## none      1      3/4    0        0
## C         1     -1/4  1/2      1/3
## S         1     -1/4 -1/2      1/3
## CS        1     -1/4    0     -2/3

```

In this particular case, the matrix inverse is simply a scaled version of the transpose ...

When assigning this matrix as the contrast matrix for an analysis, we would omit the first column ('contrasts(.) <- cmat[, -1]') since R will add an intercept automatically when we fit the model.

Other useful contrasts

Forward difference:

```

cc <- cbind(mean=1, MASS::contr.sdif(4))
print(MASS::fractions(cc))

##      mean 2-1 3-2 4-3
## 1      1 -3/4 -1/2 -1/4
## 2      1  1/4 -1/2 -1/4
## 3      1  1/4  1/2 -1/4
## 4      1  1/4  1/2  3/4

## find inverse-contrast matrix
MASS::fractions(solve(cc))

##      1  2  3  4
## mean 1/4 1/4 1/4 1/4
## 2-1   -1  1  0  0
## 3-2    0 -1  1  0
## 4-3    0  0 -1  1

## not orthogonal at all

```

Exercise. How would you modify this contrast so the intercept is the value of the first level, rather than the mean?

Interactions

Interactions as *differences in differences*

Interpretation problems: marginality principle/"type III ANOVA"

⁵ Venables, W. N. (1998). Exegeses on linear models. In *1998 International S-PLUS User Conference*, Washington, DC; and Schielzeth, H. (2010). Simple means to improve the interpretability of regression coefficients. *Methods in Ecology and Evolution* 1, 103–113


```

head(d <- expand.grid(F=LETTERS[1:3],f=letters[1:3]))

##    F f
## 1 A a
## 2 B a
## 3 C a
## 4 A b
## 5 B b
## 6 C b

m0 <- model.matrix(~F*f,d)
ff <- solve(m0)
colnames(ff) <- apply(d,1,paste,collapse=".")
ff["FB",] ## contrast between (A,a) and (B,a)

## A.a B.a C.a A.b B.b C.b A.c B.c C.c
## -1  1  0  0  0  0  0  0  0

ff["fb",] ## contrast between (A,a) and (A,b)

## A.a B.a C.a A.b B.b C.b A.c B.c C.c
## -1  0  0  1  0  0  0  0  0

```

```

old.opts <- options(contrasts=c("contr.sum", "contr.poly"))
m <- model.matrix(~F*f,d)
ff <- solve(m)*9
colnames(ff) <- apply(d,1,paste,collapse=".")
ff["F1",] ## contrast between (A,.) and (grand mean)

## A.a B.a C.a A.b B.b C.b A.c B.c C.c
##  2 -1 -1  2 -1 -1  2 -1 -1

ff["f1",] ## contrast between (a,.) and (grand mean)

## A.a B.a C.a A.b B.b C.b A.c B.c C.c
##  2  2  2 -1 -1 -1 -1 -1 -1

options(old.opts) ## reset

```

Exercise: How would you construct a version of `contr.sum` where the first, not the last, level is aliased/dropped?

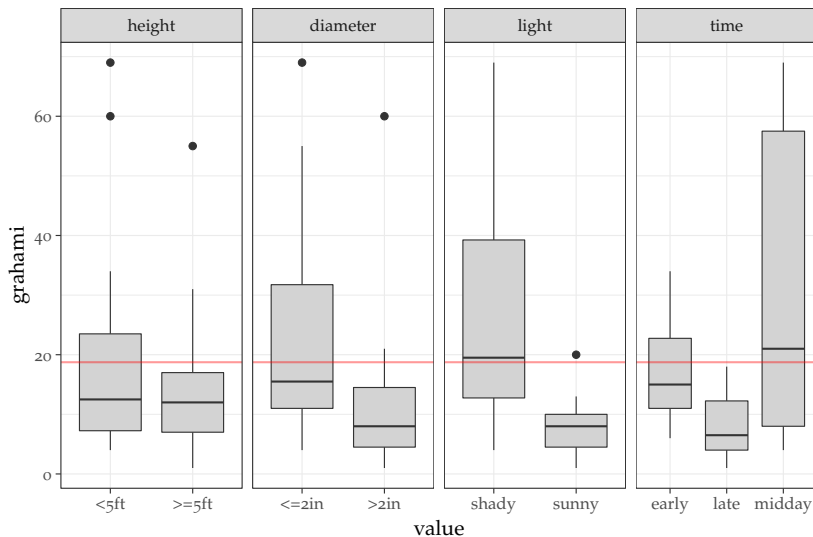
Things get slightly more interesting/complicated when we have more than two levels of a categorical variable. I'll look at some data on lizard perching behaviour, from the `brglm` package (and before that from ⁶, ultimately from ⁷). I'm going to ignore the fact that these data might best be fitted with generalized linear models.

⁶ McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models*. London: Chapman and Hall

⁷ Schoener, T. W. (1970, May). Nonsynchronous Spatial Overlap of Lizards in Patchy Habitats. *Ecology* 51(3), 408–418

```
lizards <- read.csv("lizards.csv")
```

A quick look at the data: response is number of *Anolis grahami* lizards found on perches in particular conditions.



For a moment we're going to just look at the time variable. If we leave the factors as is (alphabetical) then β_1 ="early", β_2 ="late"- β_1 , β_3 ="midday"- β_1 . At the very least, it probably makes sense to change the order of the levels:

```
lizards$time <- factor(lizards$time, levels=c("early", "midday", "late"))
```

All this does (since we haven't changed the baseline factor) is swap the definitions of β_2 and β_3 .

In a linear model, we could also use sum-to-zero contrasts:

```
pr(lm(grahami~time, data=lizards, contrasts=list(time=contr.sum)))
```

```
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept)   19.30      3.53    5.47 2.4e-05
## time1         -1.67      4.93   -0.34  0.74
## time2        12.85      5.10    2.52  0.02
```

Now the (Intercept) parameter is the overall mean: time1 and time2 are the deviations of the first ("early") and second ("midday") groups from the overall mean. (The names are useless: the car package offers a slightly better alternative called `contr.sum`). There are other ways to change the contrasts (i.e., use the `contrasts()` function to change the contrasts for a particular variable permanently, or use `options(contrasts=c("contr.sum", "contr.poly"))` to change the contrasts for *all* variables), but the way shown above may be the most transparent.

We could use `MASS::contr.sdif()`, to parameterize time according to successive differences.

```
library("MASS")
pr(lm(grahami~time,data=lizards,contrasts=list(time=contr.sdif)))
```

##	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	19.30	3.53	5.47	2.4e-05
## time2-1	14.52	8.74	1.66	0.112
## time3-2	-24.02	8.74	-2.75	0.012

You might have particular contrasts in mind (e.g. "control" vs. all other treatments, then "low" vs "high" within treatments), in which case it is probably worth learning how to set contrasts. (We will talk about testing *all pairwise differences later*, when we discuss multiple comparisons. This approach is very common, but not as useful as usually thought.)

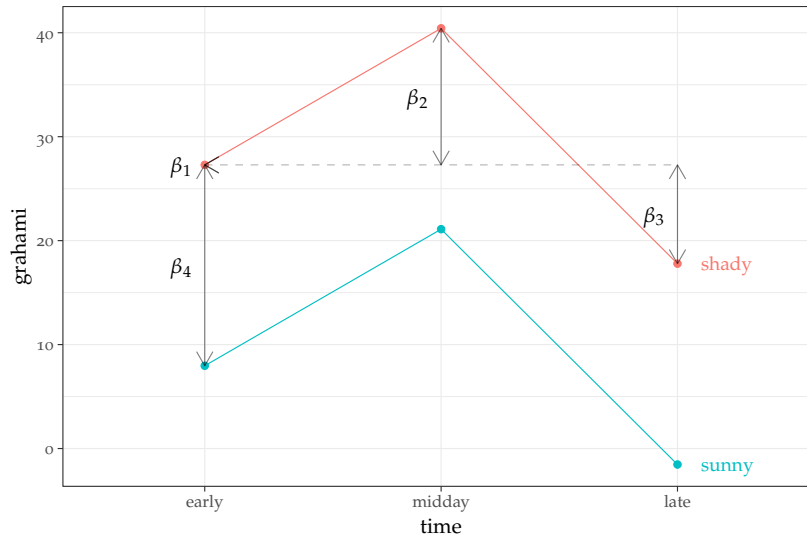
Multiple treatments and interactions

Additive model Let's consider the light variable in addition to time.

```
pr(lmTL1 <- lm(grahami~time+light,data=lizards))
```

##	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	27.29	5.63	4.85	0.00011
## timemidday	13.14	7.11	1.85	0.08010
## timelate	-9.50	6.85	-1.39	0.18174
## lightsunny	-19.32	5.73	-3.37	0.00321

Here's a graphical interpretation of the parameters:

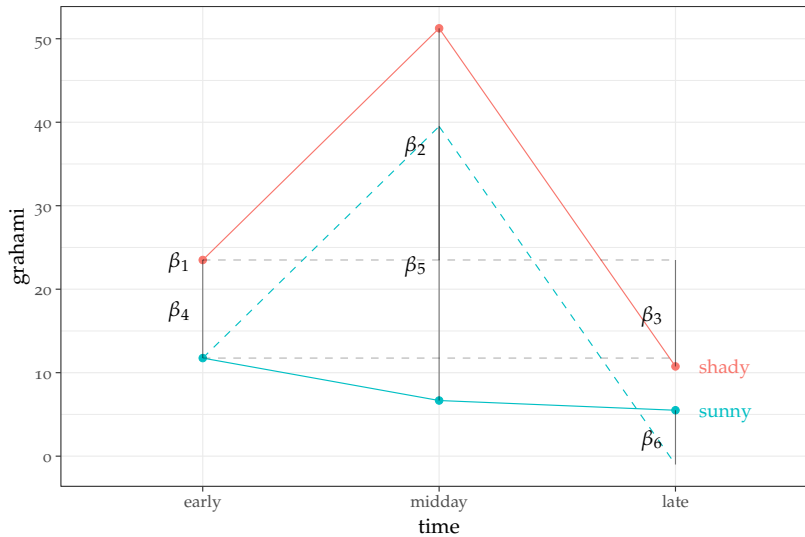


β_1 is the intercept ("early", "sunny"); β_2 and β_3 are the differences from the baseline level ("early") of the *first* variable (time) in the *baseline* level of the other parameter(s) (light="shady"); β_4 is the difference from the baseline level ("sunny") of the *second* variable (light) in the *baseline* level of time ("early").

Now let's look at an interaction model:

```
pr(lmTL2 <- lm(grahami~time*light,data=lizards))
```

	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	23.50	5.38	4.37	0.00042
## timemidday	27.75	7.60	3.65	0.00198
## timelate	-12.75	7.60	-1.68	0.11180
## lightsunny	-11.75	7.60	-1.55	0.14061
## timemidday:lightsunny	-32.83	11.19	-2.93	0.00927
## timelate:lightsunny	6.50	10.75	0.60	0.55343



Parameters β_1 to β_4 have the same meanings as before. Now we also have β_5 and β_6 , labelled "timemidday:lightsunny" and "time-late:lightsunny", which describe the difference between the expected mean value of these treatment combinations based on the additive model (which are $\beta_1 + \beta_2 + \beta_4$ and $\beta_1 + \beta_3 + \beta_4$ respectively) and their actual values.

Now re-do this for sum-to-zero contrasts ... the fits are easy:

```
pr(lmTL1S <- update(lmTL1, contrasts=list(time=contr.sum, light=contr.sum)))
```

##	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	18.84	2.87	6.57	2.7e-06
## time1	-1.21	4.01	-0.30	0.7654
## time2	11.92	4.15	2.87	0.0097
## light1	9.66	2.87	3.37	0.0032

```
pr(lmTL2S <- update(lmTL2, contrasts=list(time=contr.sum, light=contr.sum)))
```

##	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	18.236	2.255	8.09	3.1e-07
## time1	-0.611	3.146	-0.19	0.84830
## time2	10.722	3.271	3.28	0.00444
## light1	10.264	2.255	4.55	0.00028
## time1:light1	-4.389	3.146	-1.39	0.18100
## time2:light1	12.028	3.271	3.68	0.00187

(The intercept doesn't stay exactly the same when we add the interaction because the data are unbalanced: try with(lizards, table(light, time)) ...)

Other refs

- <http://sas-and-r.blogspot.com/2010/10/example-89-contrasts.html>
- see also: `gmodels::fit.contrast`, `rms::contrast.rms` for on-the-fly contrasts
- http://www.ats.ucla.edu/stat/r/library/contrast_coding.htm

References

- Crawley, M. J. (2002). *Statistical Computing: An Introduction to Data Analysis using S-PLUS*. John Wiley & Sons.
- Gelman, A. (2008, July). Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine* 27(15), 2865–2873.
- Gotelli, N. J. and A. M. Ellison (2004). *A Primer of Ecological Statistics*. Sunderland, MA: Sinauer.
- McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models*. London: Chapman and Hall.
- Schad, D. J., S. Hohenstein, S. Vasishth, and R. Kliegl (2018, July). How to capitalize on a priori contrasts in linear (mixed) models: A tutorial. *arXiv:1807.10451 [stat]*. arXiv: 1807.10451.
- Schielzeth, H. (2010). Simple means to improve the interpretability of regression coefficients. *Methods in Ecology and Evolution* 1, 103–113.
- Schoener, T. W. (1970, May). Nonsynchronous Spatial Overlap of Lizards in Patchy Habitats. *Ecology* 51(3), 408–418.
- Venables, W. N. (1998). Exegeses on linear models. In *1998 International S-PLUS User Conference*, Washington, DC.
- Wilkinson, G. N. and C. E. Rogers (1973). Symbolic description of factorial models for analysis of variance. *Applied Statistics* 22(3), 392–399.