Linear models

Outline

- What is a linear model
- Linear regression example
- Single factor ANOVA example
- Coefficients table and ANOVA table
- Analysis of covariance example
- The lure of model simplification
- Perils of correcting for covariates
- Core assumptions of linear models
- Alternative approaches when assumptions are violated

What is a linear model

A relationship between variables involving

- a response variable Y
- explanatory variables X_1, X_2, …
- random normal errors with equal variance

in the form

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

where β_0, β_1, β_2, … are the *parameters* of the linear model.

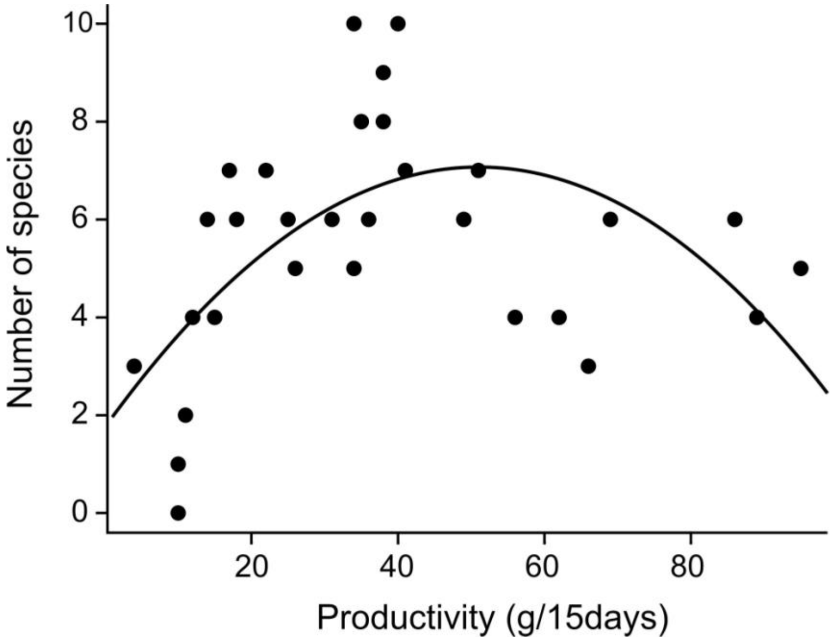
(Sometimes called "general linear model", not to be confused with "generalized linear model")

A linear model needn't be a straight line

For example, the quadratic equation

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

is also a linear model



Linear models includes methods known by other names:

- Linear regression
- Single factor ANOVA
- One-sample *t*-test
- Analysis of covariance
- Multiple regression
- Multi-factor ANOVA
- Repeated-measures ANOVA

All can be written in the form

(response variable) = intercept + (explanatory variables)

So what

Linear model approach unites these methods into a common framework that

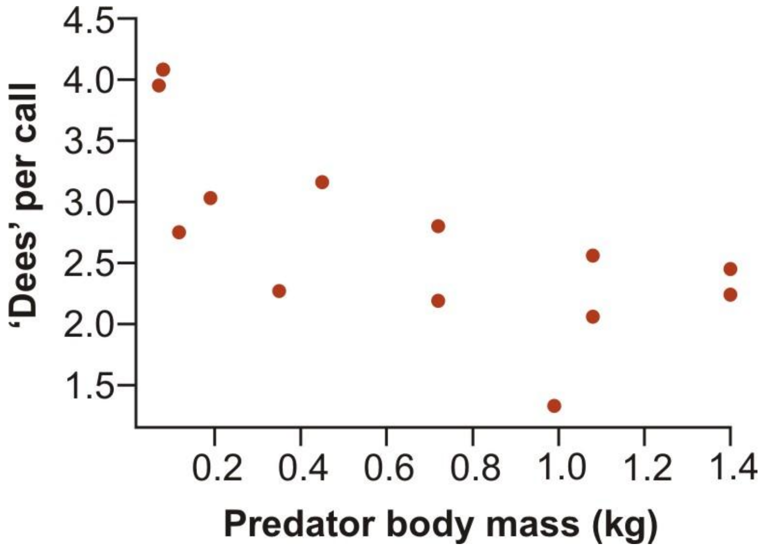
- Is flexible
- Provides a common set of tools (`lm` in R for fixed effects)
- Includes tools to estimate parameters (e.g., sizes of effects)
- Is easier to work with, especially with multiple variables
- Better handling of unbalanced designs than traditional ANOVA

Example 1: Simple linear regression

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.



http://animal.discovery.com/guides/wild-birds/a-c/black-capped-chickadee.html

Linear model

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

Meaning of parameters in this model:

- β_0 : intercept
- β_1 : slope

In words:

dees = intercept + mass

In R the intercept is implicit and doesn't need to be included in the word statement of the model formula:

```
dees ~ mass
```

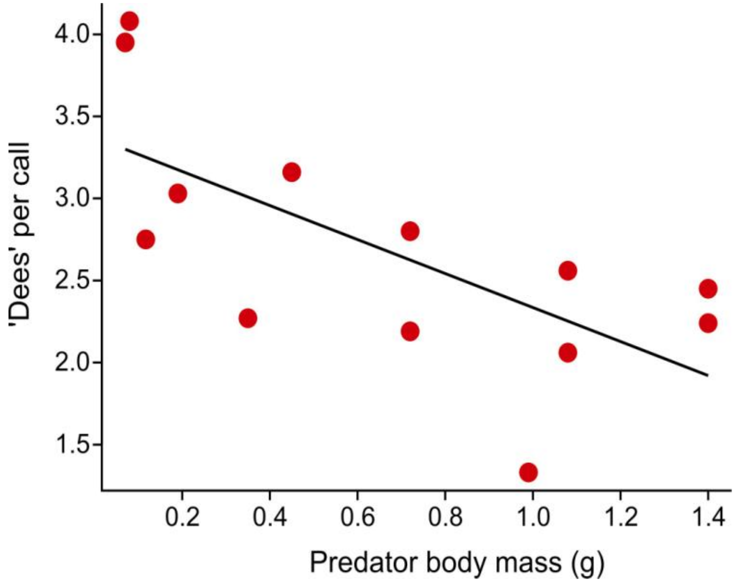
Coefficients table

The coefficients table contains parameter estimates with SE's

$$Y = b_0 + b_1X$$

```
z <- lm( dees ~ mass )
summary(z)   # yields coefficients table
```

	Estimate	Std. Error	*t* value	Pr(>|*t*|)
(Intercept)	3.3731	0.2776	12.149	1.02e-07 ***
mass	-1.0382	0.3402	-3.051	0.0110 *



[figure x axis should say kg not g]

ANOVA table

The ANOVA table contains *F*-ratios and *P*-values

```
anova(z)    # yields ANOVA table
```

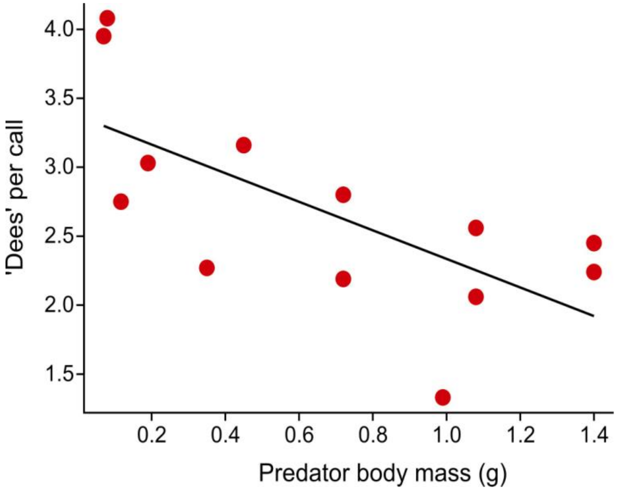
	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
mass	1	3.1268	3.1268	9.3106	0.01102 *
Residuals	11	3.6942	0.3358		

Test of null hypothesis that slope $\beta_1 = 0$

Behind the scenes

To estimate parameters, method uses least squares to find the best fit to the data of two predictors: mass, and a "dummy" column of 1's, such that there is a parameter for each column of numbers.

Observation	dees	(Intercept)	mass
1	**3.95**	**1**	**0.07**
2	**4.08**	**1**	**0.08**
3	**2.75**	**1**	**0.12**
4	**3.03**	**1**	**0.19**
5	**2.27**	**1**	**0.35**
6	**3.16**	**1**	**0.45**
7	**2.19**	**1**	**0.72**
8	**2.80**	**1**	**0.72**
9	**2.45**	**1**	**1.40**
10	**1.33**	**1**	**0.99**
11	**2.24**	**1**	**1.40**
12	**2.56**	**1**	**1.08**
13	**2.06**	**1**	**1.08**



So that dees[i] = b_0 (1) + b_1 mass[i] + residual[i], …

Behind the scenes

To test null hypothesis the method:

1) fits a “reduced” model without mass term (yielding fit under H_0)

2) fits the “full” model with mass term added back

3) compares fit of full and reduced models using an *F* test

You can optionally implement these steps “by hand” in R:

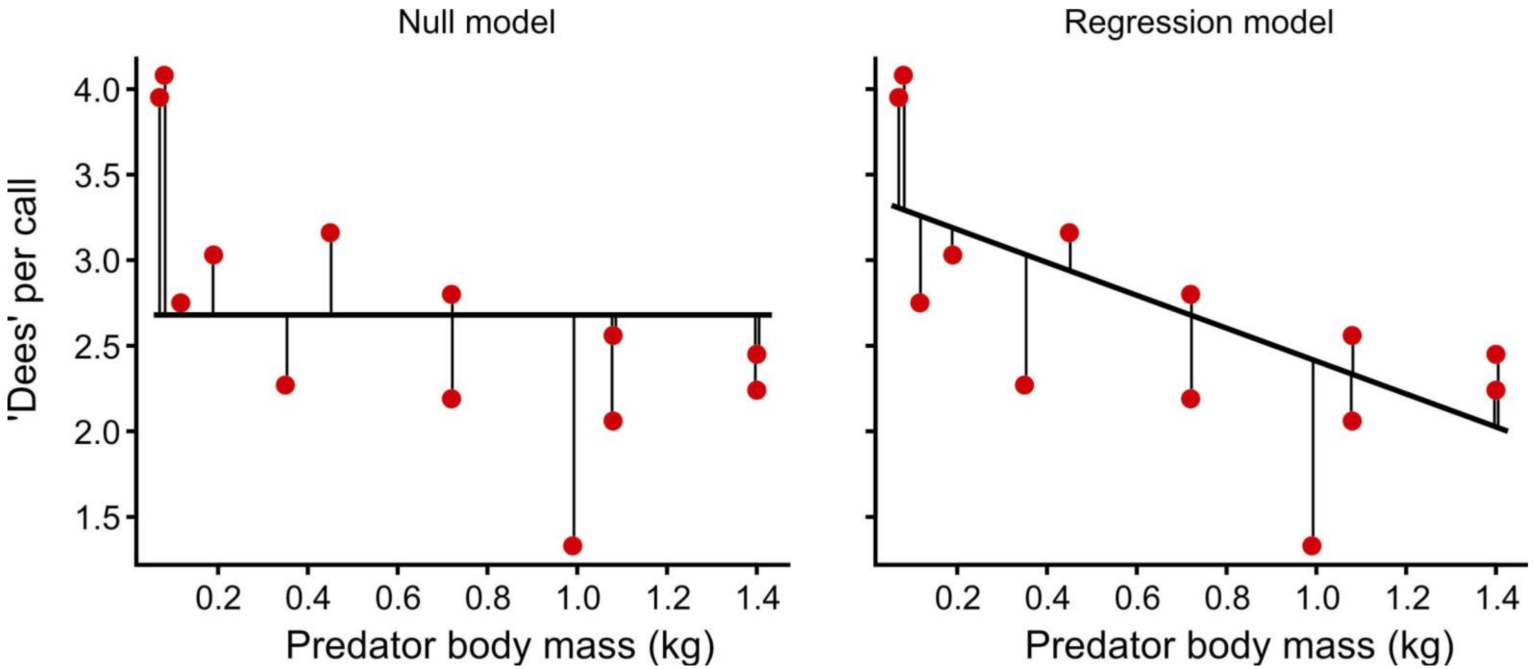
```
z1<-lm(dees ~ 1)    # fit reduced model (intercept only)
z <-lm(dees ~ mass) # fit full model
anova(z1,z)         # compare fits, yielding:
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1 [reduced]	12	6.8210				
2 [full]	11	3.6942	1	3.1268	9.3106	0.01102 *

This hands-on approach is a way for you to control exactly what R tests

Visually, this is the comparison

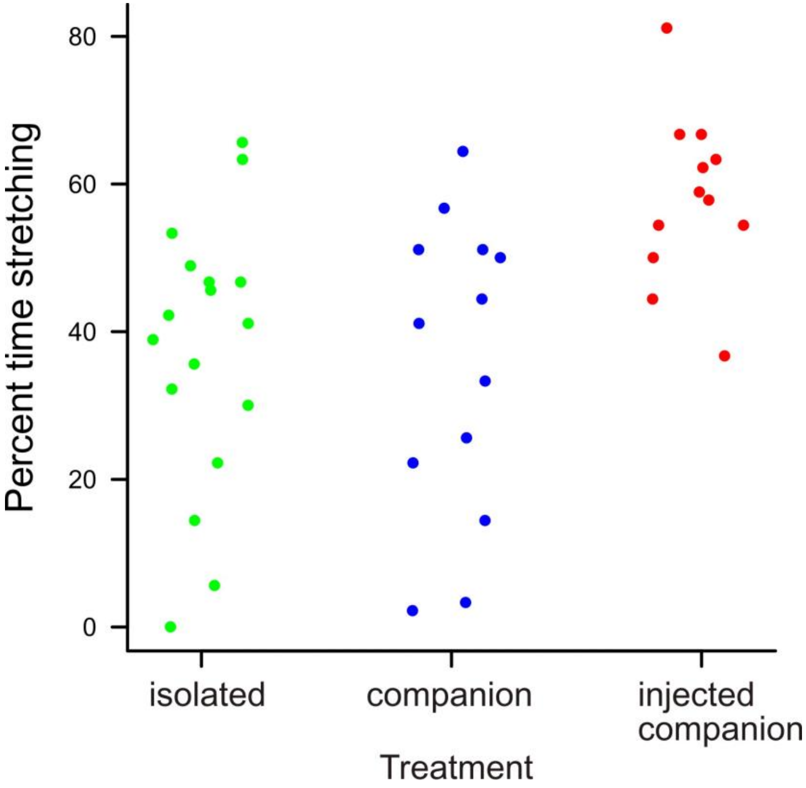
```
dees ~ 1                dees ~ mass
```

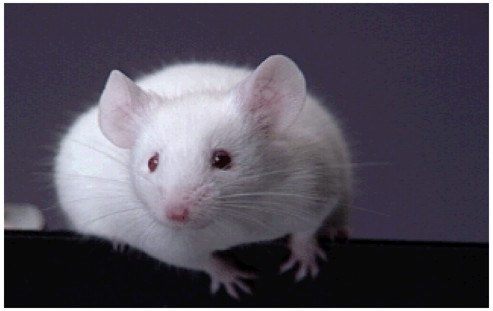


Example 2: Single-factor ANOVA

Percent time male mice experiencing discomfort spent “stretching”.

Data are from an experiment in which mice experiencing mild discomfort (result of injection of 0.9% acetic acid into the abdomen) were kept in: (1) isolation, (2) with a familiar companion mouse not injected, or (3) with a companion mouse also injected and exhibiting “stretching” behaviors associated with discomfort. The results suggest that mice stretch the most when a companion mouse is also experiencing mild discomfort. Mice experiencing pain appear to “empathize” with familiar mice also in pain.





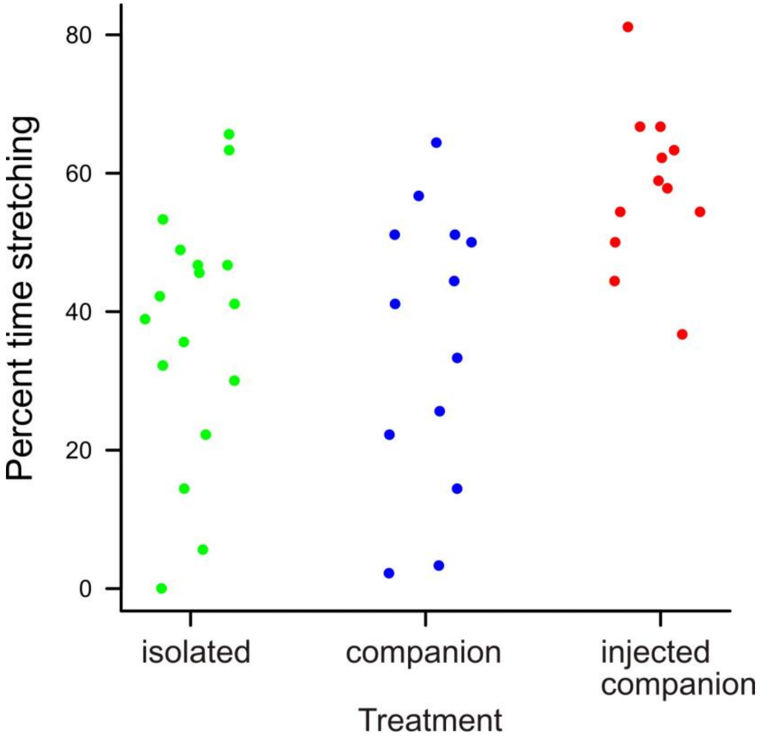
Langford, D. J.,et al. 2006. Science 312: 1967-1970

Single-factor ANOVA is a linear model, just like linear regression

In words:

stretching = intercept + treatment

The model statement includes a response variable, a constant, an explanatory variable. The only difference is that here the explanatory variable is categorical.



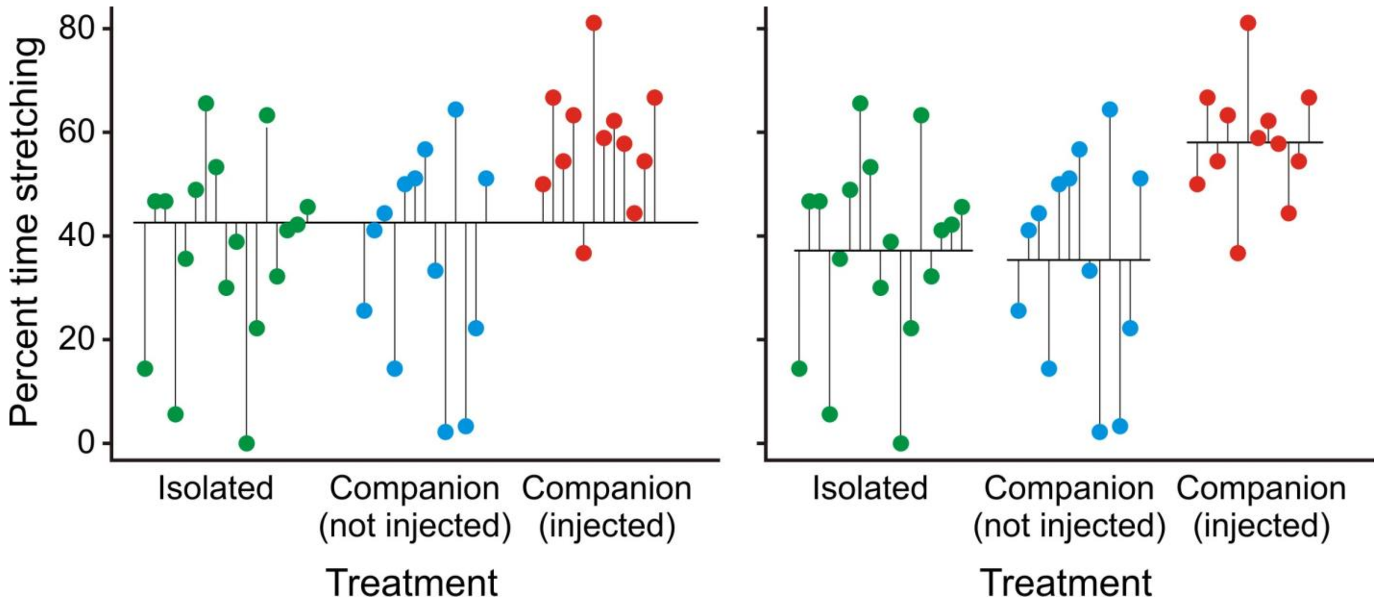
ANOVA Table

```
z <- lm( stretching ~ treatment )
anova(z)
```

	Df	Sum Sq	Mean Sq	*F* value	Pr(>*F*)
Treatment	2	4040.9	2020.5	6.6736	0.003216 **
Residuals	39	11807.4	302.8		

As before, `anova` compares the fit of "reduced" and "full" models:

`stretching ~ 1`   `stretching ~ treatment`



Coefficients table

```
z <- lm( stretching ~ treatment )
summary(z)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	37.194	4.220	8.814	8.06e-11***
treatcompanion	-1.825	6.411	-0.285	0.77741
treatcompan.inj	20.856	6.560	3.179	0.00289**

Q: what do these variables mean? (explanation below)

How to understand the coefficients table

Behind the scenes, R codes the 3 groups of the categorical variable using dummy indicator variables.). (R calls these "Treatment contrasts")

stretching	(Intercept)	treatisolation	treatcompanion	treatcompan.inj
64.4	1	1	0	0
46.7	1	1	0	0
38.9	1	1	0	0
65.6	1	1	0	0
…				
56.7	1	0	1	0
51.1	1	0	1	0
50.0	1	0	1	0
51.1	1	0	1	0
…				
36.7	1	0	0	1
81.1	1	0	0	1
66.7	1	0	0	1
66.7	1	0	0	1

By default R leaves out the dummy representing the first factor level (determined alphabetically if order is unspecified by the user)

How to understand the coefficients table

stretching	(Intercept)	treatmentcompanion	treatmentcompan.inj
64.4	1	0	0
46.7	1	0	0
38.9	1	0	0
65.6	1	0	0
…			
56.7	1	1	0
51.1	1	1	0
50.0	1	1	0
51.1	1	1	0
…			
36.7	1	0	1
81.1	1	0	1
66.7	1	0	1
66.7	1	0	1

Write out the model to interpret the coefficients:

$\text{stretching} = \beta_0 * 1 + \beta_1 * 0 + \beta_2 * 0$ (subjects in isolation group)

$\text{stretching} = \beta_0 * 1 + \beta_1 * 1 + \beta_2 * 0$ (subjects in companion group)

$\text{stretching} = \beta_0 * 1 + \beta_1 * 0 + \beta_2 * 1$ (subjects in compan.inj group)

How to understand the coefficients table

So the linear model being fitted is:

stretching = β_0 (subjects in isolation group)
stretching = $\beta_0 + \beta_1$ (subjects in companion group)
stretching = $\beta_0 + \beta_2$ (subjects in compan.inj group)

β_0 estimates the mean of the isolated (control) group

β_1 estimates the difference between the companion and control

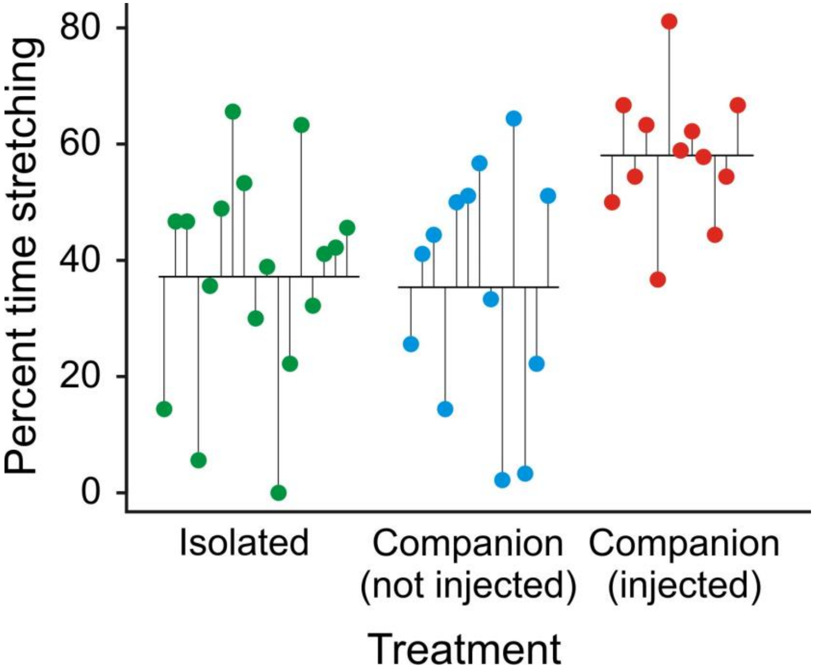
β_0 estimates the difference between the company.inj and control

How to understand the coefficients table

```
z <- lm( stretching ~ treatment )

summary(z)
```

	Estimate	Std. Error	*t* value	Pr(>|*t*|)
(Intercept)	37.194	4.220	8.814	~~8.06e-11***~~
treatcompanion	-1.825	6.411	-0.285	~~0.77741~~
treatcompan.inj	20.856	6.560	3.179	~~0.00289**~~



↑

These *P*-values are invalid except for planned comparisons

How to understand the coefficients table

The estimates and SE’s are the most useful quantities in this table.
The *P*-values in coefficients table are invalid for unplanned comparisons

Planned comparisons:

- Comparisons between group means that were decided when the experiment was designed (not after the data were in)
- Must be few in number to avoid inflating Type 1 error rates

Unplanned comparisons:

- Multiple comparisons carried out after the results are in
- Used to find where the differences lie (which means differ from which other means)
- Basically amounts to snooping, or data dredging
- Comparisons require protection for inflated Type 1 error rates:
    - Tukey tests: compare all pairs of means
    - Scheffé contrasts: compare all combinations of means

The flexibility of R is that you can choose alternative coding

```
z <- lm( stretching ~ treatment - 1 )
```

"Means model" in R

The `-1` tells R to take the intercept dummy out of the model instead

Behind the scenes, this is what R does now:

stretching	treatmentisolated	treatmentcompanion	treatmentcompan.inj
64.4	1	0	0
46.7	1	0	0
38.9	1	0	0
65.6	1	0	0
…			
56.7	0	1	0
51.1	0	1	0
50.0	0	1	0
51.1	0	1	0
…			
36.7	0	0	1
81.1	0	0	1
66.7	0	0	1
66.7	0	0	1

Still 3 columns, but the intercept column is replaced by the dummy corresponding to the first factor level.

Interpretation of parameters under the "means model"

Different parameters are being estimated because of the different coding:

Treatment	
isolated	stretching = β_0
companion	stretching = β_1
compan.inj	stretching = β_2

β_0 estimates the mean of the isolated (control) group

β_1 estimates the mean of the companion group

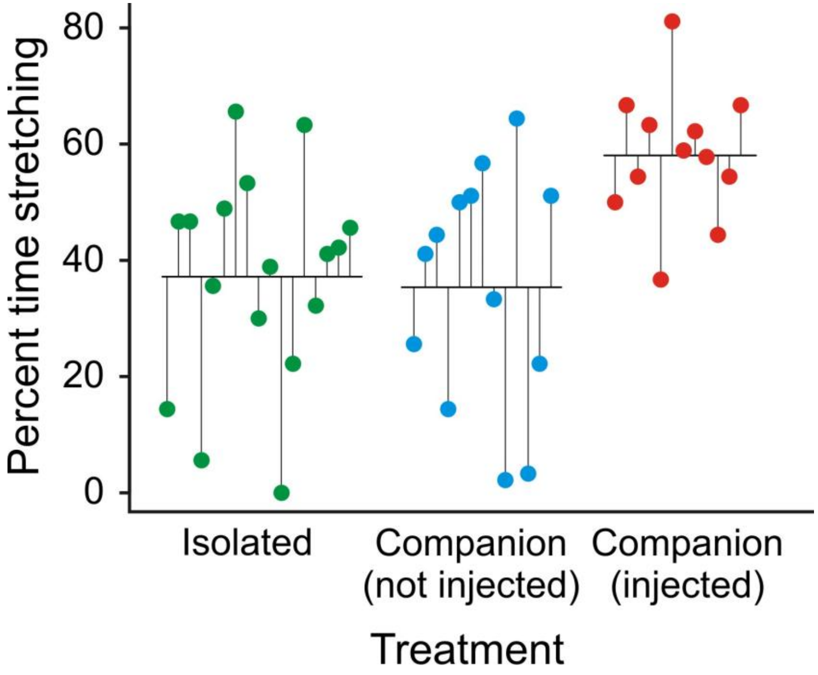
β_2 estimates the mean of the compan.inj group

Parameter estimates under the “means model”

```
z <- lm( stretching ~ treatment - 1 )
summary(z)
```

	Estimate	Std. Error	t value	Pr(>|t|)
treatisolation	37.194	4.220	8.814	~~8.06e-11***~~
treatcompanion	35.369	4.826	7.329	~~7.06e-09***~~
treatcompan.inj	58.050	5.023	11.557	~~3.64e-14***~~

Standard errors here use the $MS_{residual}$



Useless

(tests of null hypotheses that means are zero)

R^2 also useless

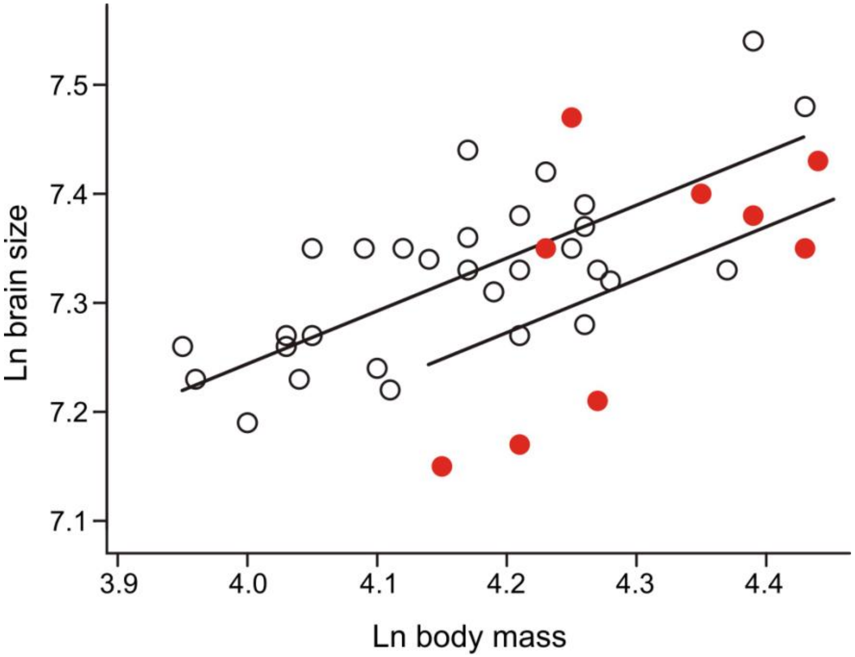
Summary so far

- Linear models can handle numeric and categorical variables
- You don’t need to know too much about how the method handles categorical variables behind the scenes (i.e., as indicator variables).
- But organizing your categories (e.g., control group ordered first) or altering the formula slightly (e.g., to use “means model”) will enable you to maximize the usefulness of the parameter estimates from the fitted model.
- The flexibility of linear models will allow you to extract the most information possible from parameter estimates.

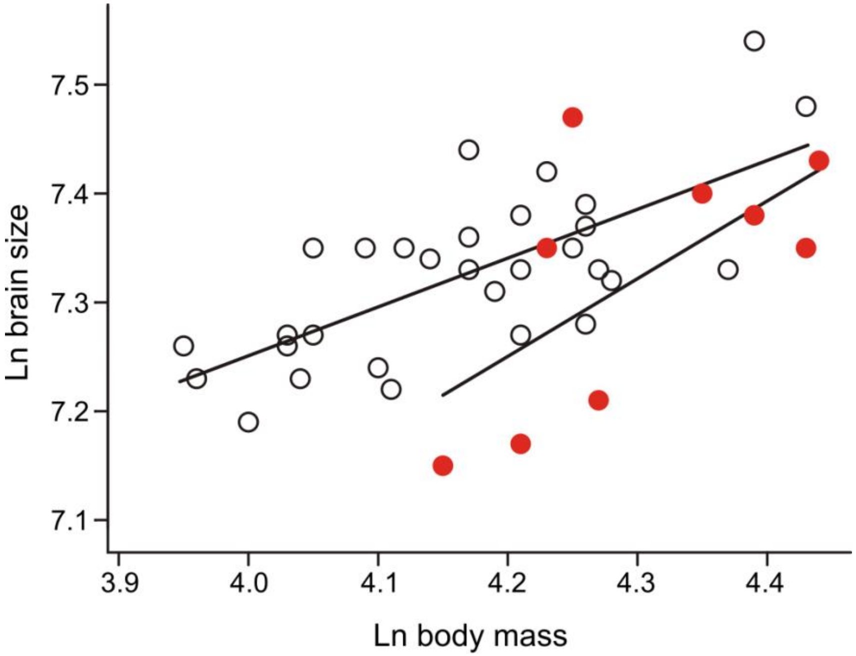
Example 3: Analysis of covariance

Data: Brain and body sizes of Neanderthal specimens (●) and early modern humans (○). Does our species have different brain sizes, after accounting for differences in body size? (Ruff et al 1977). This is easiest if we could use the model on the left.

```
brain ~ mass + species
```



```
brain ~ mass + species
              + mass*species
```



Fitting models with more than one explanatory variable

This includes analysis of covariance, multiple regression, multi-factor ANOVA, etc.

In R, `anova` fits model terms sequentially (“Type 1 SS”)

1. Model terms are tested in the sequence in which the user enters them in the `lm` formula statement.

2. Exception: `anova` respects hierarchy: “lower” terms (e.g., intercept) are fitted before “higher” terms (e.g. slope). For example, interaction terms are always fitted after the corresponding main effects are included in the model.

This is different from what SAS and other programs do, which instead use a “drop-1” approach (“Type III Sums of Squares”).

R doesn’t use drop-1 fitting of terms

Most other programs use a drop-1 approach by default, in which each term in the model is tested by comparing fit of the full model to a reduced model with the term of interest dropped but retaining all other terms. The order in which the terms are entered in the model formula has no effect. In particular, there is no respect for hierarchy. For example, interaction terms are left in the model when testing a main effect. Many statisticians feel strongly that this makes no sense.

Note: The distinction doesn’t matter when the study design is completely balanced.

Sequential vs drop-1 fitting

The *P*-values in the coefficients table correspond to the drop-1 method, whereas the *P*-values in the ANOVA table are based on sequential fitting of terms. As a consequence, the *P*-values in the ANOVA and coefficients tables will not agree when study designs are not balanced.

Which method is best?

See http://afni.nimh.nih.gov/sscc/gangc/SS.html for further discussion.

In my view:

Use the coefficients table for parameter estimation (`summary`).

Use the ANOVA table for hypothesis testing (`anova`).

Use the optional “hands-on” approach to force `anova` to do something different from its default.

Analysis of covariance

- Most ANCOVA designs are unbalanced, because specific *x*-values are often unique to a group.

- Because `anova` fits terms sequentially, the ANOVA table in R depends on the order in which variables are entered into the model.

  ```
  brain ~ mass + species + mass:species
  ```

  will yield a different ANOVA table than

  ```
  brain ~ species + mass + mass:species
  ```

- Because `anova` respects hierarchy, the interaction term `mass:species` (a "higher" term) is tested only after `mass` and `species` ("lower" terms) are included.

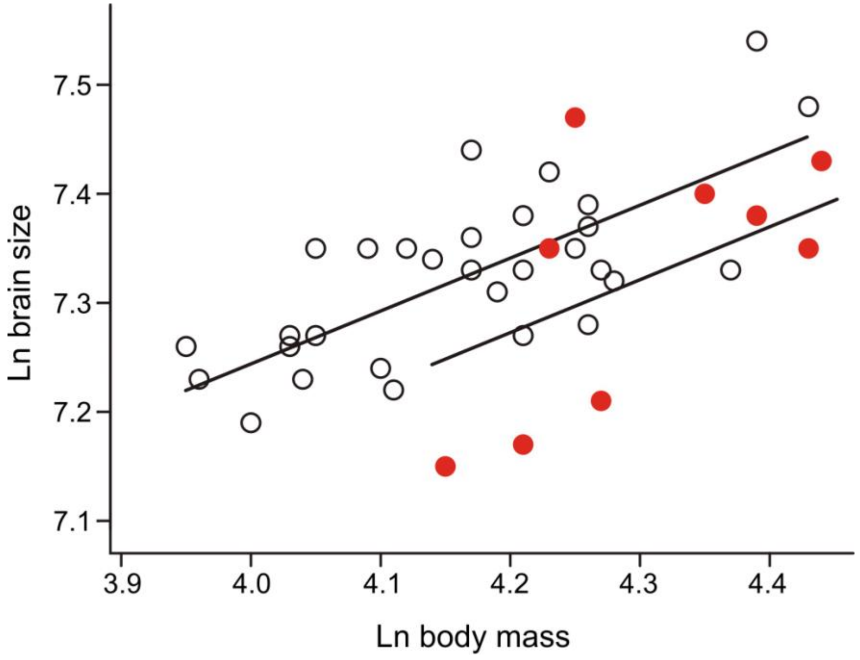
- You can force specific comparisons in R. For example, use `anova(z1,z2)` to compare any two user-specified models of interest, `z1` (reduced) and `z2` (full).

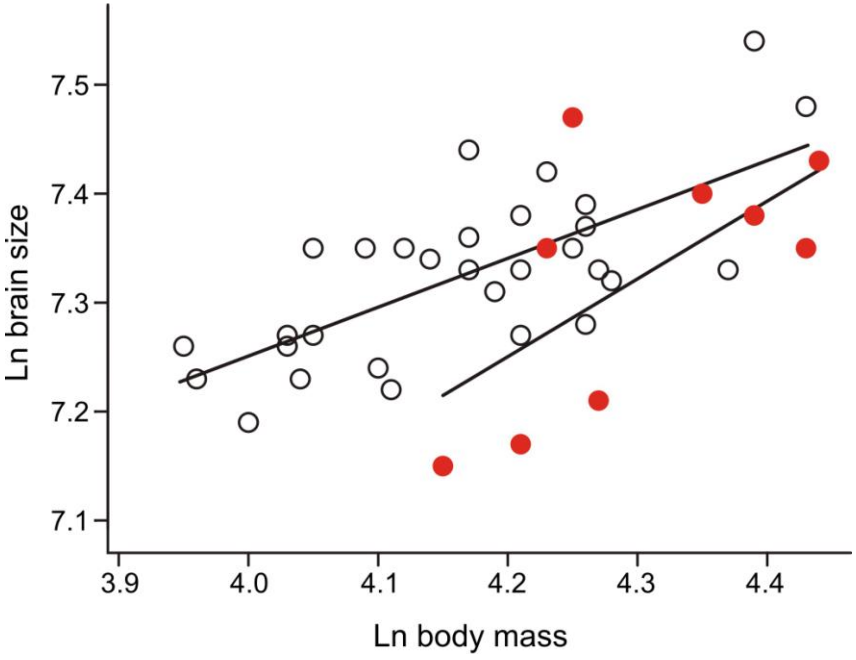
Back to analysis of covariance example

Does our species have different brain sizes after accounting for differences in body size? This is easiest if we could use the model on the left, but to do this involves model simplification, the leaving out of terms (usually those that are not statistically significant). Is this allowed?

```
brain ~ mass + species
```

```
brain ~ mass + species
        + mass*species
```





The perils of model simplification

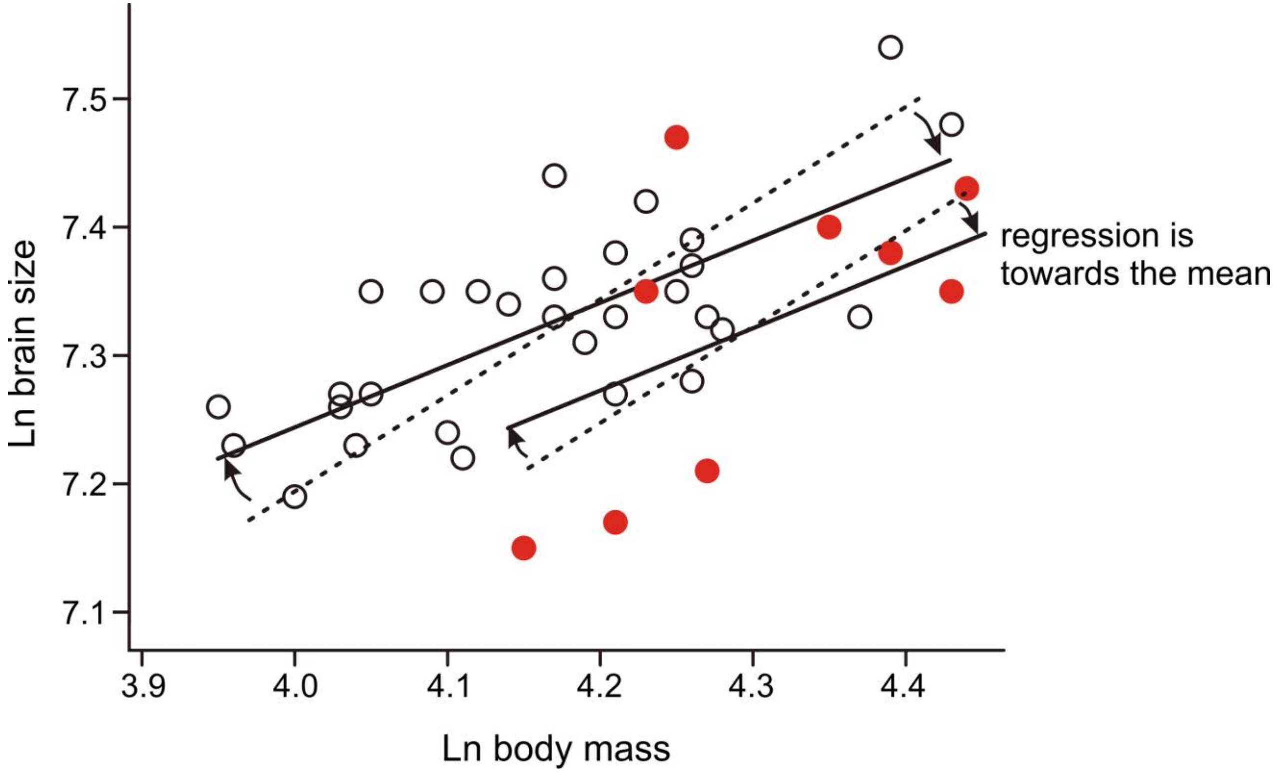
“models should be pared down until they are minimal adequate”

-- Crawley 2007, The R book, p325

- If the interaction term is not statistically significant, it is often dropped from the model, which is then refit without it.
- However, the *P*-value may be a poor guide to whether a model should include a variable or not. Other criteria may be better.
- Dropping a term purely because $P > 0.05$ involves “accepting” a null hypothesis, which is risky. Too many accepted null hypotheses can lead to the wrong model.
- At the same time, a simpler model with fewer terms may be justified to achieve particular goals. Be aware of the dangers, and be explicit about how you got to the model you decided to use.

Perils of correcting for covariates: regression towards the mean

Beware: Body mass is not “true size”, nor is it the cause of brain size. Mass is affected by sources of variation that don’t affect brain size (e.g., exercise, diet). As a result, regression on body mass will tend to “under-correct” for size.



Core assumptions of linear models

- Independent errors (residuals)
- Equal variance of residuals in all groups
- Normally-distributed residuals
- Robustness to departures from these assumptions is improved when sample size is large and design is balanced
- R has built-in diagnostics for `lm` objects using `plot` method (workshop)

Handling violations of assumptions within a linear model framework

What if your residuals are not independent because of serial autocorrelations or phylogeny?

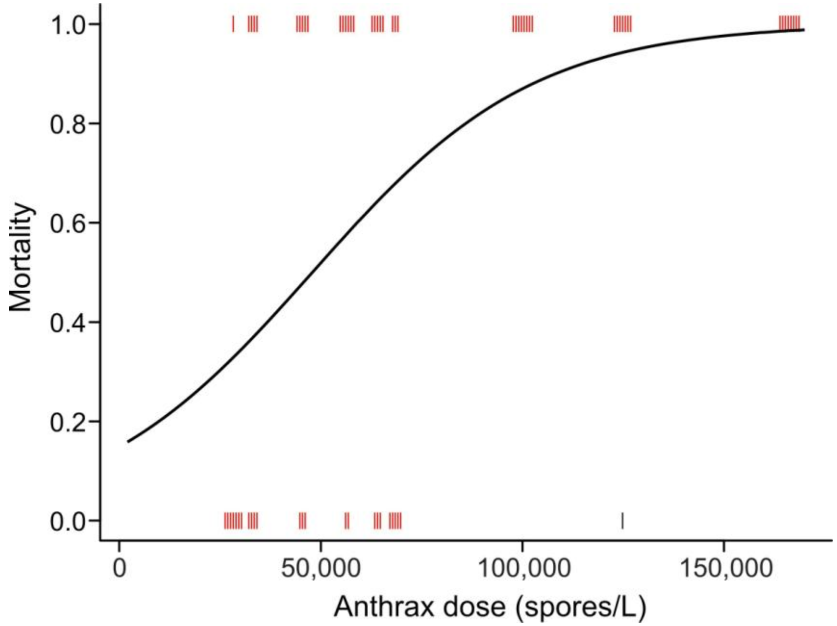
- General least squares (`gls` in R's `nlme` library)

What if your residuals aren't normal because of outliers? Nonparametric methods exist, but these don't provide parameter estimates.

- Robust regression (`rlm`)

What if your response variable is not normal but is some other distribution instead?

- Generalized linear models (`glm`)



What if the relationship between variables is has linear and nonlinear parts?

- Generalized additive models (`gam`)