BISC-869, Linear Models

February 8, 2021

A relationship between variables involving

- a response variable Y
- explanatory variable(s) X_1, X_2, ...
- normally distributed random errors with equal variance

in the form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \text{error}$$

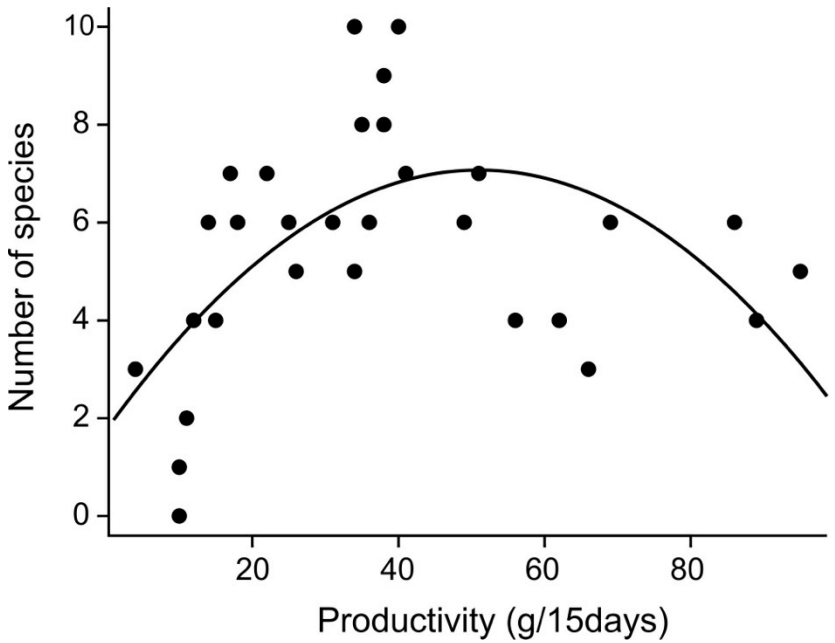
where β_0, β_1, β_2, $\cdots$ are the parameters of the linear model

For example

- fit a mean to data: $Y = \beta_0$
- simple linear regression: $Y = \beta_0 + \beta_1 X$
- multiple regression: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \cdots$
- quadratic regression: $Y = \beta_0 + \beta_1 X + \beta_2 X^2$
- single-factor ANOVA: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots$
- $\cdots$

A linear model needn't be a straight line. For example, the quadratic equation is a linear model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2$$



Note: the term 'factor' in reference to a `lm` usually refer to a predictor (not a 'factor', as we have used in R).

Linear models go by other names:

- Fit a mean
- Linear regression
- Multiple regression
- Fitting different means to two groups
- Single factor ANOVA
- Multi-factor ANOVA
- Analysis of covariance

All can be written in the same form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \text{error}$$

"Linear models" unites these methods into a common framework that

- Provides a common set of tools (`lm` in R)
- Is flexible enough to handle different study designs
- Has tools to estimate parameters (e.g., sizes of effects)
- Is easy to use, even when there are multiple variables
- Better handling of unbalanced designs than traditional ANOVA calculations

Data: The average number of "dee" notes per alarm call by black-capped chickadees presented with a live, perched predator.

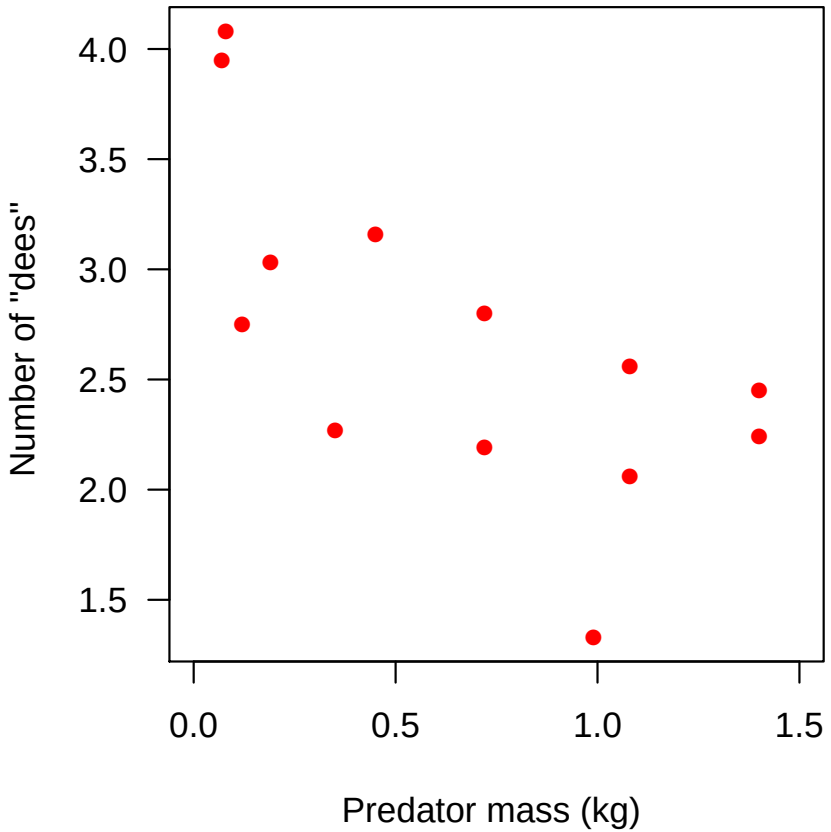
Predator species	Predator body mass (kg)	Number of "dee" notes per call
Northern pygmy-owl	0.07	3.95
Saw-whet owl	0.08	4.08
American kestrel	0.12	2.75
Merlin	0.19	3.03
Short-eared owl	0.35	2.27
Cooper's hawk	0.45	3.16
Prairie falcon	0.72	2.19
Peregrine falcon	0.72	2.80
Great horned owl	1.40	2.45
Rough-legged hawk	0.99	1.33
Gyrfalcon	1.40	2.24
Red-tailed hawk	1.08	2.56
Great gray owl	1.08	2.06

Templeton, C. N., E. Greene, and K. Davis. 2005. *Science* 308: 1934-1937.

Data: The average number of "dee" notes per alarm call by black-capped chickadees presented with a live, perched predator.

Predator species	Predator body mass (kg)	Number of "dee" notes per call
Northern pygmy-owl	0.07	3.95
Saw-whet owl	0.08	4.08
American kestrel	0.12	2.75
Merlin	0.19	3.03
Short-eared owl	0.35	2.27
Cooper's hawk	0.45	3.16
Prairie falcon	0.72	2.19
Peregrine falcon	0.72	2.80
Great horned owl	1.40	2.45
Rough-legged hawk	0.99	1.33
Gyrfalcon	1.40	2.24
Red-tailed hawk	1.08	2.56
Great gray owl	1.08	2.06

Templeton, C. N., E. Greene, and K. Davis. 2005. *Science* 308: 1934-1937.



Example: Simple linear regression with no predictors

Linear model for simple linear regression with no predictors

$$Y = \beta_0$$

There is only one parameter in this equation:

β_0: intercept

The model in plain language:

$$dees = intercept$$

In R this is written as:

```
dees~1
```

Our data-frame, `dd`:

```
head(dd)
```

```
                  pred mass dees
1 Northern pygmy-owl 0.07 3.95
2       Saw-whet owl 0.08 4.08
3   American kestrel 0.12 2.75
4             Merlin 0.19 3.03
5    Short-eared owl 0.35 2.27
6       Coopers Hawk 0.45 3.16
```

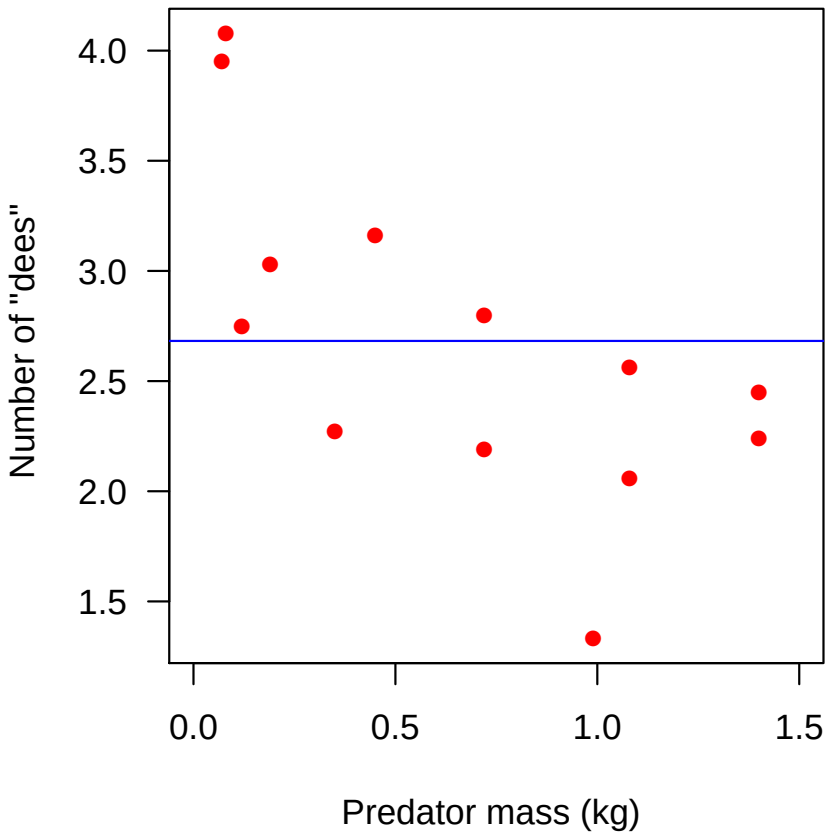
To fit the model:

```
out <- lm(dees~1, data=dd)
summary(out)
```

```
Call:
lm(formula = dees ~ 1, data = dd)

Residuals:
    Min      1Q  Median      3Q     Max
-1.3523 -0.4423 -0.1223  0.3477  1.3977

Coefficients:
            Estimate Std. Error t value     Pr(>|t|)
(Intercept)   2.6823     0.2091   12.83 0.0000000229 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7539 on 12 degrees of freedom
```

Don't pay attention to P-values from `summary`.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.6823	0.2091	12.83	0.0000000229 ***



And, as a quick check:

```
mean(dd$dees)
```

```
2.682308
```

Linear model for simple linear regression

$$Y = \beta_0 + \beta_1 X$$

Here, we have an intercept and a slope (a 'fixed effect'):

β_0: intercept

β_1: slope

The model in plain lenguage:

$$dees = intercept + mass$$

In R, the intercept is implicit and doesn't need to be in the model formulation:

```
dees~mass
```

but we could also write this as

```
dees~1+mass
```

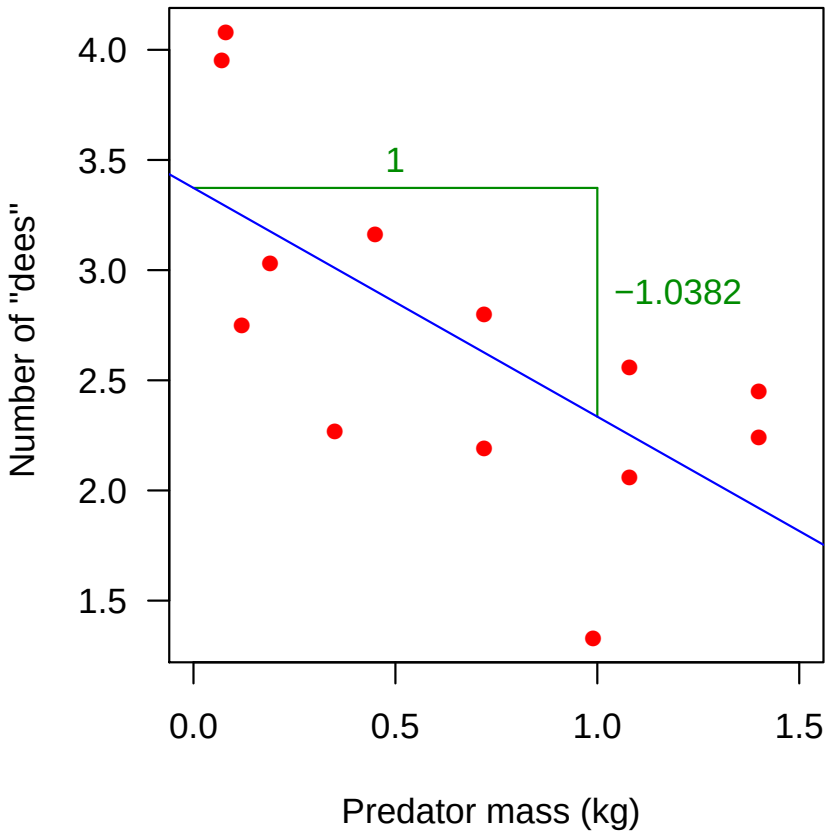
Run the model:

```
out <- lm(dees~mass, data=dd)
summary(out)
```

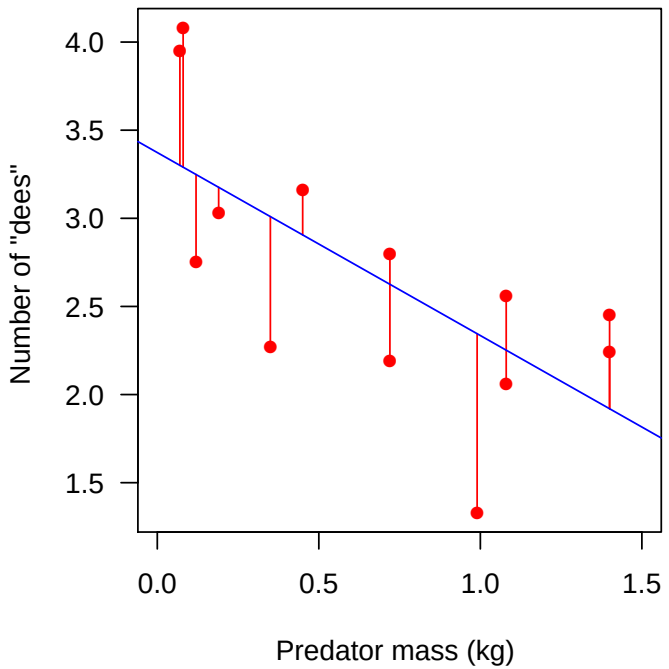
```
Call:
lm(formula = dees ~ mass, data = dd)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0153 -0.4356  0.1744  0.3204  0.7899

Coefficients:
            Estimate Std. Error t value    Pr(>|t|)
(Intercept)   3.3731     0.2776  12.149 0.000000102 ***
mass         -1.0382     0.3402  -3.051       0.011 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5795 on 11 degrees of freedom
Multiple R-squared:  0.4584,  Adjusted R-squared:  0.4092
F-statistic: 9.311 on 1 and 11 DF,  p-value: 0.01102
```

Example: Simple linear regression with one predictor

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	3.3731	0.2776	12.149	0.000000102 ***
mass	-1.0382	0.3402	-3.051	0.011 *



Residuals



To view the residuals:

```
resid(out)
```

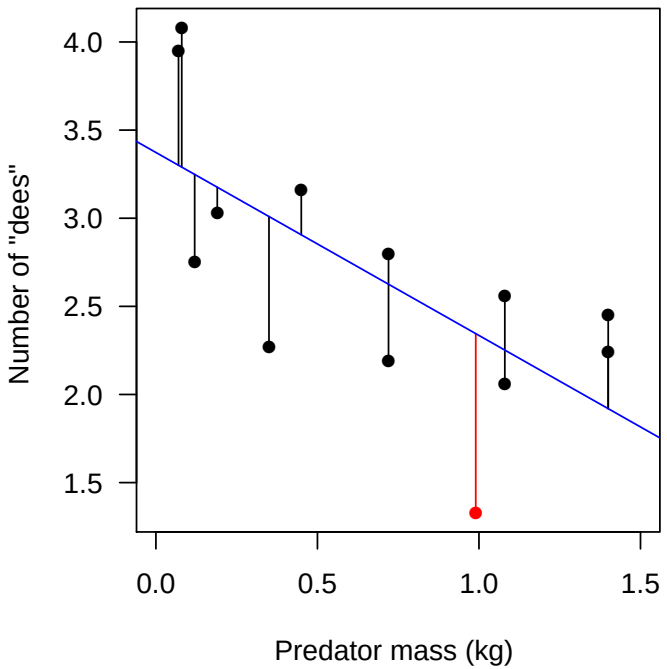
where `out` is the saved output from the `lm` command.

What's happening 'under the hood'?

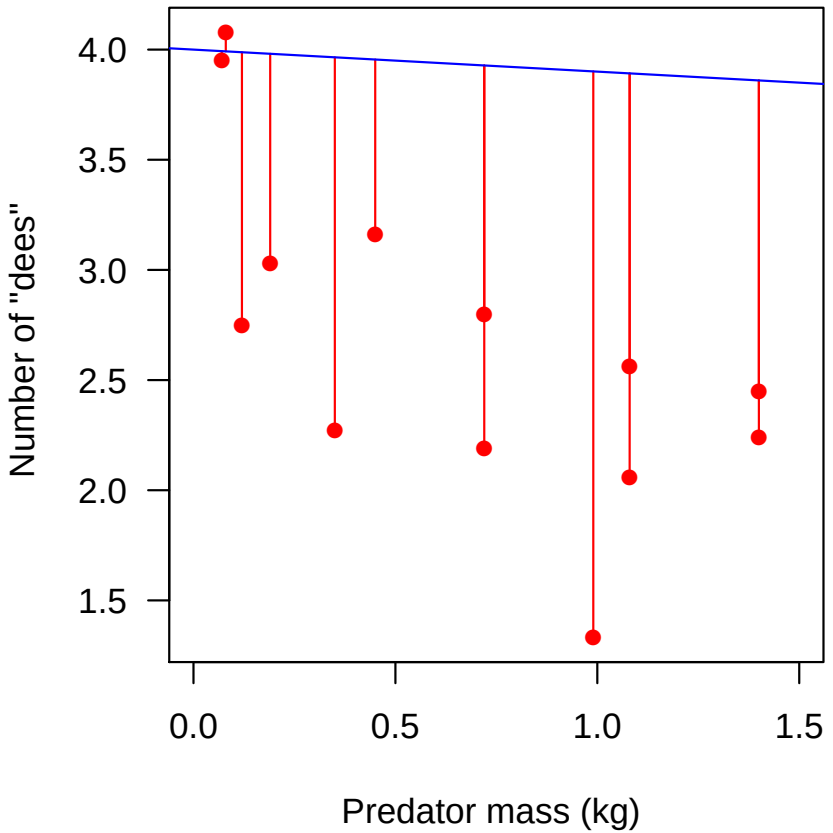
We can write each data point as a linear combination of the model parameter estimates and the residual:

$$\underset{\text{dees}}{\begin{bmatrix}3.95\\4.08\\2.75\\3.03\\2.27\\3.16\\2.19\\2.80\\2.45\\1.33\\2.24\\2.56\\2.06\end{bmatrix}} = \beta_0 * \underset{\text{dummy}}{\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}} + \beta_1 * \underset{\text{mass}}{\begin{bmatrix}0.07\\0.08\\0.12\\0.19\\0.35\\0.45\\0.72\\0.72\\1.40\\0.99\\1.40\\1.08\\1.08\end{bmatrix}} + \underset{\text{residual}}{\begin{bmatrix}0.65\\0.79\\-0.50\\-0.15\\-0.74\\0.25\\-0.44\\0.17\\0.53\\-1.02\\0.32\\0.31\\-0.19\end{bmatrix}}$$

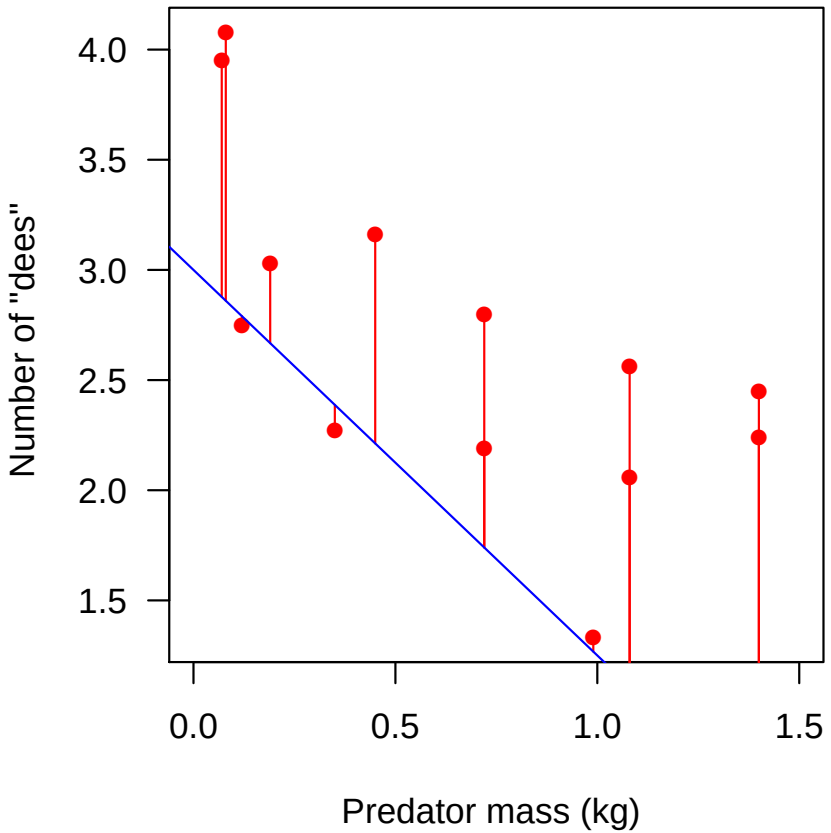
Note: $\beta_0 = 3.3731$, $\beta_1 = -1.0382$



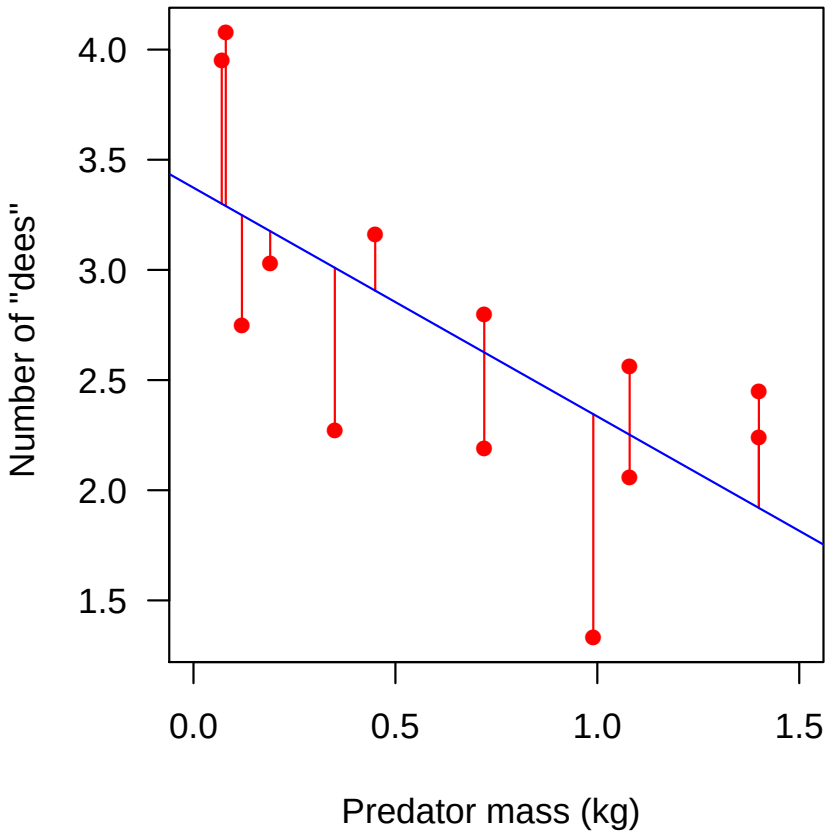
How are β_0 and β_1 chosen? R uses 'least squares'. In other words, R finds the values of β_0 and β_1 that minimize the sum of squared residuals, $\sum_i (\text{residual}_i)^2$.



How are β_0 and β_1 chosen? R uses 'least squares'. In other words, R finds the values of β_0 and β_1 that minimize the sum of squared residuals, $\sum_i (\text{residual}_i)^2$.



How are β_0 and β_1 chosen? R uses 'least squares'. In other words, R finds the values of β_0 and β_1 that minimize the sum of squared residuals, $\sum_i (\text{residual}_i)^2$.



Use `anova` or `drop1` to test hypothesis.

```
anova(out)
```

```
Analysis of Variance Table

Response: dees
          Df Sum Sq Mean Sq F value  Pr(>F)
mass       1 3.1268 3.12683  9.3106 0.01102 *
Residuals 11 3.6942 0.33584
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
drop1(out, test='F')
```

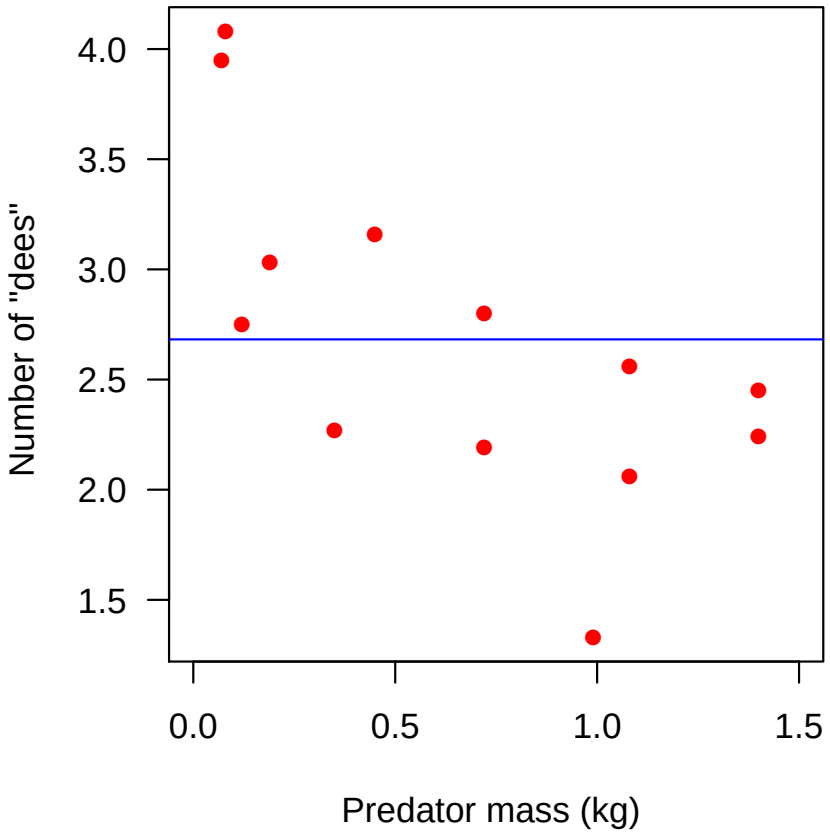
```
Single term deletions

Model:
dees ~ mass
       Df Sum of Sq    RSS      AIC F value  Pr(>F)
<none>              3.6942 -12.3564
mass    1    3.1268 6.8210  -6.3842  9.3106 0.01102 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

These tests fit and compare two models. Specifically, they compare a reduced model (representing the null hypothesis) to a full model (representing the alternative hypothesis). The reduced model contains a subset of terms present in the full model (it is "nested"). An F-test tests whether the full model fits the data significantly better than the reduced model.

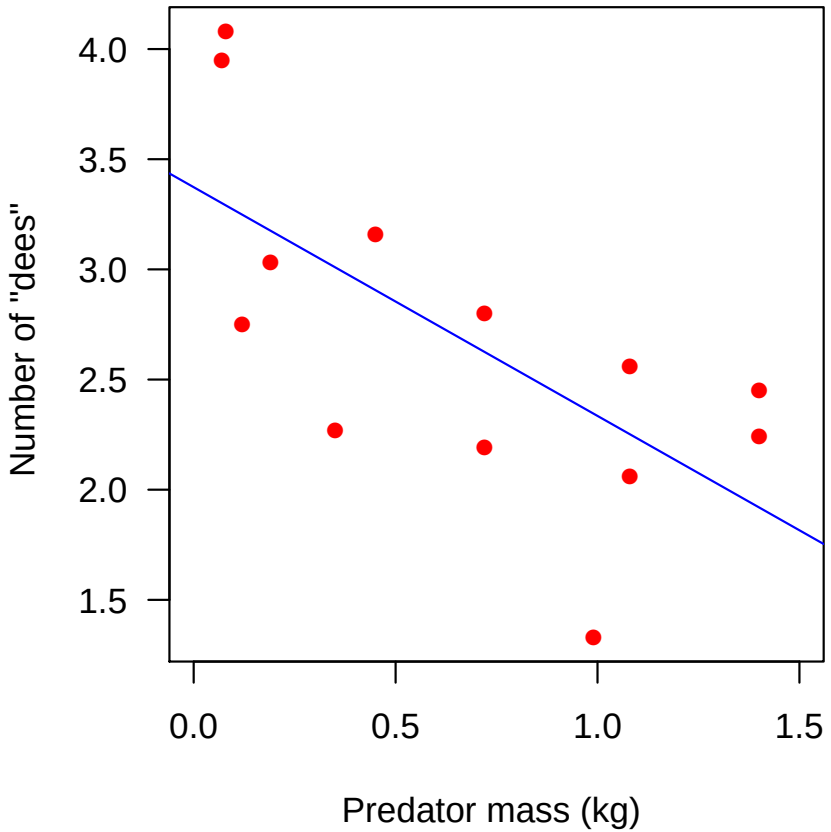
Note that with `anova`, the order you enter the terms into the model (if you have more than one predictor) matters (more on this later).

Visually, these tests are comparing the below two models:

Reduced model: `dees~1`



Full model: `dees~mass`



Data: Effects of latitude and elevation on ant species richness. $n = 22$ forest plots.

Gotelli, N. J. & Ellison, A. M. 2002. Biogeography at a regional scale: determinants of ant species density in bogs and forests of New England. Ecology, 83, 1604–1609.

$$\ln(\textit{nspecies}) = \beta_0 + \beta_1 \cdot \textit{latitude} + \beta_2 \cdot \textit{elevation} + \beta_3 \cdot (\textit{latitude} \times \textit{elevation})$$

Parameters in this model:

- β_0: intercept
- β_1: slope for latitude
- β_2: slope for elevation
- β_3: slope for interaction

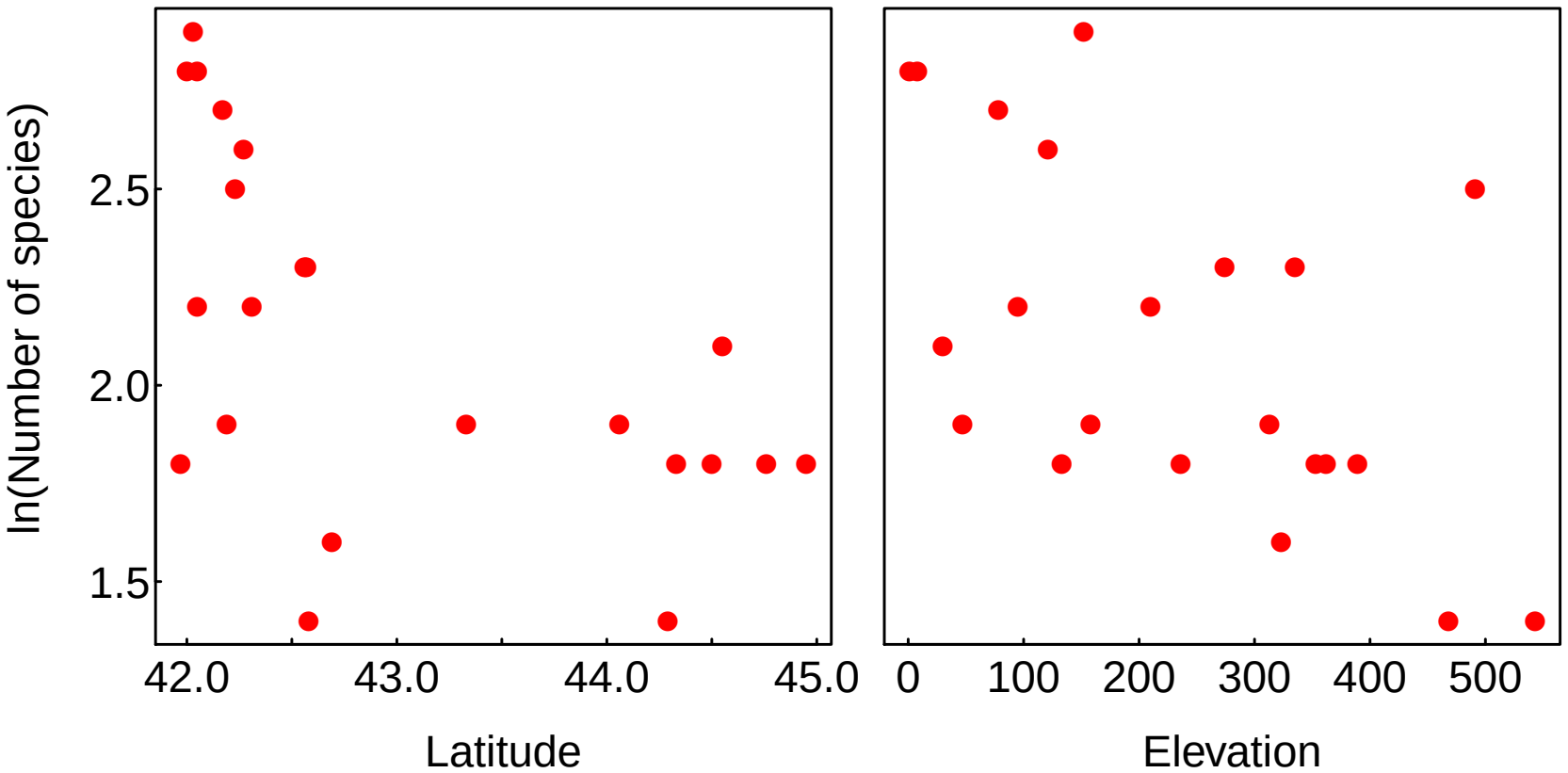
```
head(dd)
```

	nsp	latitude	elevation
1	6	41.97	389
2	16	42.00	8
3	18	42.03	152
4	16	42.05	1
5	9	42.05	210
6	15	42.17	78

Note: sample size too small to fit so many parameters, but for this example let's keep going anyway.

$$\ln(\textit{nspecies}) = \beta_0 + \beta_1 \cdot \textit{latitude} + \beta_2 \cdot \textit{elevation} + \beta_3 \cdot (\textit{latitude} \times \textit{elevation})$$

With multiple predictors, plotting isn't quite as simple (could use 3D plots, or multiple 2D plots).



To fit the model (with interaction):

```
out <- lm(log(nsp)~latitude*elevation, data=dd)
summary(out)
```

```
Call:
lm(formula = log(nsp) ~ latitude * elevation, data = dd)

Residuals:
     Min       1Q   Median       3Q      Max
-0.59789 -0.19520  0.07043  0.15743  0.59422

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       12.2494191  4.9830250   2.458   0.0243 *
latitude          -0.2284237  0.1166225  -1.959   0.0658 .
elevation         -0.0066702  0.0185131  -0.360   0.7228
latitude:elevation  0.0001236  0.0004320   0.286   0.7781
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3116 on 18 degrees of freedom
Multiple R-squared:  0.5805,  Adjusted R-squared:  0.5106
F-statistic: 8.304 on 3 and 18 DF,  p-value: 0.001117
```

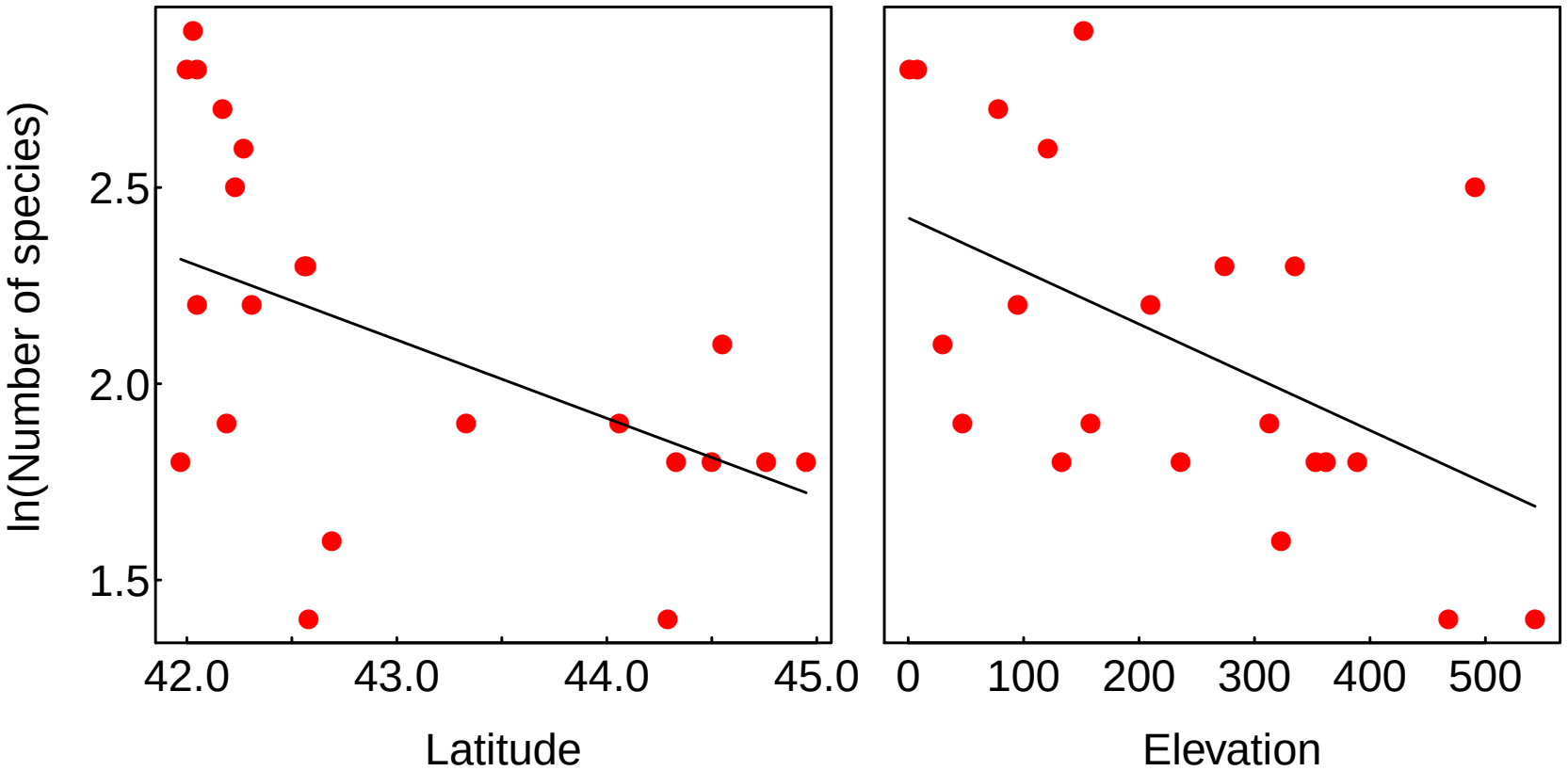
Example: Multiple regression

log(nsp)		dummy		lat		elev		lat × elev		residual
1.8	$= \beta_0 *$	1	$+\beta_1 *$	41.97	$+\beta_2 *$	389	$+\beta_3 *$	16326.33	$+$	−0.29
2.8		1		42.00		8		336.00		0.15
2.9		1		42.03		152		6388.56		0.47
2.8		1		42.05		1		42.05		0.16
2.2		1		42.05		210		8830.50		−0.14
2.7		1		42.17		78		3289.26		0.19
1.9		1		42.19		47		1982.93		−0.65
2.5		1		42.23		491		20734.93		0.60
2.6		1		42.27		121		5114.67		0.18
2.2		1		42.31		95		4019.45		−0.25
2.3		1		42.56		274		11661.44		0.16
2.3		1		42.57		335		14260.95		0.24
1.4		1		42.58		543		23120.94		−0.36
1.6		1		42.69		323		13788.87		−0.45
1.9		1		43.33		158		6846.14		−0.25
1.9		1		44.06		313		13790.78		0.10
1.4		1		44.29		468		20727.72		−0.17
1.8		1		44.33		362		16047.46		0.11
1.8		1		44.50		236		10502.00		−0.01
2.1		1		44.55		30		1336.50		0.06
1.8		1		44.76		353		15800.28		0.18
1.8		1		44.95		133		5978.35		−0.03

$\beta_0 = 12.25$, $\beta_1 = -0.23$, $\beta_2 = -0.0067$, $\beta_3 = 0.00012$

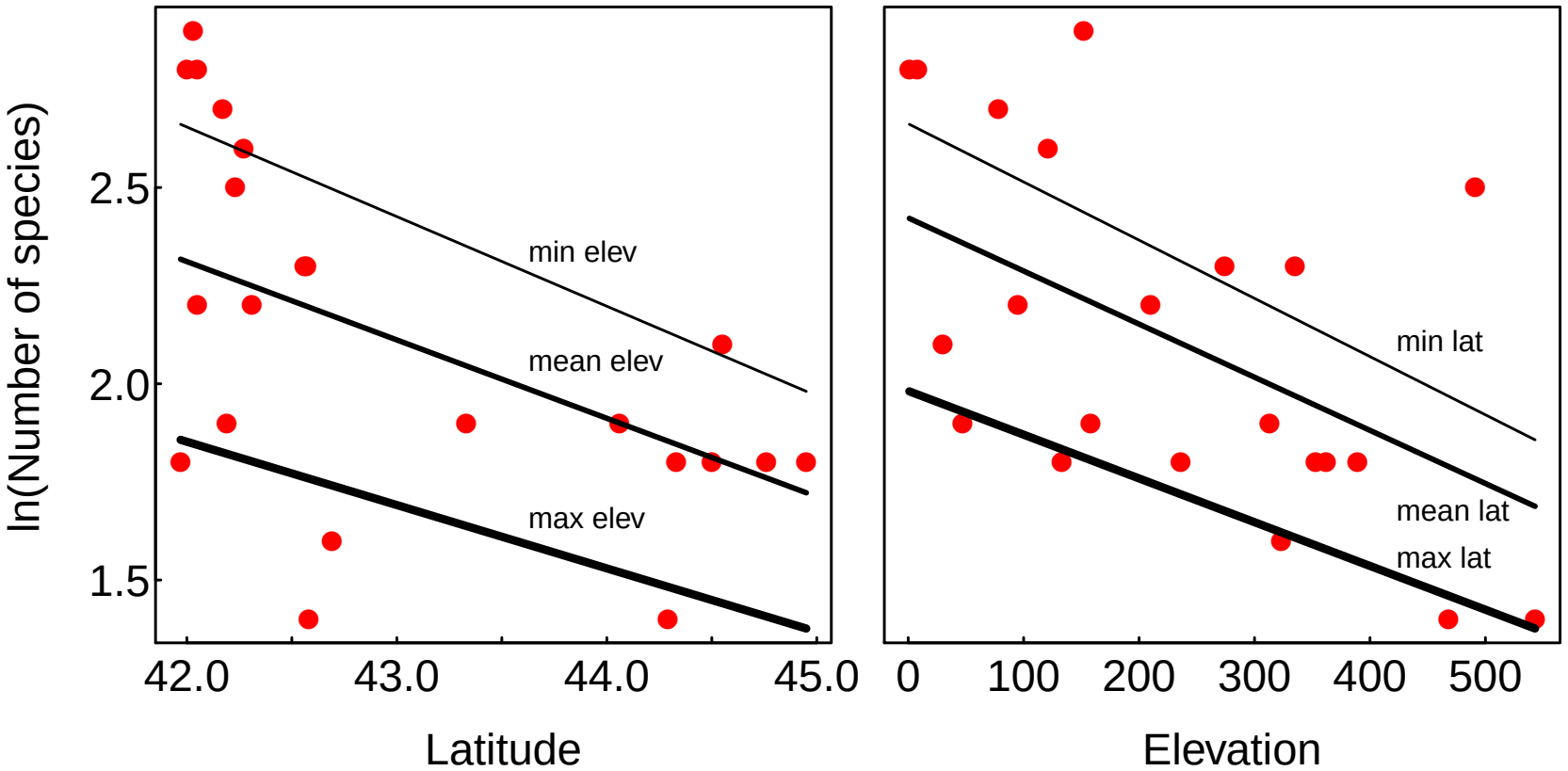
Note: Unlike the other columns, the values in the 'residual' vector are calculated from the fitted model.

Plotting best fit lines



But... how did I plot these? What assumption(s) did I have to make?

Plotting best fit lines



You need to specify the values for all other predictors in the model in order to plot a curve.

What would an "interaction" between these variables look like?

Let's use `drop1` to test hypotheses:

```
drop1(out, test='F')
```

```
Single term deletions

Model:
log(nsp) ~ latitude * elevation
                   Df Sum of Sq    RSS     AIC F value Pr(>F)
<none>                          1.7482 -47.714
latitude:elevation  1 0.0079494 1.7562 -49.614  0.0818 0.7781
```

- `drop1` will not test main effects if an interaction is included. In general, interaction effects do not make sense in the absence of their main effects.
- Here, we find no evidence for a significant interaction effect. Therefore, we remove that term using the `update` command, and run `drop1` again.

```
out2 <- update(out,~.-latitude:elevation)
drop1(out2, test='F')
```

```
Single term deletions

Model:
log(nsp) ~ latitude + elevation
          Df Sum of Sq    RSS     AIC F value   Pr(>F)
<none>                  1.7562 -49.614
latitude   1   0.95781 2.7140 -42.038  10.363 0.004517 **
elevation  1   1.02220 2.7784 -41.522  11.059 0.003555 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

We have evidence that both main effects are 'significant'.

This last step could equivalently be accomplished by re-running a simpler model, without the interaction, and then using `drop1` on that.

```
out <- lm(log(nsp)~latitude+elevation, data=dd)
drop1(out, test='F')
```

Consider a model that includes two factors A and B; there are, therefore, two main effects, and an interaction, $A \times B$. Let's represent the full model by $SS(A, B, AB)$.

Lets define the incremental sum of squares
$SS(AB|A, B) = SS(A, B, AB) – SS(A, B)$
$SS(A|B, AB) = SS(A, B, AB) – SS(B, AB)$
and so on...

Type I (also called "sequential" sum of squares)

$SS(A)$ for factor A
$SS(B|A)$ for factor B
$SS(AB|B, A)$ for interaction AB

`anova` produces these - order matters!

Type III

$SS(A|B, AB)$ for factor A
$SS(B|A, AB)$ for factor B

`drop1` produces these - order doesn't matter.

$\cdots$ but, it means that testing for significant main effects doesn't make sense if there is an interaction $\cdots$

https://mcfromnz.wordpress.com/2011/03/02/anova-type-iiiiii-ss-explained/

For our example:

```
out <- lm(log(nsp)~latitude+elevation, data=dd)
drop1(out, test='F')
```

```
Single term deletions

Model:
log(nsp) ~ latitude + elevation
          Df Sum of Sq    RSS     AIC F value   Pr(>F)
<none>                 1.7562 -49.614
latitude   1   0.95781 2.7140 -42.038  10.363 0.004517 **
elevation  1   1.02220 2.7784 -41.522  11.059 0.003555 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
out <- lm(log(nsp)~latitude+elevation, data=dd)
anova(out)
```

```
Analysis of Variance Table

Response: log(nsp)
          Df Sum Sq Mean Sq F value   Pr(>F)
latitude   1 1.3894 1.38937  15.031 0.001015 **
elevation  1 1.0222 1.02220  11.059 0.003555 **
Residuals 19 1.7562 0.09243
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
out <- lm(log(nsp)~elevation+latitude, data=dd)
anova(out)
```

```
Analysis of Variance Table

Response: log(nsp)
          Df  Sum Sq Mean Sq F value    Pr(>F)
elevation  1 1.45375 1.45375  15.728 0.0008283 ***
latitude   1 0.95781 0.95781  10.363 0.0045166 **
Residuals 19 1.75617 0.09243
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For our example (with interaction):

```
out <- lm(log(nsp)~latitude*elevation, data=dd)
drop1(out, test='F')
```

```
Single term deletions

Model:
log(nsp) ~ latitude * elevation
                   Df Sum of Sq    RSS     AIC F value Pr(>F)
<none>                          1.7482 -47.714
latitude:elevation  1 0.0079494 1.7562 -49.614  0.0818 0.7781
```

```
out <- lm(log(nsp)~latitude*elevation, data=dd)
anova(out)
```

```
Analysis of Variance Table

Response: log(nsp)
                 Df  Sum Sq Mean Sq F value   Pr(>F)
latitude          1 1.38937 1.38937 14.3052 0.001364 **
elevation         1 1.02220 1.02220 10.5247 0.004503 **
latitude:elevation  1 0.00795 0.00795  0.0818 0.778074
Residuals        18 1.74822 0.09712
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
out <- lm(log(nsp)~elevation*latitude, data=dd)
anova(out)
```

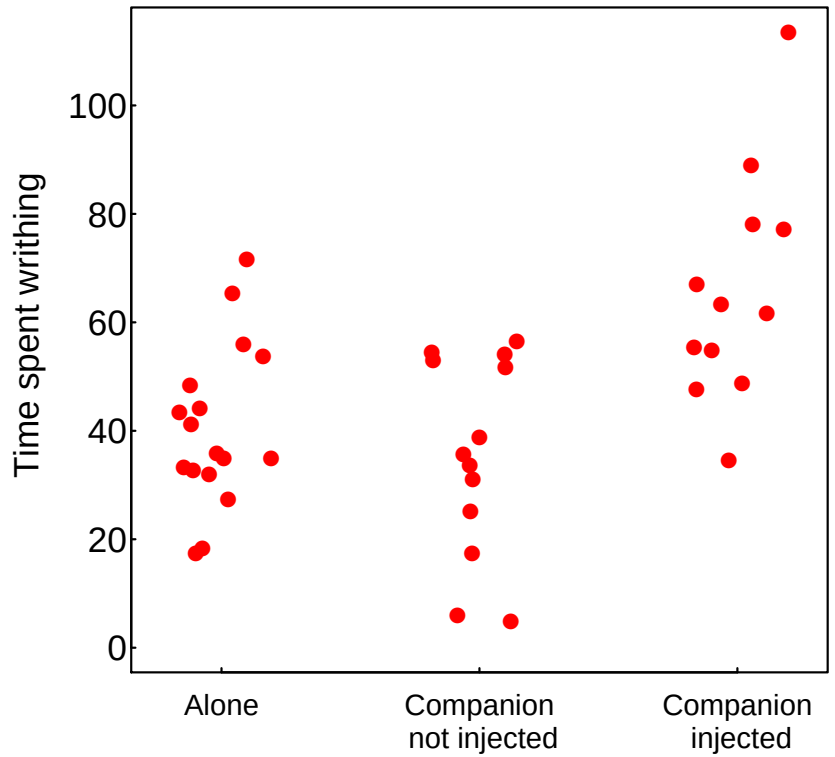
```
Analysis of Variance Table

Response: log(nsp)
                 Df  Sum Sq Mean Sq F value   Pr(>F)
elevation         1 1.45375 1.45375 14.9681 0.001125 **
latitude          1 0.95781 0.95781  9.8618 0.005657 **
elevation:latitude  1 0.00795 0.00795  0.0818 0.778074
Residuals        18 1.74822 0.09712
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note: the P-values from `anova` for latitude and elevation are not the same as they were in the models without the interaction. This is because the F-value is calculated as "`Mean Sq` for parameter of interest" / "`Mean Sq` for Residuals", and the latter changes as more parameters are added to the model.

Model simplification: The interaction term in the model was not significant. Can we drop it and refit?

- The temptation is strong to drop non-significant terms from models, to find a "minimum adequate model" or to provide more power to test remaining effects.
- Dropping a term when $P > 0.05$ involves "accepting" a null hypothesis as true. Why is this a good idea? Remaining P-values become "exploratory."
- Later, we will cover the topic of model selection - how to choose the "best" model using less arbitrary criteria for what is "best".
- `drop1` vs `anova` represent different approaches. Downside to `anova` is that you have to decide on the order of importance *a priori*. Downside to `drop1` - your final analysis doesn't follow your design.

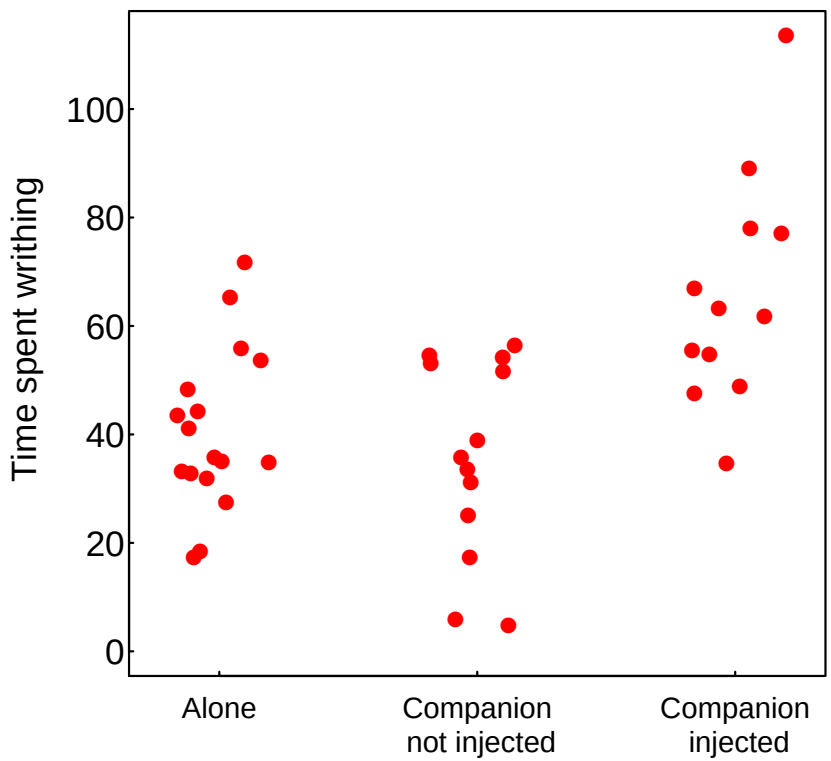
The percentage of time that male mice given an injection to cause mild pain spent "writhing" in different familiar-companion treatments.
Data simulated based on: *Langford, D. J.,et al. 2006. Science 312: 1967-1970*



Note: don't worry about ANOVA/ANCOVA notation, if this is unfamiliar.

ANOVA is fundamentally the same as linear regression

- There's a response variable, a constant, an explanatory variable.
- `out <- lm(writhing~treatment, data=dd)`
- The only difference from previous examples is that the explanatory variable is *categorical*.

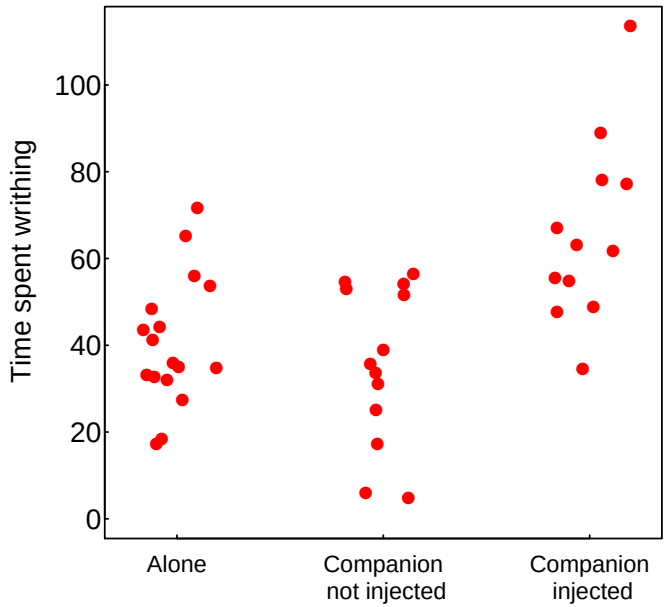


```
head(dd)
```

```
writhing  treatment
33.59252 injected.n
77.09400 injected.y
41.17232      alone
54.50535 injected.n
55.44013 injected.y
31.97051      alone
```

```
out <- lm(writhing~treatment, data=dd)
summary(out)
```

```
Call:
lm(formula = writhing ~ treatment, data = dd)

Residuals:
    Min      1Q  Median      3Q     Max
-31.290 -10.476  -2.319  12.877  47.620

Coefficients:
                   Estimate Std. Error t value        Pr(>|t|)
(Intercept)          40.620      4.341   9.357 0.0000000000162 ***
treatmentinjected.n  -5.066      6.595  -0.768        0.446978
treatmentinjected.y  25.288      6.749   3.747        0.000579 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 17.9 on 39 degrees of freedom
Multiple R-squared:  0.3478,  Adjusted R-squared:  0.3144
F-statistic:  10.4 on 2 and 39 DF,  p-value: 0.00024
```



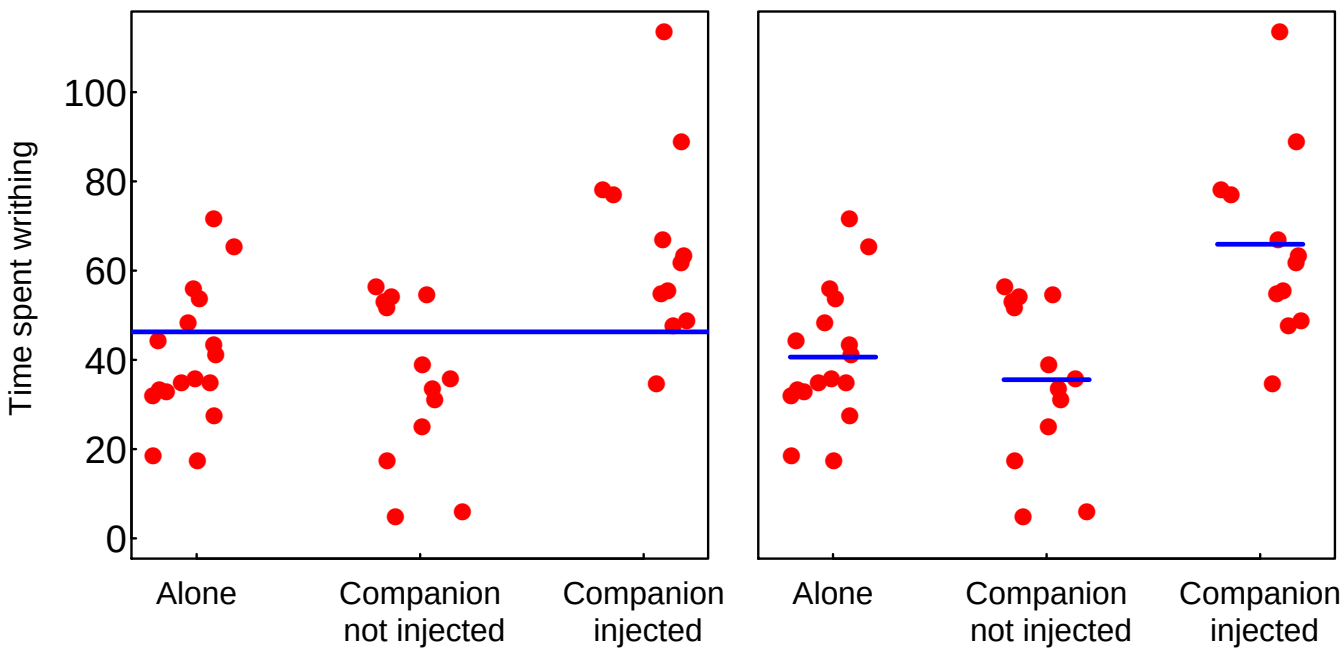
What do these estimates mean?

Let's look at the `drop1`.

```
drop1(out, test='F')
```

```
Single term deletions

Model:
writhing ~ treatment
          Df Sum of Sq   RSS    AIC F value  Pr(>F)
<none>                 12495 245.21
treatment  2    6663.4 19158 259.16  10.399 0.00024 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

`drop1` compares a model without 'treatment' to one with it



In order to understand the coefficients, let's look at the "model matrix":

writhing	dummy	treatalone*	treatinjected.n	treatinjected.y
33.2	1	1	0	0
17.4	1	1	0	0
44.2	1	1	0	0
41.2	1	1	0	0
34.9	1	1	0	0
⋮	⋮	⋮	⋮	⋮
31.1	1	0	1	0
5.9	1	0	1	0
54.5	1	0	1	0
51.7	1	0	1	0
33.6	1	0	1	0
⋮	⋮	⋮	⋮	⋮
89.0	1	0	0	1
34.6	1	0	0	1
48.8	1	0	0	1
77.1	1	0	0	1
63.2	1	0	0	1

Use `model.matrix(out)` to view this matrix.

*R leaves out the first level for each categorical variable (in this case, 'alone') in order to avoid redundancy.

$$
\begin{array}{cccccccccc}
\text{writhing} & & \text{dummy} & & \text{treat}\textcolor{blue}{\text{injected.n}} & & \text{treat}\textcolor{blue}{\text{injected.y}} & & \text{residuals} \\
\begin{bmatrix} 33.2 \\ 17.4 \\ 44.2 \\ 41.2 \\ 34.9 \\ \vdots \\ 31.1 \\ 5.9 \\ 54.5 \\ 51.7 \\ 33.6 \\ \vdots \\ 89.0 \\ 34.6 \\ 48.8 \\ 77.1 \\ 63.2 \end{bmatrix} & = \beta_0 * & \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} & + \beta_1 * & \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \vdots \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} & + \beta_2 * & \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \\ \textcolor{blue}{1} \end{bmatrix} & + & \begin{bmatrix} \\ \\ \\ \\ \\ \vdots \\ \\ \\ \\ \\ \\ \vdots \\ \\ \\ \\ \\ \end{bmatrix}
\end{array}
$$

$$\underset{\text{writhing}}{\begin{bmatrix}33.2\\17.4\\44.2\\41.2\\34.9\\\vdots\\31.1\\5.9\\54.5\\51.7\\33.6\\\vdots\\89.0\\34.6\\48.8\\77.1\\63.2\end{bmatrix}} = \beta_0 * \underset{\text{dummy}}{\begin{bmatrix}1\\1\\1\\1\\1\\\vdots\\1\\1\\1\\1\\1\\\vdots\\1\\1\\1\\1\\1\end{bmatrix}} + \beta_1 * \underset{\text{treatinjected.n}}{\begin{bmatrix}0\\0\\0\\0\\0\\\vdots\\1\\1\\1\\1\\1\\\vdots\\0\\0\\0\\0\\0\end{bmatrix}} + \beta_2 * \underset{\text{treatinjected.y}}{\begin{bmatrix}0\\0\\0\\0\\0\\\vdots\\0\\0\\0\\0\\0\\\vdots\\1\\1\\1\\1\\1\end{bmatrix}} + \underset{\text{residuals}}{\begin{bmatrix}\\\\\\\\\\\vdots\\\\\\\\\\\vdots\\\\\\\\\\\end{bmatrix}}$$

```
Coefficients:
                    Estimate Std. Error
(Intercept)           40.620      4.341
treatmentinjected.n   -5.066      6.595
treatmentinjected.y   25.288      6.749
```

For example, for a data-point in the 'injected.n' treatment, we have

$5.9 = \beta_0 * 1 + \beta_1 * 1 + \beta_2 * 0 + \text{residual[i]}$

So, what do the `summary()` coefficients mean?

The linear model being fitted is:

subjects in 'alone' group $\rightarrow$ writhing $= \beta_0 +$ residual

subjects in 'injection.n' group $\rightarrow$ writhing $= \beta_0 + \beta_1 +$ residual

subjects in 'injection.y' group $\rightarrow$ writhing $= \beta_0 + \beta_2 +$ residual

Stare at this long enough and you'll realize that:

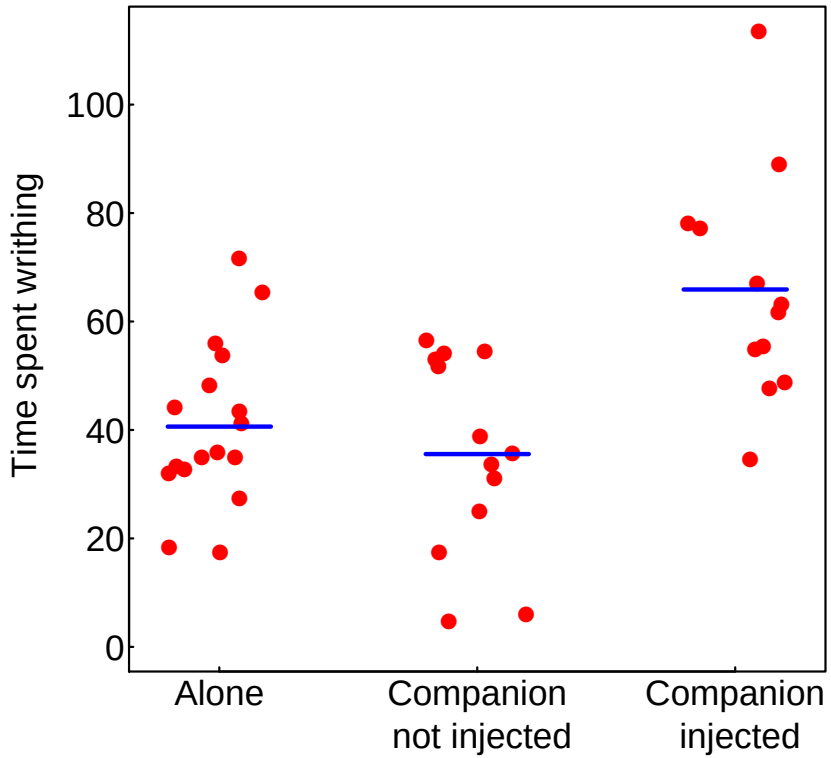
- β_0 is the mean of the 'alone' (control) group
- β_1 is the difference between 'injection.n' and control groups
- β_2 is the difference between 'injection.y' and control groups

```
Coefficients:
                  Estimate Std. Error
(Intercept)         40.620      4.341
treatmentinjected.n -5.066      6.595
treatmentinjected.y 25.288      6.749
```

40.62 is an estimate of β_0

-5.06 is an estimate of β_1

25.28 is an estimate of β_2




```
Coefficients:
                   Estimate Std. Error
(Intercept)          40.620      4.341
treatmentinjected.n  -5.066      6.595
treatmentinjected.y  25.288      6.749
```

40.62 is an estimate of β_0

-5.06 is an estimate of β_1

25.28 is an estimate of β_2

How does `drop1` test a categorical predictor with more than two values? The reduced model *drops all columns* corresponding to that predictor. In this example, the three levels of treatment are coded by two dummy indicator variables, both of which are dropped in the reduced model.

```
drop1(out, test='F')
```

```
Single term deletions

Model:
writhing ~ treatment
          Df Sum of Sq   RSS    AIC F value  Pr(>F)
<none>                 12495 245.21
treatment  2    6663.4 19158 259.16  10.399 0.00024 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Here `Df=2`, because there are two estimated parameters being removed when this single predictor is being dropped from the model.

`emmeans()` (which stands for "estimated marginal means") will calculate fitted means under the specific model

```
library(emmeans)
out <- lm(writhing~treatment, data=dd)
emmeans(out, spec='treatment')
```

```
 treatment    emmean       SE df lower.CL upper.CL
 alone      40.61995 4.341188 39 31.83907 49.40083
 injected.n 35.55367 4.964339 39 25.51235 45.59500
 injected.y 65.90806 5.167048 39 55.45672 76.35940

Confidence level used: 0.95
```

SE and confidence intervals are not the same as those you would calculate based on the data for each group separately, because they are based on the error (residual) mean square for the model (this is why `df` $= 39$ for each estimate here).

Note: `emmeans()` yields the predicted or marginal means according to the model. These predicted means are not necessarily the same as the individual group means when there are multiple predictors in the model (here, they are, as there is only one predictor).

Summary

- Linear models can fit categorical variables.
- Use `summary()` to obtain parameter estimates. To interpret the estimates, it is useful to know about how R handles categorical variables behind the scenes (dummy indicator variables).
- Ordering your categories well (e.g., control group first) will maximize the usefulness of the parameter estimates from the fitted model (e.g., estimates of differences between each treatment group and the control group).
- Use `drop1` for hypothesis testing (P-values, sums of squares).
- Use `emmeans()` to estimate predicted group means.

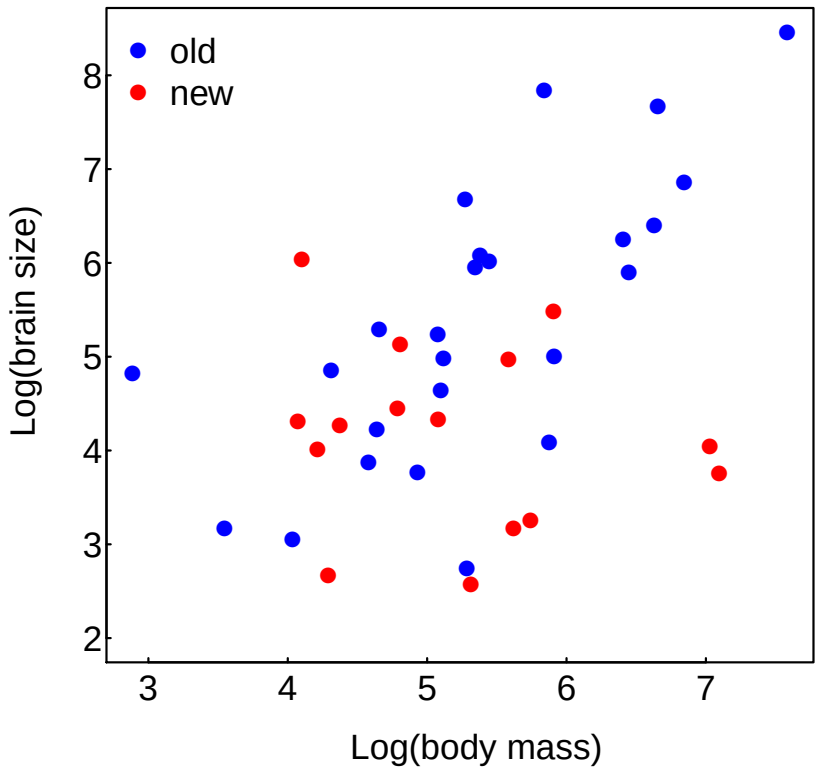
Unplanned ("post hoc") comparisons:

- Multiple comparisons among means after ANOVA done.
- Used to find which pairs of means are statistically significantly different.
- A kind of data dredging (i.e., no plan).
- Incorporates special protection against high false positive rate.
- Can't use *P*-values in `summary()` table.

Planned ("a priori") comparisons:

- Comparisons between group means that were decided when the experiment was designed (not after the data were in).
- For example, compare a key treatment against the control.
- Must be few in number to avoid inflating false positive rate.
- *P*-values in `summary()` can be used for planned comparisons (but careful with summary - for `glms` they are not reliable).

Simulated data: Is there a relationship between body mass and brain size, and does it differ between the new and old world?



```
out <- lm(log.brain.size~log.mass*world, data=dd)
summary(out)
```

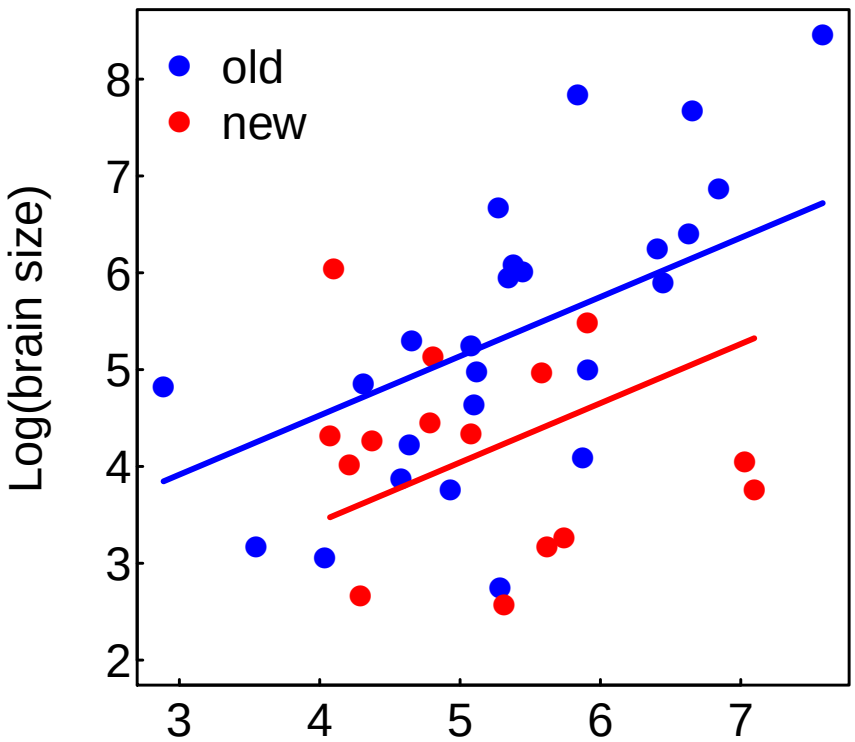
```
Call:
lm(formula = log.brain.size ~ log.mass * world, data = dd)

Residuals:
     Min       1Q   Median       3Q      Max
-2.54360 -0.58075 -0.03047  0.64371  2.00576

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)       5.0015     1.5436   3.240  0.00257 **
log.mass         -0.1610     0.2920  -0.551  0.58494
worldold         -4.9104     1.8983  -2.587  0.01388 *
log.mass:worldold  1.1443     0.3554   3.220  0.00272 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.068 on 36 degrees of freedom
Multiple R-squared:  0.4965,  Adjusted R-squared:  0.4546
F-statistic: 11.83 on 3 and 36 DF,  p-value: 0.00001526
```

```
drop1(out, test='F')
```

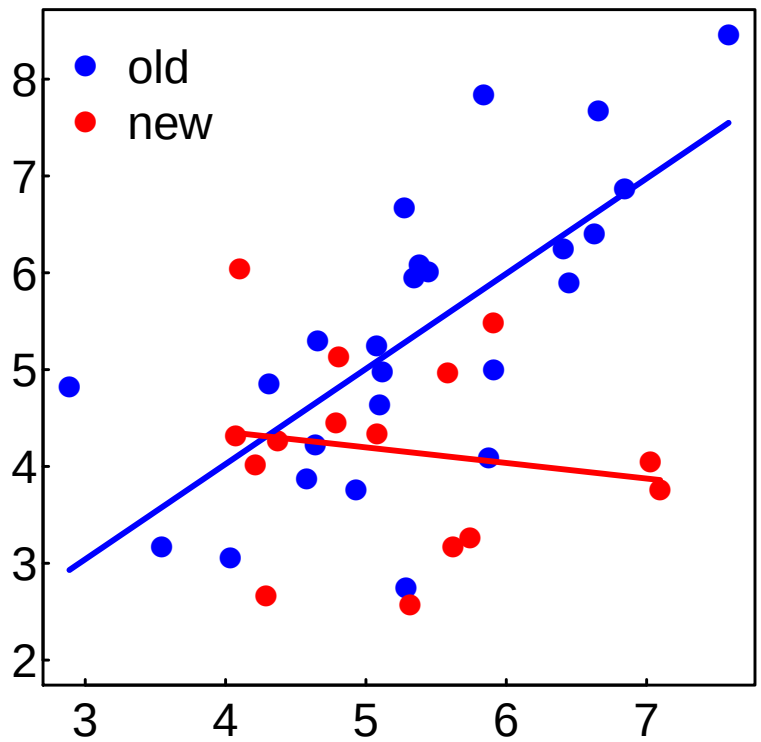
```
Single term deletions

Model:
log.brain.size ~ log.mass * world
             Df Sum of Sq    RSS     AIC F value   Pr(>F)
<none>                    41.062  9.0479
log.mass:world  1    11.825 52.887 17.1709  10.367 0.002718 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

So, we have evidence for a signficant interaction.

`log.brain.size~log.mass+world`



`log.brain.size~log.mass*world`



Log(body mass)

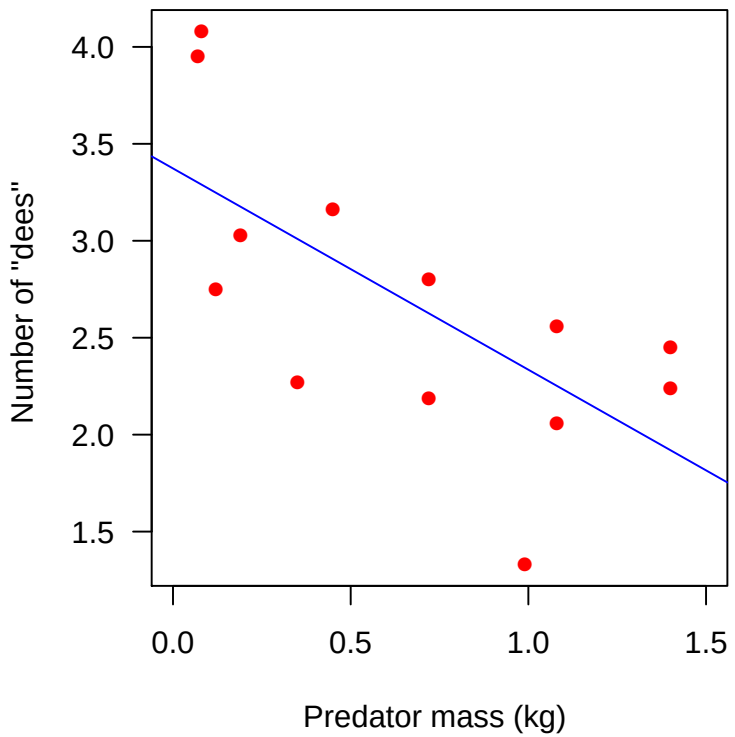
```
Coefficients:
             Estimate Std. Error
(Intercept)    0.9831     1.0169
log.mass       0.6117     0.1863
worldold       1.0970     0.3915
```

```
Coefficients:
                Estimate Std. Error
(Intercept)       5.0015     1.5436
log.mass         -0.1610     0.2920
worldold         -4.9104     1.8983
log.mass:worldold  1.1443     0.3554
```

R has lots of built in tools for plotting the output of linear models (although, I rarely use them).

Making figures by hand (at least, at first), by adding the best fit lines, is a great exercise!

```
plot(dd$mass, dd$dees)
aa <- coef(out)['(Intercept)']
bb <- coef(out)['mass']
abline(a=aa, b=bb)
```



Evaluating model fit is critical when running any linear model. Linear models assume:

- Normally-distributed errors
- Independent errors
- Equal variance of residuals in all groups

R has built-in diagnostics for `lm` objects (workshop this week).

What if there are random effects?
`lme`

What if response data are binary or discrete?
`glm`

What if residuals are not independent because of temporal autocorrelation or phylogeny?
`gls`