Outline

- What is a linear model
- Several examples
- Estimating parameters vs testing hypotheses
- Model comparison: *full* vs *reduced* models
- Sequential vs marginal testing of terms
- The lure of model simplification
- Perils of correcting for covariates
- Assumptions of linear models
- Related methods in R

What is a linear model

A relationship between variables involving

- a response variable Y
- explanatory variables X_1, X_2, …
- normal random 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

What is a linear model

For example:

fit a mean to data: $Y = \beta_0$

simple linear regression: $Y = \beta_0 + \beta_1 X$

multiple regression: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots$

quadratic regression: $Y = \beta_0 + \beta_1 X + \beta_2 X^2$

single-factor ANOVA: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots$ (I will explain)

More names for types of linear models:

- Fitting different means to two groups
- Multi-factor ANOVA
- Analysis of covariance

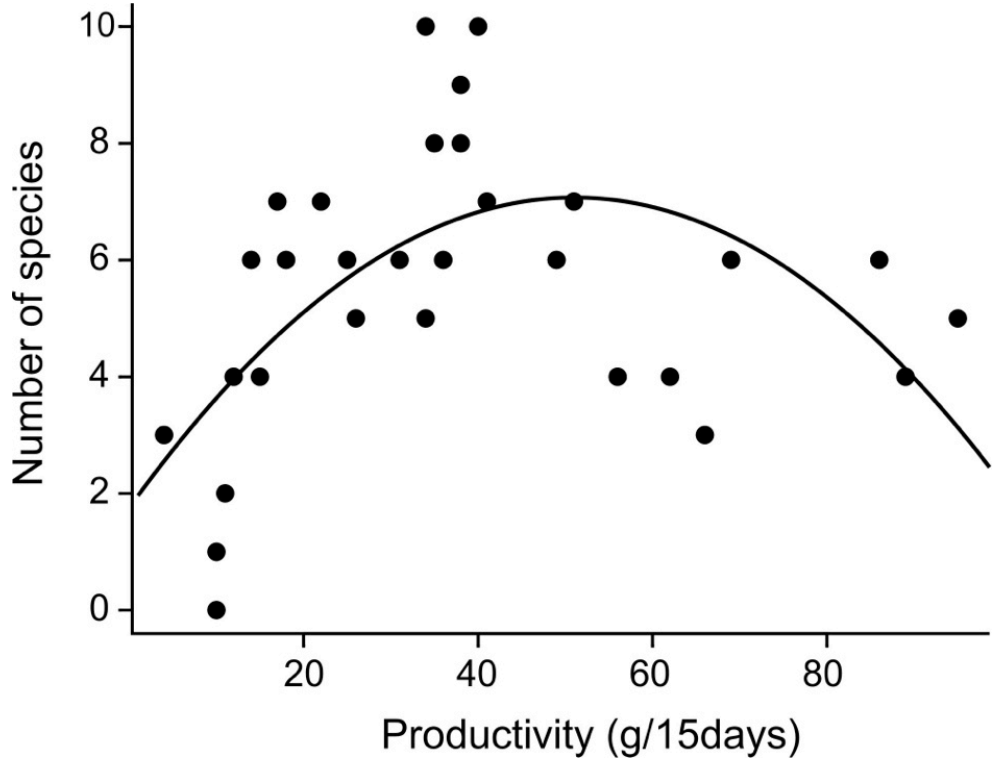
All can be written in the same form

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

A linear model needn't be a straight line

For example, the quadratic equation is a linear model

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



So what

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

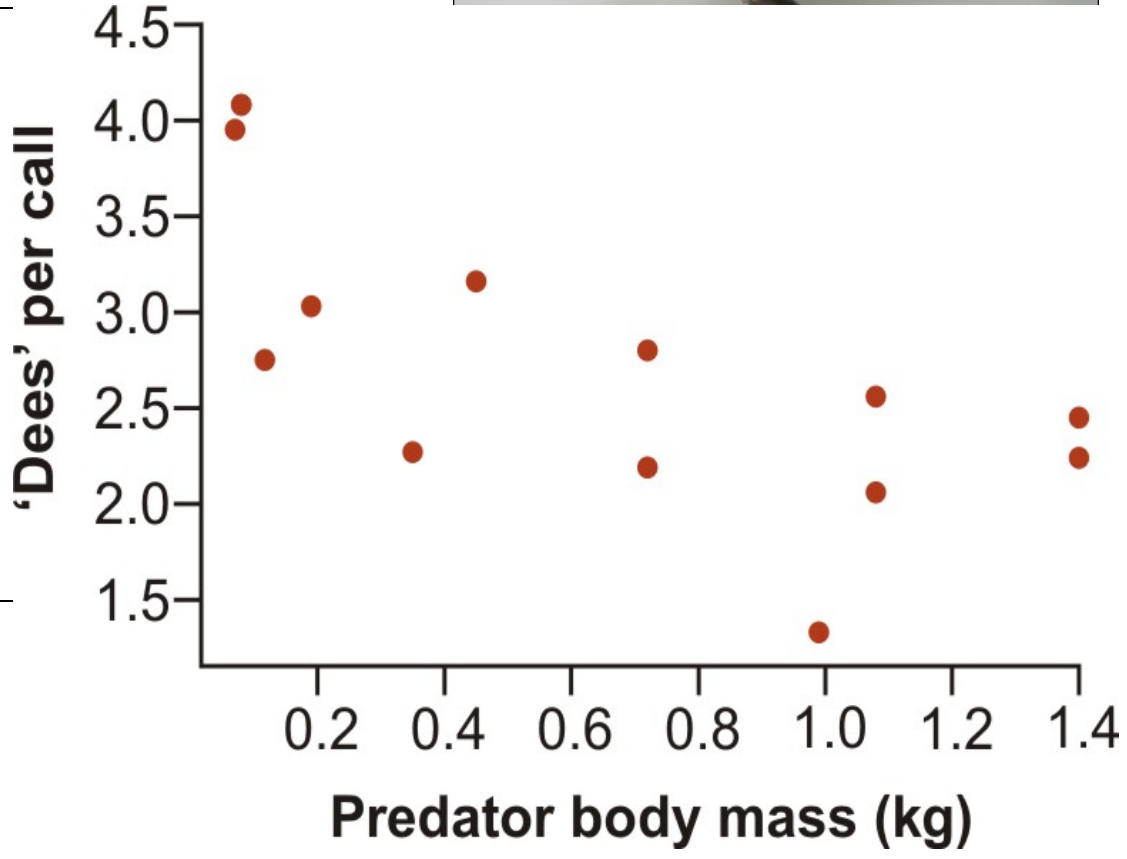
- Provides a common set of tools (`lm` in R for fixed effects)
- Is flexible and handles different study designs
- Has tools to estimate parameters (e.g., sizes of effects – biological significance)
- Is easy to use, even when there are multiple variables
- Better handling of unbalanced designs than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

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.



Linear model for simple linear regression

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

Parameters in this equation – these are the "effects":

- β_0 : intercept, β_1 : slope
- b_0 : estimate of intercept, b_1 : estimate of slope

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

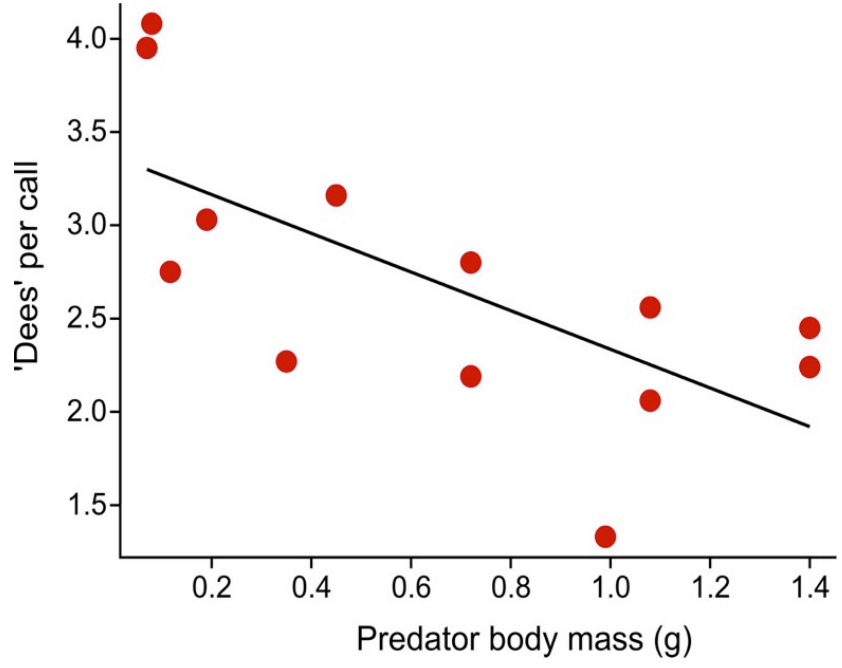
```
z <- lm(dees ~ mass)
```

Use `summary()` to get <u>parameter estimates</u> (ignore the tests)

Formula for the least squares estimate: $Y = b_0 + b_1X$

`summary(z)` # produces the coefficients table (ignore the tests)

	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 *



`summary()` What R does behind the scenes to estimate parameters

R fits two "variables" to the data: mass and a column of 1's,.

dees		dummy		mass	
3.95		1		0.07	
4.08		1		0.08	
2.75		1		0.12	
3.03		1		0.19	
2.27		1		0.35	
3.16	$= b_0$	1	$+ b_1$	0.45	+ residuals
2.19		1		0.72	
2.80		1		0.72	
2.45		1		1.40	
1.33		1		0.99	
2.24		1		1.40	
2.56		1		1.08	
2.06		1		1.08	

See that for each point *i*, dees[*i*] = b_0 (1) + b_1 mass[*i*] + residual[*i*]

e.g.: 3.95 = b_0 (1) + b_1(1.07) + residual[1st]

`summary()` What R does behind the scenes to estimate parameters

R uses least squares to fit a multiple regression to the X-variables ("dummy" and mass). The best estimates of b_0 and b_1 minimize the sum of squared residuals.

dees		dummy		mass	
3.95		1		0.07	
4.08		1		0.08	
2.75		1		0.12	
3.03		1		0.19	
2.27		1		0.35	
3.16	= b_0	1	+ b_1	0.45	+ residuals
2.19		1		0.72	
2.80		1		0.72	
2.45		1		1.40	
1.33		1		0.99	
2.24		1		1.40	
2.56		1		1.08	
2.06		1		1.08	

You can see the behind-the-scenes coding system in R as follows.

```
z <- lm(dees ~ mass)
model.matrix(z)
```

Use `summary()` to get <u>parameter estimates</u>

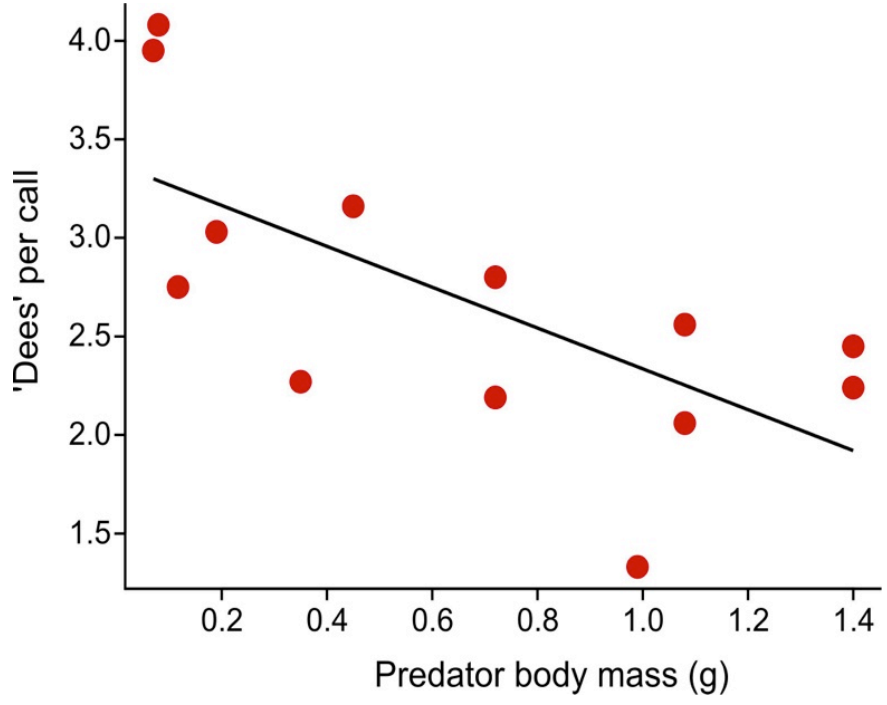
```
z <- lm(dees ~ mass)
summary(z)
```

yields the coefficients table with estimates b_0 and b_1 (Ignore the tests):

	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 *

```
visreg(z, "mass")
```

Produces a plot of the fitted model.

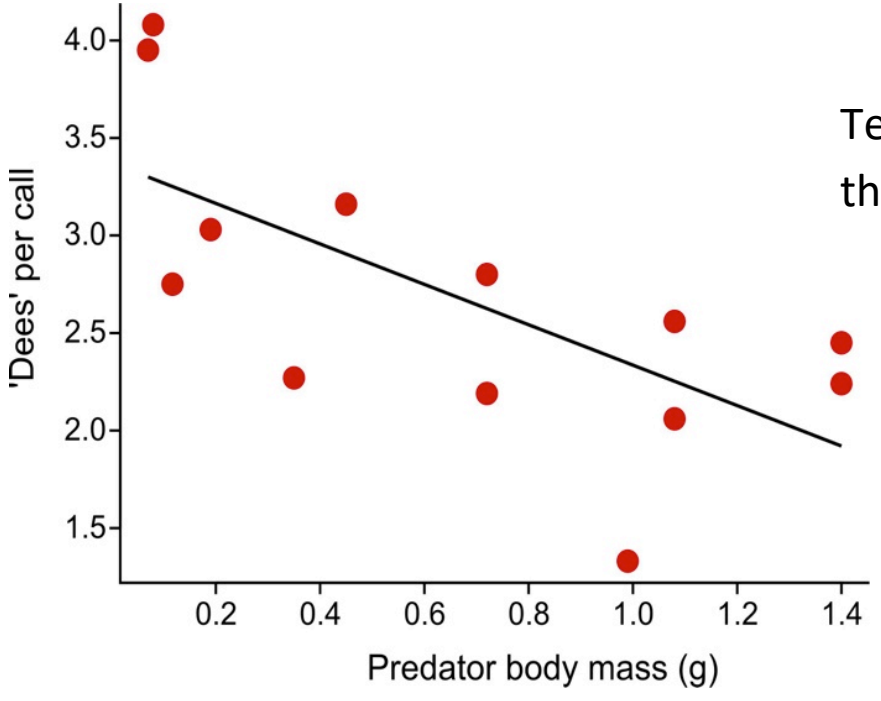


Use `anova()` or `Anova()` to <u>test hypothesis</u>

```
z <- lm(dees ~ mass)
anova(z)
```

yields the 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 $\beta_1 = 0$

Factors are tested using model comparison

`anova()` tests each term or factor by comparing fits of two models to the data. Comparison is always between a *reduced* model and a *full* model. The *reduced* model contains a subset of terms contained in the *full* model. The *F*-test is used to determine whether the *full* model fits the data significantly better than the *reduced* model.

Behind the scenes, this is how R tests the effect of predator body mass:

```
z0 <- lm(dees ~ 1)      # fits reduced model (intercept only)
z1 <- lm(dees ~ mass)   # fits full model with intercept and mass
anova(z0,z1)            # compares fits with F test, 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

Visually, `anova(z0,z1)` makes the following comparison:

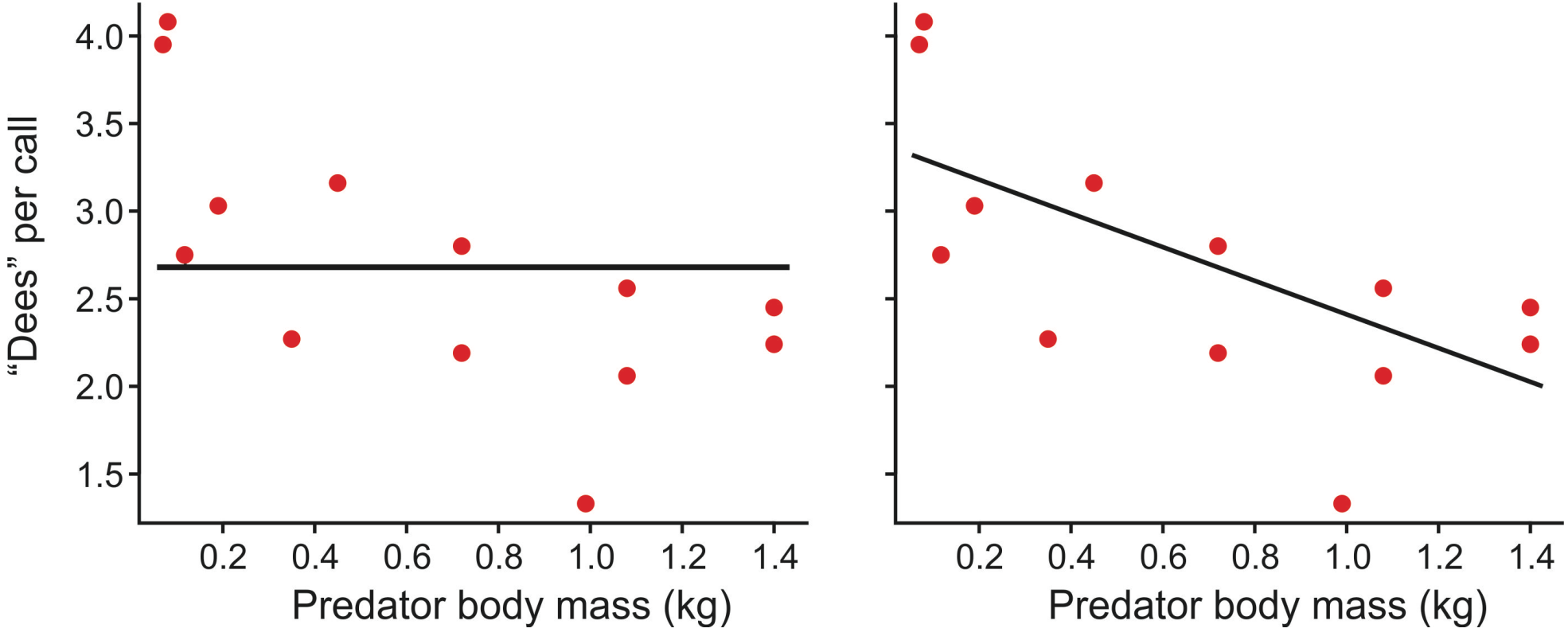
The test of predator body mass involves a comparison of these two models:

`dees ~ 1`

reduced model (fits only an intercept)

`dees ~ mass`

full model (intercept and slope)



Example 2: Multiple regression

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

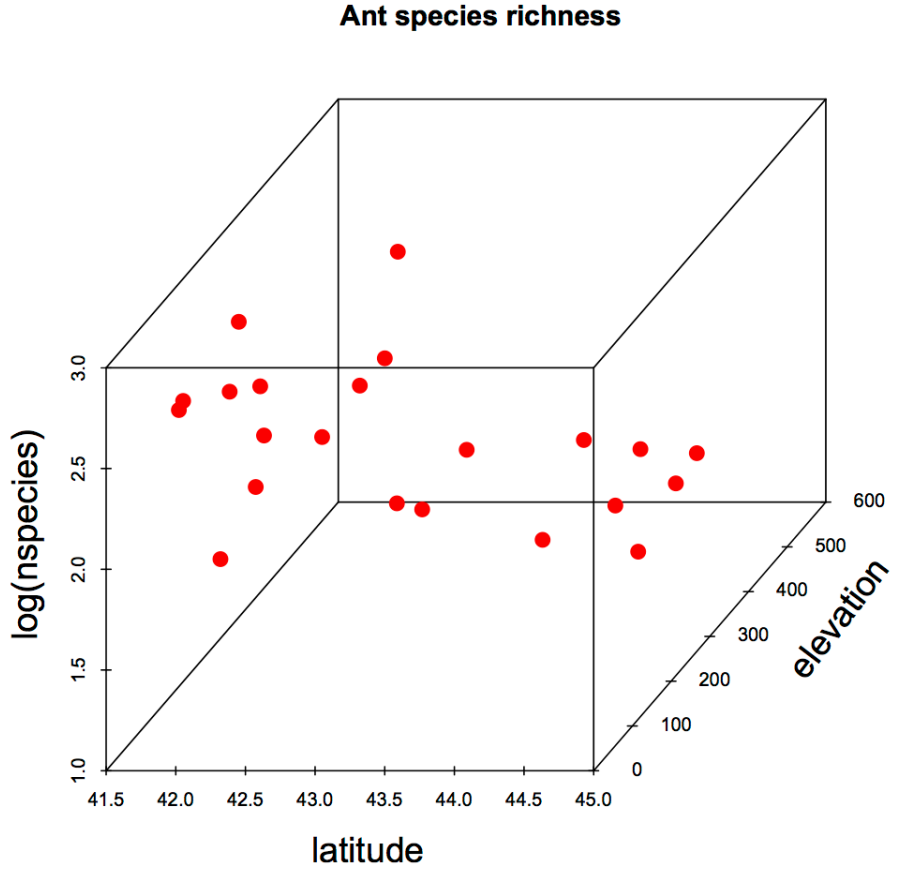
Gotelli, N.J. & Ellison, A.M. (2002b). Biogeography at a regional scale: determinants of ant species density in bogs and forests of New England. Ecology, 83, 1604–1609.

$$\log(\text{nspecies}) = \beta_0 + \beta_1(\text{latitude}) + \beta_2(\text{elevation}) + \beta_3(\text{latitude} \times \text{elevation})$$

Parameters in this model

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

(NB: sample size too small to fit so many parameters but for this example let's keep going anyway)



Example 2: Multiple regression

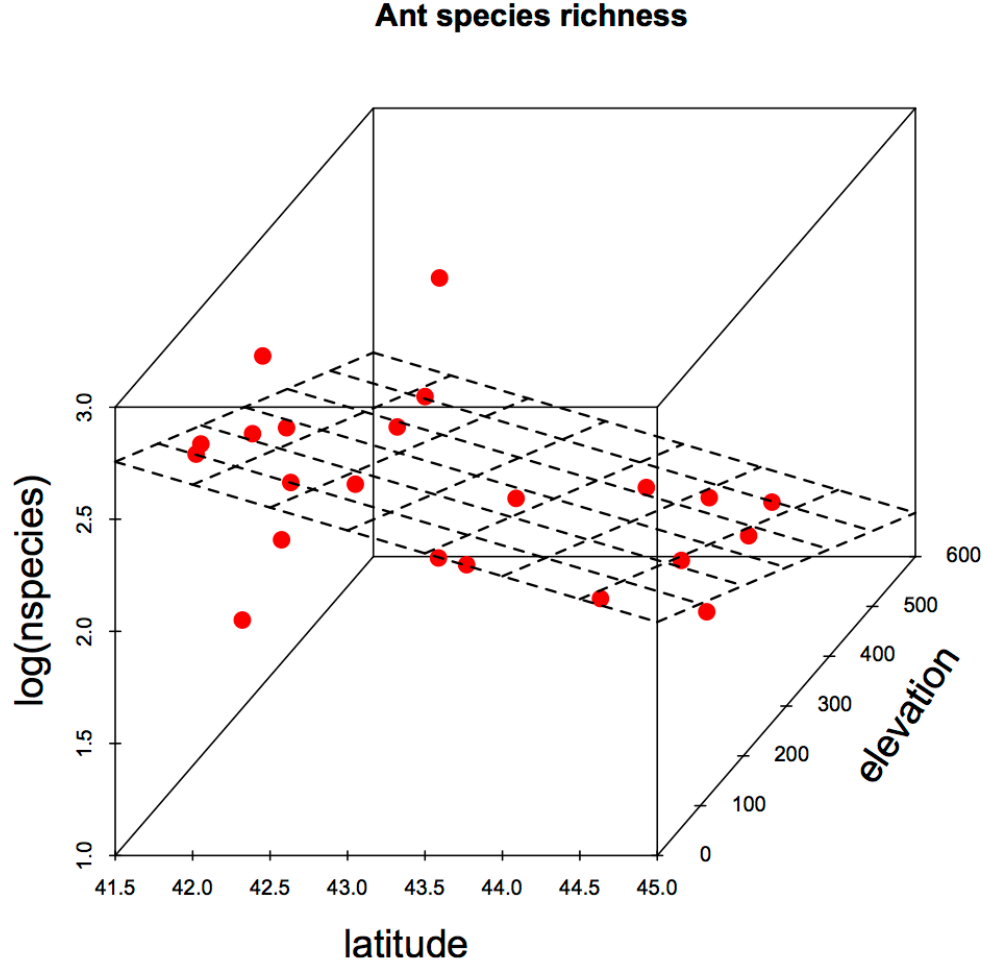
log(nsp)		dummy		latitude		elevation		lat*elev	
1.8		1		41.97		389		16326.33	
2.8		1		42.00		8		336.00	
2.9		1		42.03		152		6388.56	
2.8		1		42.05		1		42.05	
2.2		1		42.05		210		8830.50	
2.7		1		42.17		78		3289.26	
1.9		1		42.19		47		1982.93	
2.5		1		42.23		491		20734.93	
2.6		1		42.27		121		5114.67	
2.2	= b_0	1	+ b_1	42.31	+ b_2	95	+ b_3	4019.45	+ residuals
2.3		1		42.56		274		11661.44	
2.3		1		42.57		335		14260.95	
1.4		1		42.58		543		23120.94	
1.6		1		42.69		323		13788.87	
1.9		1		43.33		158		6846.14	
1.9		1		44.06		313		13790.78	
1.4		1		44.29		468		20727.72	
1.8		1		44.33		362		16047.46	
1.8		1		44.50		236		10502.00	
2.1		1		44.55		30		1336.50	
1.8		1		44.76		353		15800.28	
1.8		1		44.95		133		5978.35	

Use `summary()` to get <u>parameter estimates</u>

```
z <- lm(log(nspecies) ~ latitude * elevation)
summary(z)
```

yields the estimates b_0, b_1, b_2, b_3 (Ignore the tests):

	Estimate	Std.Error	*t* value	Pr
(Intercept)	12.6271	5.0457	2.503	(
latitude	-0.2369	0.1181	-2.006	(
elevation	-0.0076	0.0187	-0.406	(
latitude:elevation	0.0001	0.0004	0.331	(



Use `anova()` **or** `Anova()` **to <u>test hypothesis</u>**

```
z <- lm(log(nspecies) ~ latitude * elevation)
anova(z)
```

yields the ANOVA table

	Df	**Sum Sq**	**Mean Sq**	***F***	**Pr(>*F*)**	
latitude	1	1.44425	1.44425	14.5030	0.0013	**
elevation	1	1.07581	1.07581	10.8032	0.0041	**
latitude:elevation	1	0.01091	0.01091	0.1096	0.7444	
Residuals	18	1.79249	0.09958			

How does R know what *full* and *reduced* models to use?

`anova()` tests model terms *sequentially*, by default ("Type 1 SS")

```
z <- lm(log(nspecies) ~ latitude * elevation)
anova(z)
```

If you don't give `anova()` explicit *full* and *reduced* models to compare (like I did earlier as `anova(z0,z1)`), R tests all terms following its own program of action.

1. `anova(z)` tests all model terms sequentially ("Type 1 SS") in the order you provided them in the formula.

2. `anova(z)` respects hierarchy: intercept tested first, then main effects, then interactions. To test an interaction between 2 or more variables, `anova(z)` uses a *reduced* model that contains the main effects of those variables.

With sequential testing, order of terms in the model formula matters

```
z <- lm(log(nspecies) ~ latitude * elevation)
anova(z)
```

	Df	Sum Sq	Mean Sq	*F*	Pr(>*F*)	
latitude	1	1.44425	1.44425	14.5030	0.0013	**
elevation	1	1.07581	1.07581	10.8032	0.0041	**
latitude:elevation	1	0.01091	0.01091	0.1096	0.7444	
Residuals	18	1.79249	0.09958			

Term	***Reduced* model**	***Full* model**	**Improvement in SS resid**
latitude	intercept	Intercept + latitude	1.44425
elevation	Intercept + latitude	Intercept + latitude + elevation	1.07581
latitude:elevation	Intercept + latitude + elevation	Intercept + latitude * elevation	0.01091

With sequential testing, order of terms in the model formula matters

```
z <- lm(log(nspecies) ~ elevation * latitude)
anova(z)
```

	Df	Sum Sq	Mean Sq	*F* value	Pr(>F)	
elevation	1	1.52670	1.52670	15.3309	0.0010	**
latitude	1	0.99336	0.99336	9.9752	0.0054	**
latitude:elevation	1	0.01091	0.01091	0.1096	0.7444	
Residuals	18	1.79249	0.09958			

Term	*Reduced* model	*Full* model	Improvement in SS resid
elevation	intercept	Intercept + elevation	1.52670
latitude	Intercept + elevation	Intercept + elevation + latitude	0.99336
latitude:elevation	Intercept + elevation + latitude	Intercept + elevation * latitude	0.01091

`Anova()` in the `car` package can fit model terms *marginally* ("Type 3 SS")

```
library(car)
z <- lm(log(nspecies) ~ latitude * elevation,
                  contrasts = c("contr.sum","contr.poly"))
Anova(z)
```

	Df	Sum Sq	*F* value	Pr(>F)	
latitude	1	0.40078	4.0246	0.0601	.
elevation	1	0.01643	0.1650	0.6894	
latitude:elevation	1	0.01091	0.1096	0.7444	
Residuals	18	1.79249			

Order of terms in model formula doesn't matter. Hierarchy is not respected. The improvement in SS residual for a given term in the *full* model is measured against a *reduced* model that contains all other terms, including any interactions. Marginal testing also called "drop 1" testing.

Type 3 SS is the default in SAS, JMP and some other computer packages.

Warning: The lure of model simplification

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

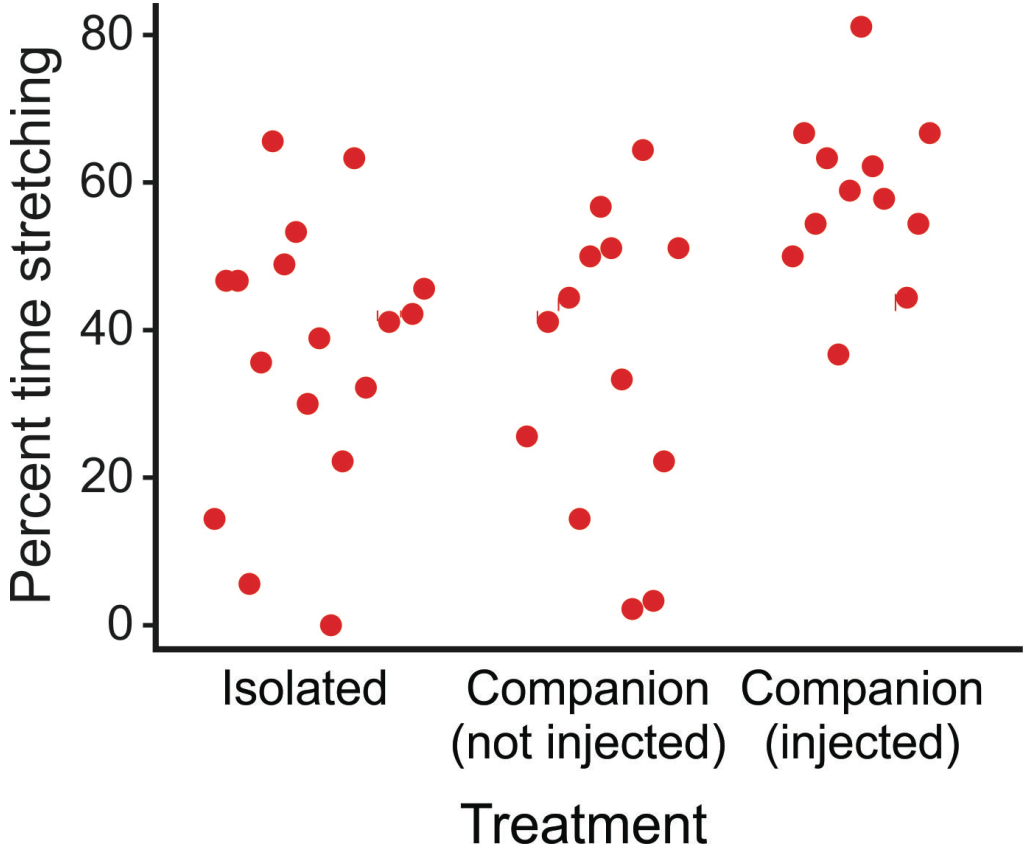
"models should be pared down until they are minimal adequate"

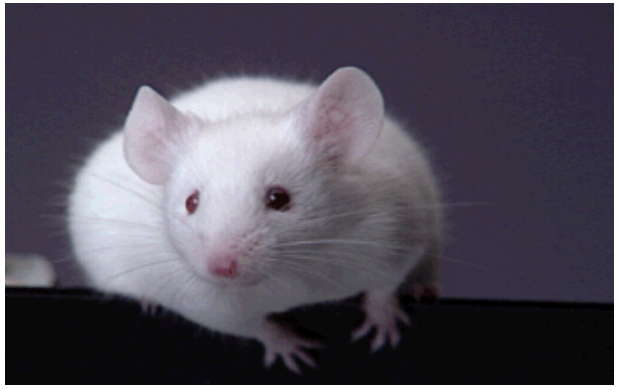
-- Crawley 2007, The R book, p325

- 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. Is this a good idea? Remaining *P*-values become heuristic.
- Later, we will cover the topic of model selection – how to choose the best model using explicit criteria for what constitutes "best."
- In the case of experiments, a good general rule is that *analysis should follow design*. Shouldn't a factor in your experiment also be in your linear model?

Example 3: Single-factor ANOVA

Data: the percentage of time that male mice given an injection to cause mild pain spent “stretching” in different familiar-companion treatments.





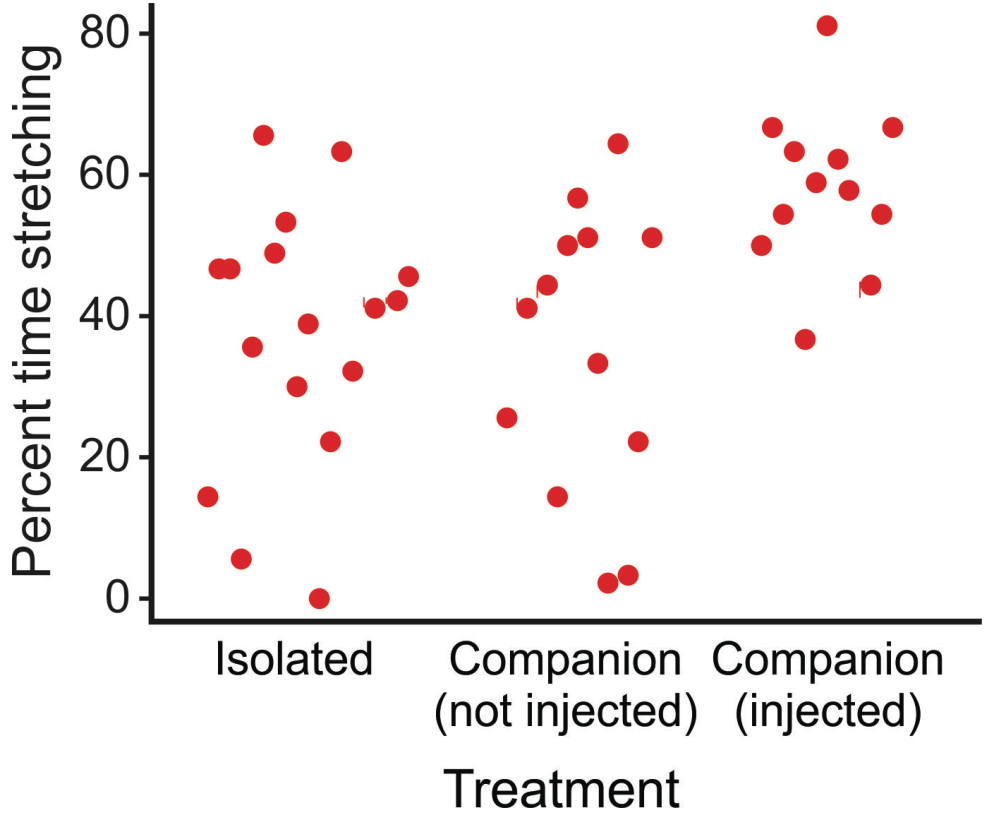
Langford, D. J.,et al. 2006. *Science* 312: 1967-1970

ANOVA is fundamentally the same as linear regression

There's a response variable, a constant, an explanatory variable.

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

The only difference is that the explanatory variable is categorical.



Use `summary()` to get <u>parameter estimates</u> (ignore the tests)

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

`summary(z)` # yields the estimates b_0, b_1, b_2 (Ignore the tests)

	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 incorrect except in the case of planned comparisons

What are b_0, b_1, b_2 ? I will explain.

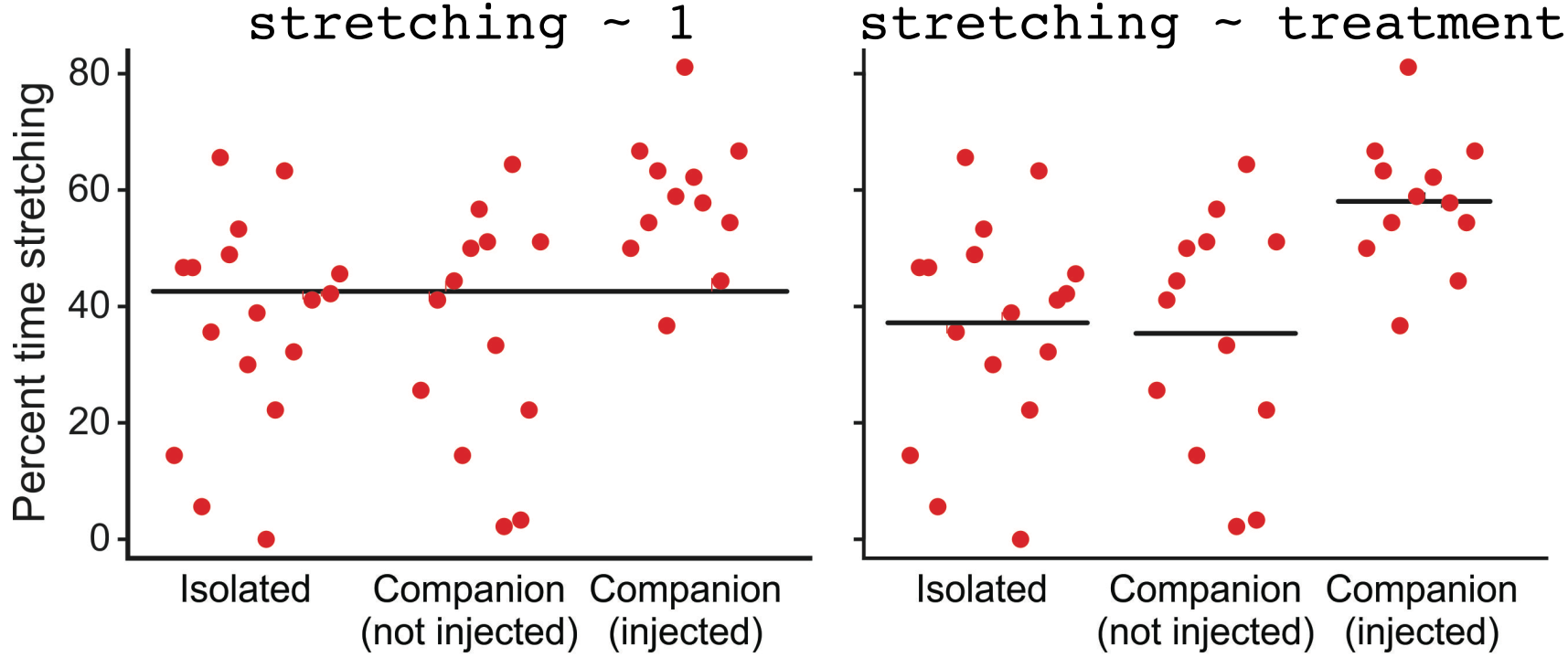
Let's look at the `anova()` table first.

Use `anova()` to test <u>hypotheses</u>

`anova(z)` # Produces the ANOVA table

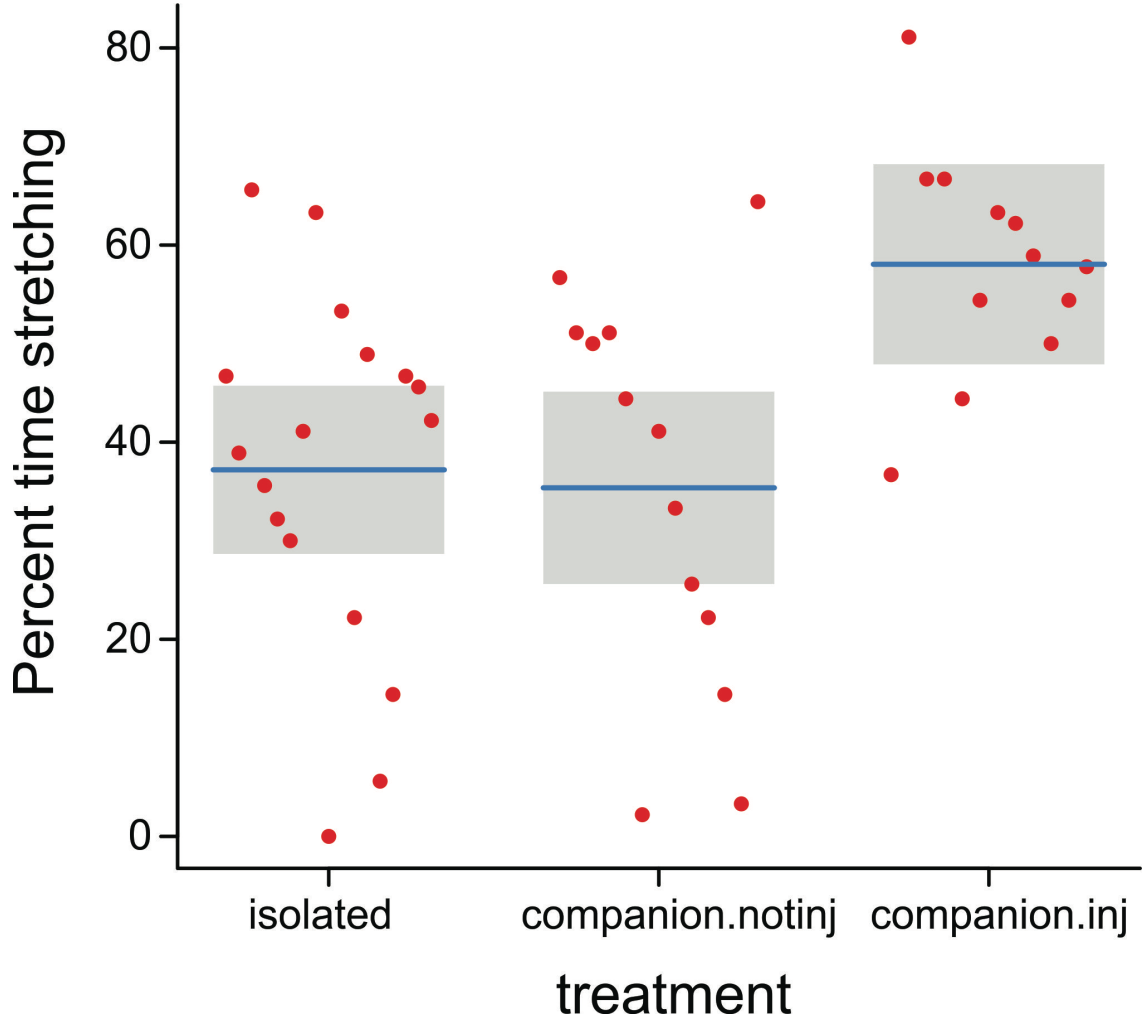
	Df	Sum Sq	Mean Sq	*F*	Pr(>*F*)
Treatment	2	4040.9	2020.5	6.6736	0.003216 **
Residuals	39	11807.4	302.8		

As before, each test in `anova()` compares the fit of TWO models:



Use `visreg()` to visualize model fits

```
visreg(z, "treatment")
```



Use `emmeans()` to get fitted means under the specific model

```
library(emmeans)
z <- lm(stretching ~ treatment)
emmeans(z, "treatment")
```

treatment	emmean	SE	df	lower.CL	upper.CL
isolated	37.19412	4.220082	39	28.65820	45.73004
companion.notinj	35.36923	4.825848	39	25.60803	45.13043
companion.inj	58.05000	5.022902	39	47.89022	68.20978

The SE's 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 (here, this is why df = 39 for each estimate).

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 factors.

What the `summary()` coefficients mean

```
z <- lm(stretching ~ treat)
summary(z)     # yields the following parameter estimates:
```

	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**

P-values are incorrect except for planned comparisons

What the `summary()` coefficients mean

Behind the scenes, R codes the 3 groups of the categorical variable with indicator variables that indicate group membership.

stretching	dummy	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

To analyze, R leaves out the indicator representing the first factor level to avoid a particular form of redundancy (a sum of three of the columns exactly equals the fourth). Use `model.matrix(z)` to see how indicators are coded.

Linear model for the indicator variables

stretching		dummy		treatcompanion		treatcompan.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	$= \beta_0$	1	$+ \beta_1$	1	$+ \beta_2$	0	+ residuals
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	

stretching $= \beta_0(1) + \beta_1(0) + \beta_2(0)$ + residual (subjects in isolation treatment)

stretching $= \beta_0(1) + \beta_1(1) + \beta_2(0)$ + residual (subjects in companion treatment)

stretching $= \beta_0(1) + \beta_1(0) + \beta_2(1)$ + residual (subjects in compan.inj treatment)

What the `summary()` coefficients mean

In other words, the linear model being fitted is:

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

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

β_0 is the mean of the isolated (control) group

β_1 is the difference between companion and control groups

β_2 is the difference between compan.inj and control groups

(Other codings are possible, in which case the interpretations of the parameters will change. Read the fine print. R's 0/1 scheme is relatively straightforward.)

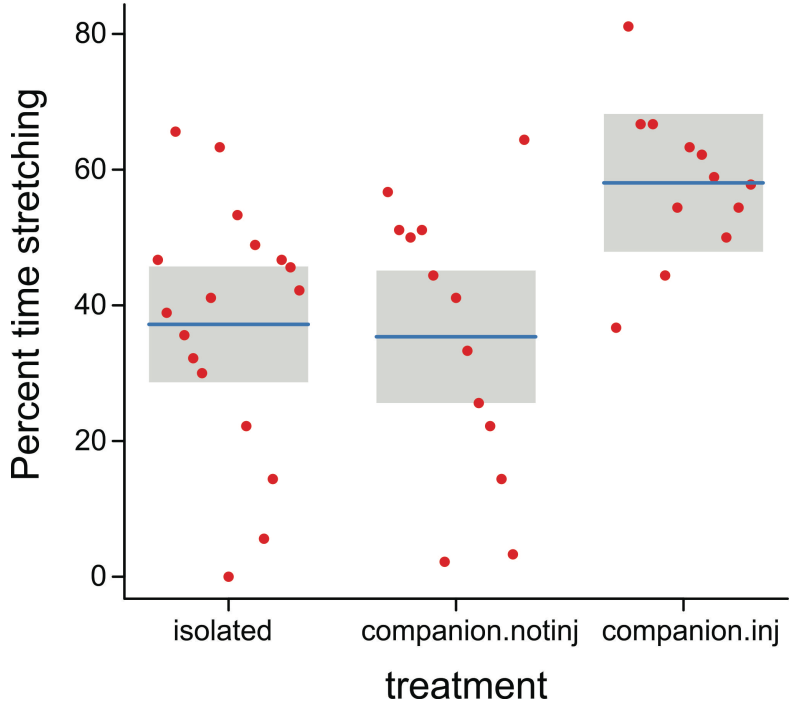
What the `summary()` coefficients mean

b_0 estimates the mean of the isolated (control) group

b_1 estimates the difference between companion and control groups

b_2 estimates the difference between compan.inj and control groups

	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**



How does `anova()` test a term having more than one indicator variable?

To test a factor/term, the *reduced* model drops <u>all</u> columns coding for that factor

In this example, the three levels of treatment are coded by two dummy indicator variables, both of which are dropped in the *reduced* model.

```
z0 <- lm(percent.stretching ~ 1)          # reduced model (1 column)
z1 <- lm(percent.stretching ~ treatment) # full model (3 columns)
anova(z0,z1)
```

	Res.Df	RSS	Df	Sum.of.Sq	*F*	Pr(>*F*)	
1 [reduced]	41	15848					
2 [full]	39	11807	2	4040.9	6.6736	0.003216	**

Summary of Example 3 so far

- Linear models can fit categorical variables too.
- Use `visreg()` to visualize model fits.
- Use `emmeans()` to estimate predicted group means.
- Use `summary()` for parameter estimation. To interpret the estimates, it is useful to know about how R handles categorical variables behind the scenes (0/1 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 `anova()` or `Anova()` for hypothesis testing (*P* values, sums of squares).
- Use `plot()` to check assumptions (workshop)

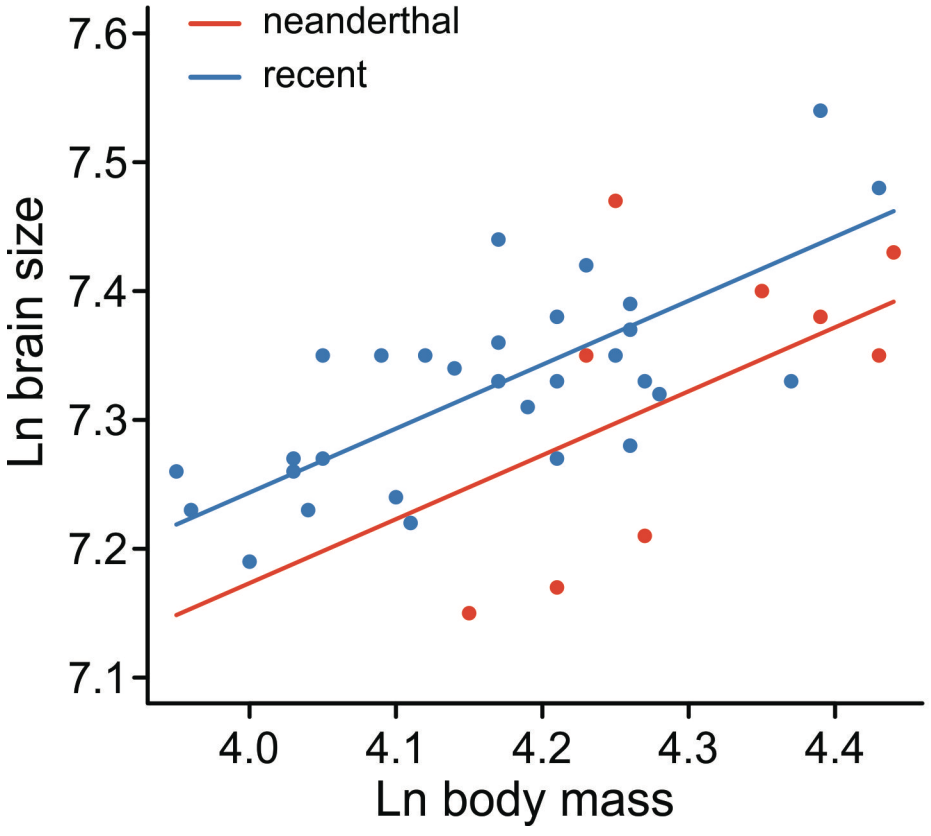
Example 4: Models with both numeric and categorical variables (ANCOVA)

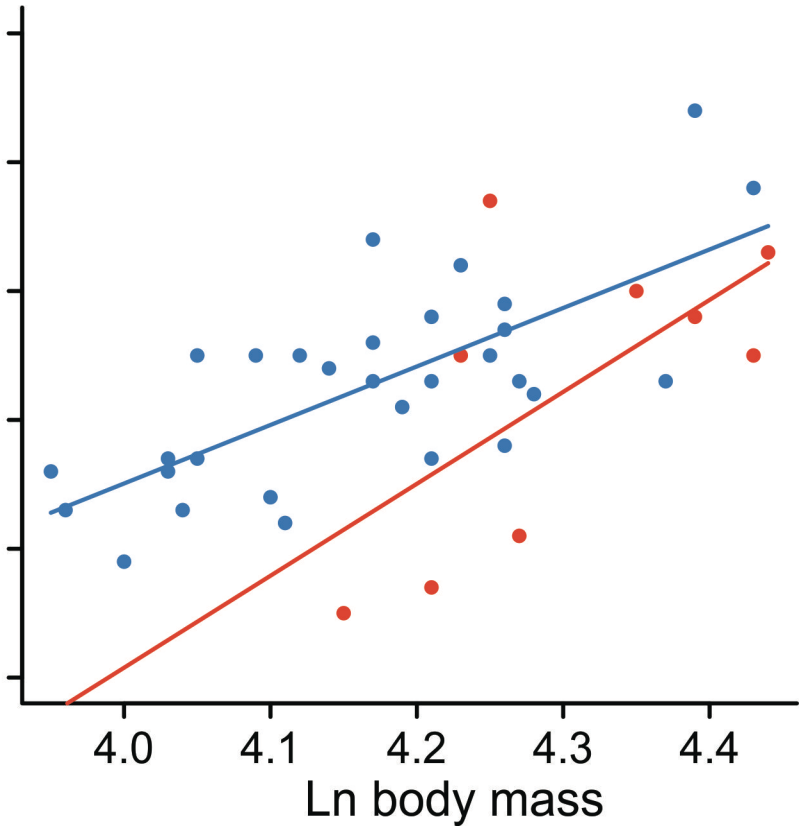
Brain and body sizes of Neanderthal specimens (●) and early modern humans (●). Ruff et al 1977).

Do they (we) have different brain sizes, after accounting for differences in body size?
Answering this is easiest if we can assume the model on the left is correct.

```
brain ~ mass + species
```

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





`anova()` tests terms sequentially

```
z <- lm(brain ~ mass * species)
anova(z)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
mass	1	0.102528	0.102528	23.1465	2.835e-05	***
species	1	0.027553	0.027553	6.2203	0.0175	*
mass:species	1	0.004845	0.004845	1.0938	0.3028	
Residuals	35	0.155033	0.004430			

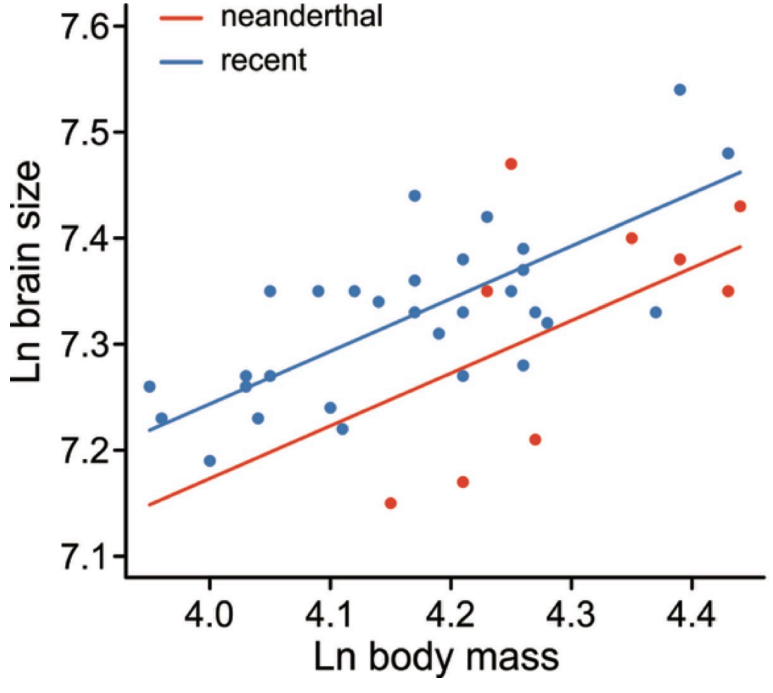
Interaction is not significant, but equal slopes remains an assumption not a conclusion (one not contradicted by the data).

`summary()` obtains the parameter estimates

Model with no interaction (equal slopes)

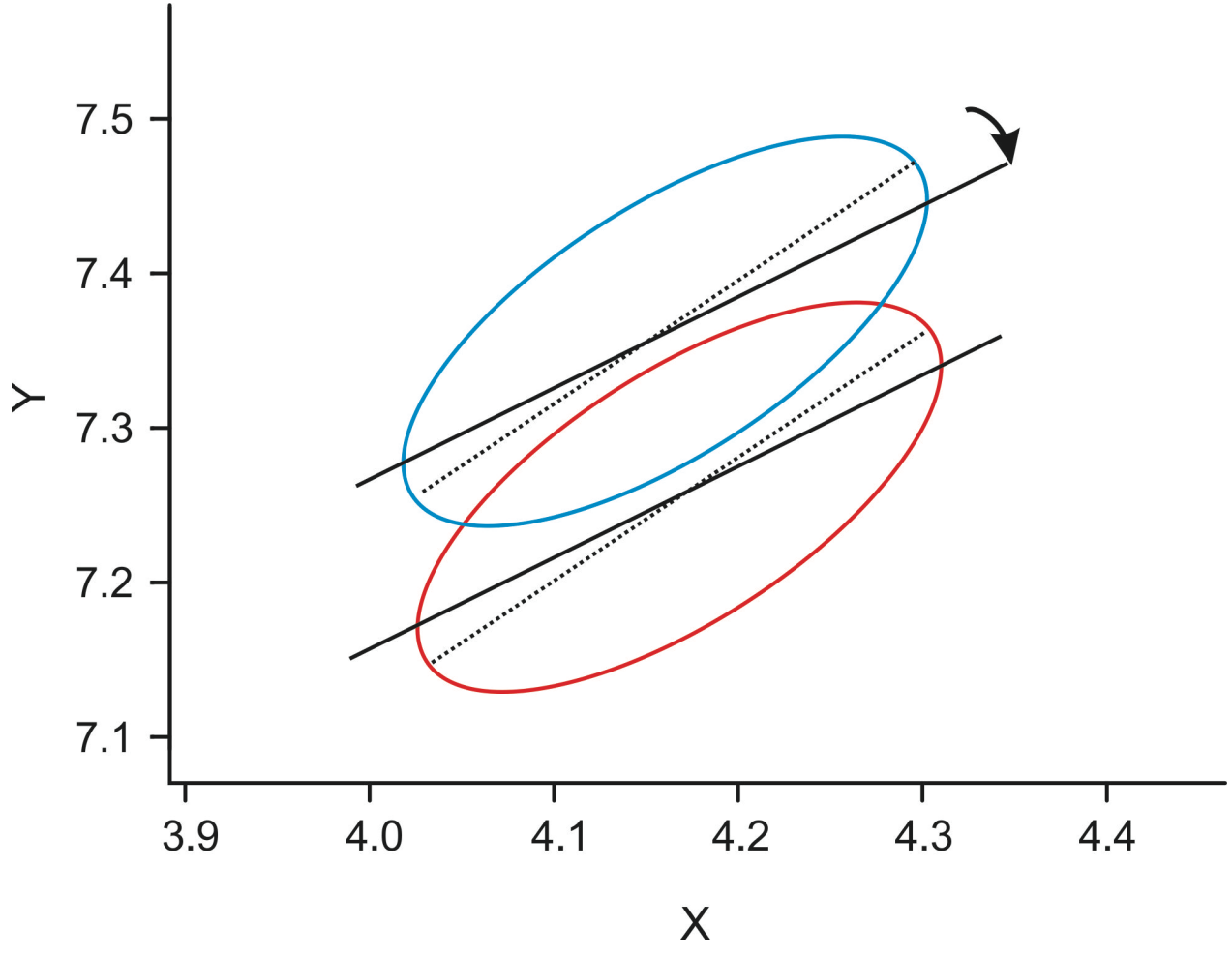
```
z <- lm(brain ~ mass + species)
summary(z)
```

	Estimate	Std. Error	Interpretation of parameters estimated
(Intercept)	5.22321	0.38862	Intercept for species 1 (recent humans)
lnmass	0.49632	0.09173	Slope for species 1 (same slope fit to both)
species1	-0.03514	0.01411	Difference between intercepts (i.e., size-corrected difference)



Size-correction is valid only when range of X-values is similar in all groups

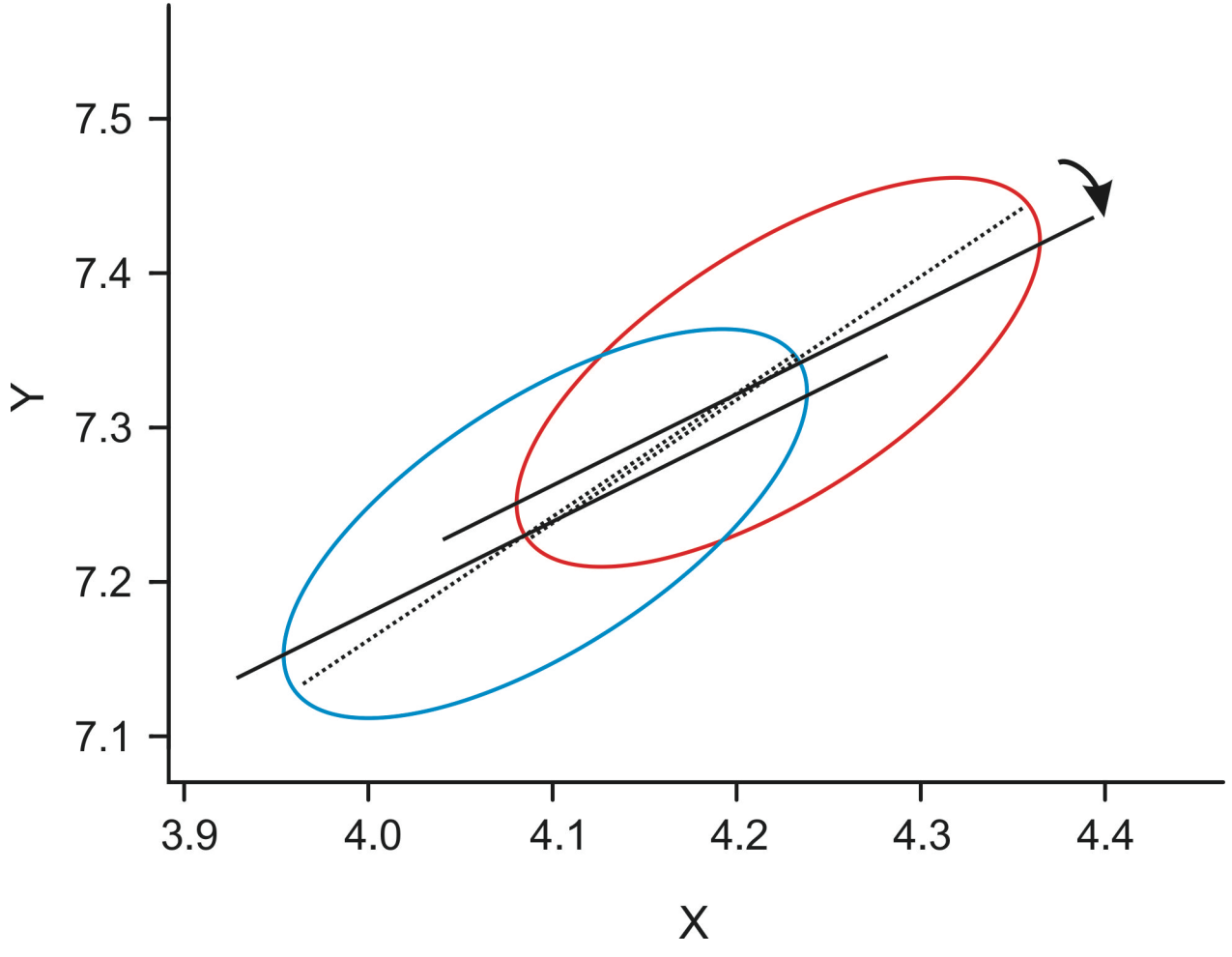
Although our goal is to “correct” for variation in X in order to comparing Y among groups, X is not the cause of Y. Hence, there is “regression toward the mean”.



Problems arise when the range of X-values is not the same among groups

Differences in Y might persist even after “correcting” for differences in X.

Major axis regression methods are more suitable instead (available in R!).



Core assumptions of linear models

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

Use `plot(z)` to assess departures from the assumptions of normality and equal variance (workshop this week).

Linear models are reasonably robust to departures from these assumptions, especially if sample size is large and balanced. However, outliers can cause problems.

The assumption of independent errors is met if your sample constitutes a random sample and you have not pseudoreplicated.

Related topics

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

- Robust regression methods (`rlm`)

What if response data are binary or discrete?

- Generalized linear models (`glm`)

What if there are random effects?

- Linear mixed effects models (`lme`)

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

- General least squares methods (`gls`)

Discussion paper:

Kelly and Price (2005). Correcting for regression to the mean in behavior and ecology. *American Naturalist* 166: 700-707.

Download from "**assignments**" tab on course web site.

Presenters: _____ & _____

Moderators: _____ & _____

A word about planned vs unplanned comparisons

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.
- *P*-values in `summary()` table are not protected, so can't use them.

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.
- Other types of planned contrasts are also possible (`glht()` command in `multcomp` package)