Outline

Interaction between 2 categorical predictors

Two-way interaction between X_1 (k_1 levels, think gender) and X_2 (k_2 levels, think food type): when the mean response (Y or log odds) differences between X_1 levels depend on the level of X_2.

- With *no* interaction, there are constraints on group means. X_1 requires $k_1 - 1$ extra coefficients (other than intercept), X_2 requires $k_2 - 1$ extra coefficients.
- In R, $X_1 * X_2$ is a shortcut for $X_1 + X_2 + X_1 : X_2$.
- The interaction $X_1 : X_2$ requires $(k_1 - 1) * (k_2 - 1)$ df, i.e. extra coefficients.
- *With* interaction, total # of coefficients (including intercept) = total # of groups, $k_1 * k_2$: No constraints on groups means.

Degrees of freedom

For model $Y \sim X_1 * X_2$:

source	df
intercept	1
X_1	$k_1 - 1$
X_2	$k_2 - 1$
$X_1 : X_2$	$(k_1 - 1)(k_2 - 1)$
total # coefs	$k_1 * k_2$
residual	$n - k_1 k_2$

where *n* is the number of observations (rows) in the data.

Interactions between 3 categorical predictors

Two-way interactions between X_1, X_2 and X_3 (k_3 levels, think preterm):

- $X_1 : X_2$ is the 2-way interaction between X_1 and X_2 when $X_3 = 0$ or reference level.
- $X_1 : X_3$ is the 2-way interaction between X_1 and X_3 when $X_2 = 0$ or reference level.
- $X_2 : X_3$ is the 2-way interaction between X_2 and X_3 when $X_1 = 0$ or reference level.

There is a three-way interaction $X_1 : X_2 : X_3$ if the interaction coefficient(s) $X_1 : X_2$ depend on the level of X_3, or, equivalently, if the $X_2 : X_3$ interaction coefficient(s) depend on the level of X_1, or if the $X_1 : X_3$ interaction coefficient(s) depends on the level of X_2.

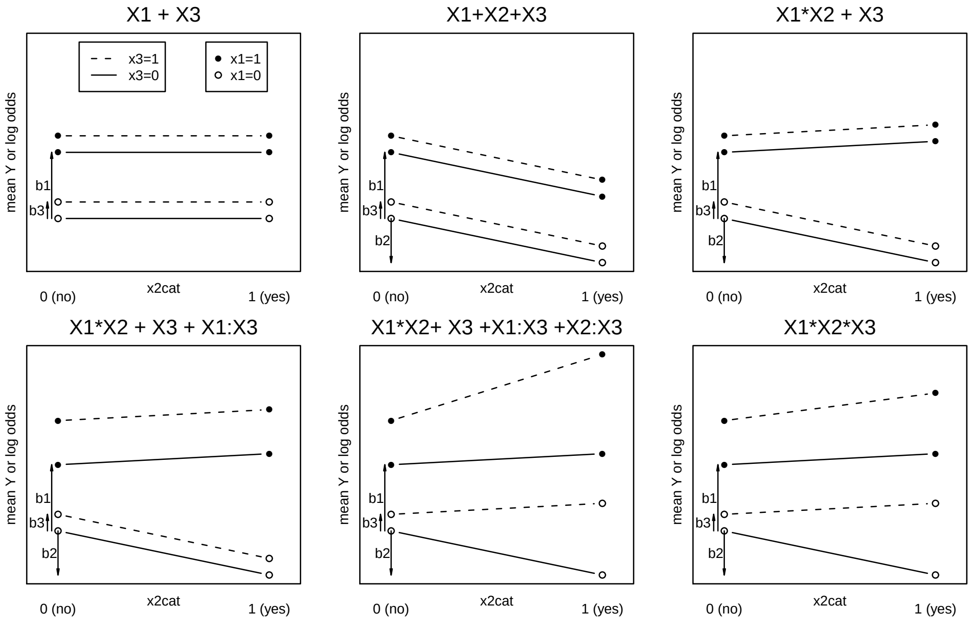
- With the 3-way interaction and all 2-way interactions, the total # of coefficients (including intercept) equals the total # of groups, $k_1 * k_2 * k_3$: No constraints on groups means.

Degrees of freedom

For $Y \sim X_1 * X_2 * X_3$:

source	df
intercept	1
X_1	$k_1 - 1$
X_2	$k_2 - 1$
$X_1 : X_2$	$(k_1 - 1)(k_2 - 1)$
$X_1 : X_3$	$(k_1 - 1)(k_3 - 1)$
$X_2 : X_3$	$(k_2 - 1)(k_3 - 1)$
$X_1 : X_2 : X_3$	$(k_1 - 1)(k_2 - 1)(k_3 - 1)$
total # coef	$k_1 * k_2 * k_3$
residual	$n - k_1 k_2 k_3$

where n is the number of observations (rows) in the data.


X1 + X3
X1+X2+X3
X1*X2 + X3
X1*X2 + X3 + X1:X3
X1*X2+ X3 +X1:X3 +X2:X3
X1*X2*X3
x3=1
x3=0
x1=1
x1=0
mean Y or log odds
x2cat
0 (no)
1 (yes)
b1
b2
b3

Notations in R

(and many other programs)

These are equivalent notations for model formulas:

- $(X_1 + X_2)^2$ and $X_1 * X_2$, and $X_1 + X_2 + X_1 : X_2$.
- Main effects and one 2-way interaction: $X_1 * X_2 + X_3$ and $X_1 + X_2 + X_3 + X_1 : X_2$
- Mains effects and two 2-way interactions: $X_1 * X_2 + X_3 + X_2 : X_3$ and $X_1 + X_2 + X_3 + X_1 : X_2 + X_2 : X_3$ and $X_1 + X_2 * X_3 + X_1 : X_2$
- Main effects and all 2-way interactions: $X_1 * X_2 + X_3 + X_2 : X_3 + X_1 : X_3$ and $(X_1 + X_2 + X_3)^2$
- All: $X_1 * X_2 * X_3$ and $X_1 + X_2 + X_3 + X_1 : X_2 + X_1 : X_3 + X_2 : X_3$ and $(X_1 + X_2 + X_3)^3$ (and many others).

Hierarchy principle

Include all lower-level interactions.

- If we include $X_1 : X_2$, then we must also include X_1 and X_2.

  If we allow X_1 to have an effect that depends on the level of X_2, it makes little sense to assume that X_1 has no effect when $X_2 = 0$ or reference level.
- If we include $X_1 : X_2 : X_3$ then we must also include all three 2-way interactions ($X_1 : X_2$, $X_1 : X_3$, $X_2 : X_3$) and all three main effects (X_1, X_2, X_3).
- A predictor X_1 is said to have an effect as soon as one interaction with X_1 is present.

Hierarchy principle

Example: if breastfeeding (X_1) has no effect except on preterm baby girls, and if the reference levels are bottle feeding, boys, full term, then we may find

source	df	coef	p-value
breast	1	0.01	.92
girl	1	...	...
preterm	1	...	...
breast:girl	1	0.02	.85
breast:preterm	1	0.01	.87
girl:preterm	1	...	...
breast:girl:preterm	1	1.5	$< 10^{-5}$ $***$

Then there is evidence that breastfeeding has an effect.

Interpretation

When we find complex interactions, we often break down the data into smaller data sets, for ease of interpretation.

If there is evidence that breastfeeding affects the probability of respiratory disease differently in boys/girls and in preterm/full term babies, then we may **break down** the data into 2 sets of data: preterm babies only, and full term babies only.

- cons: not one, but 2 analyses are required.
- pros: no need to consider 3-way interactions and simpler interpretation: only two-way interaction (breast/girl) in each analysis.

Outline

Dispersion

The binomial distribution assumes that

$$\mathrm{var}(Y_i) = n_i p_i (1 - p_i)$$

In real situations, the variance could differ from $n_i p_i (1 - p_i)$ if the n_i individuals from observation i are not independent (all from the same inoculation batch? all from the same plate? etc.) Could it be that

$$\mathrm{var}(Y_i) = \sigma^2 n_i p_i (1 - p_i) \text{ for some } \sigma^2 \text{ ??}$$

violating the binomial assumption? σ^2 is called the dispersion.

- $\sigma^2 < 1$: underdispersion (rare)
- $\sigma^2 = 1$: binomial okay: not violated
- $\sigma^2 > 1$: overdispersion

Dispersion

We can estimate the dispersion in two (non-equivalent) ways.

- from the Pearson residuals, used in `summary` and `anova`:

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n} (r_i^P)^2}{n-k}$$

  Most appropriate because the Pearson residuals were standardized based on their estimated SD:

$$r_i^P = \frac{y_i - n_i \hat{p}_i}{\hat{n}_i p_i (1 - \hat{p}_i)}$$

- from the deviance residuals, used in `drop1`, technically easier:

$$\hat{\sigma}_D^2 = \frac{\sum_{i=1}^{n} (r_i^D)^2}{n-k} = \frac{\text{resisual D}}{\text{residual df}}$$

- often, these two estimates are very similar.

If the data truly follow a Binomial distribution: the dispersion should be ~ 1.

Bush lupine seedling survival

Do entomopathogenic nematodes protect plants indirectly, by killing root-feeding ghost moth caterpillars? (Strong et al, 1999).

Field experiment with 120 bush lupine seedlings, randomly assigned to treatments (15 per treatment): with 0 or up to 32 hatchlings of ghost moth caterpillars, and with/without nematodes.

```
> dat = read.table("lupine.txt", header=T)
> dat$n = 15
> dat
  caterpillars nematodes lupine  n
1            0    absent     15 15
2            0   present     14 15
3            8    absent     11 15
4            8   present     14 15
5           16    absent      8 15
6           16   present     13 15
7           32    absent      5 15
8           32   present     13 15
```

Bush lupine seedling survival

```
> fit = glm(lupine/n ˜caterpillars+nematodes, family=binomial,
            weights=n, data=dat)
> summary(fit)

Deviance Residuals:
     1      2      3      4      5      6      7      8
 2.191 -0.773 -0.248 -0.291 -0.826 -0.606 -0.131  0.999
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       1.75137    0.45567   3.844 0.000121 ***
caterpillars     -0.07417    0.02142  -3.463 0.000535 ***
nematodespresent  1.79561    0.55082   3.260 0.001115 **

(Dispersion parameter for binomial family taken to be 1)
Residual deviance:  7.6087  on 5  degrees of freedom

> resP = residuals(fit, type="pearson")
> resP
     1      2      3      4      5      6      7      8
 1.613 -0.908 -0.251 -0.305 -0.840 -0.650 -0.130  0.938
```

Bush lupine seedling survival

```
> sum(resPˆ2)/(8-3)
[1] 1.121281     # based on Pearson’s residuals: better
> sum(residuals(fit, type="deviance")ˆ2)/5
[1] 1.521735     # based on deviance
> fit$deviance
[1] 7.608677
> fit$df.residual
[1] 5
> fit$deviance / fit$df.residual # easier way to estimate the
[1] 1.521735     # dispersion based on deviance residuals
```

Here, the dispersion parameter is estimated to be 1.12, more than 1: slight overdispersion. Can this difference be due to chance alone?

Bush lupine seedling survival

Is overdispersion significant?

- Test for lack of fit: compare the residual deviance to a χ^2 distribution with df = df residual. But many other reasons can result in a lack of fit.

  ```
  > pchisq(7.608677, df=5, lower.tail=F)
  [1] 0.1791618
  ```

  Here: no significant lack of fit, interpreted as no significant overdispersion.

- With a simulation –more later.

If we really need to, how to account for over-dispersion?

Quasi-Binomial analysis

We can account for overdispersion (or underdispersion) using a quasi-binomial analysis.

```
> fitQuasi = glm(lupine/n ~ caterpillars+nematodes,
                 family=quasibinomial, weights=n, data=dat)
> summary(fitQuasi)

Deviance Residuals:
     1       2       3       4       5       6       7       8
 2.191 -0.773 -0.248 -0.291 -0.826 -0.606 -0.131  0.999
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       1.75137    0.48251   3.630   0.0151 *
caterpillars     -0.07417    0.02268  -3.270   0.0222 *
nematodespresent  1.79561    0.58327   3.079   0.0275 *

(Dispersion parameter for quasibinomial family taken to
                                                 be 1.121281)
    Null deviance: 32.4825  on 7  degrees of freedom
Residual deviance:  7.6087  on 5  degrees of freedom
```

Quasi-Binomial analysis

```
> summary(fit)  # was standard logistic regression
...
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       1.75137    0.45567   3.844 0.000121 ***
caterpillars     -0.07417    0.02142  -3.463 0.000535 ***
nematodespresent  1.79561    0.55082   3.260 0.001115 **
> 0.45567 * sqrt(1.121281)
[1] 0.48251
> pt(3.630, df=5, lower.tail=F) * 2
[1] 0.01506
```

Compared with the standard binomial (logistic) regression:

- Same deviance residuals, same null/residual deviance, same df's.
- Same estimated coefficients
- Different SE for coefficients: there were multiplied by $\hat{\sigma}$
- Wald z-tests in logistic regression become t-tests in quasi-binomial regression, on df=residual df.

Model comparison

Analysis of Deviance in logistic regression (which uses the χ^2 distribution) becomes ANOVA in a quasi-binomial analysis: uses the F distribution as in standard regression, where Sums of Squares are replaced by deviances and MSError $= \hat{\sigma}^2$ is replaced by the dispersion.

```
> anova(fitQuasi, test="F")
Terms added sequentially (first to last)
             Df Deviance Resid.Df Resid.Dev      F  Pr(>F)
NULL                            7    32.482
caterpillars  1   12.183        6    20.300 10.865 0.02156 *
nematodes     1   12.691        5     7.609 11.318 0.02001 *

> (32.482-20.300)/(7-6) / 1.12128
[1] 10.86437
> pf(10.865, df1=1, df2=5, lower.tail=F)   # df1=7-6=1
[1] 0.02156
> (20.300-7.609)/(6-5) / 1.12128
[1] 11.31831
> pf(11.318, df1=1, df2=5, lower.tail=F)
[1] 0.02001
```

Outline

Why use simulations?

to assess uncertainty in predictions, in estimated regression coefficients, etc. This is in contrast to deriving formulas for standard errors.

pros:

- we do not need to learn how to derive formulas.
- we can use simulations even when formulas are only approximations
- we can assess uncertainty in quantities for which there are not formulas.

cons:

- we will learn how to write small programs in R.
- we will do ‘parametric bootstrapping’.

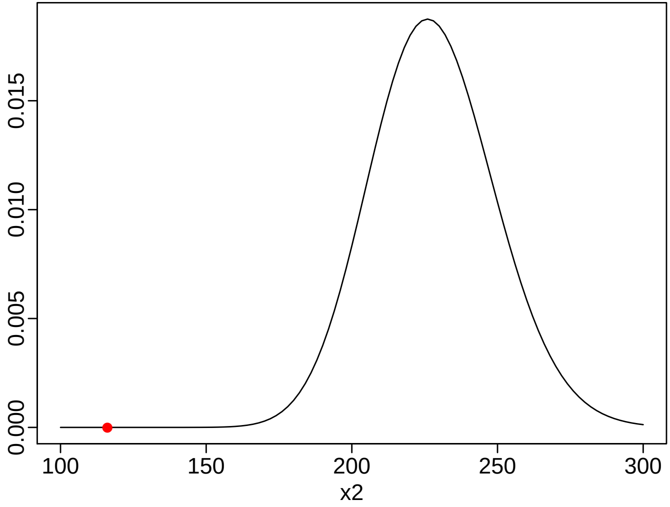
Recall runoff data

We explained runoff events with total amount of precipition (inches) and maximum intensity at 10 minutes (in/min).
We used logistic regression (why?)

```
> fit2 = glm(RunoffEvent˜Precip+MaxIntensity10,
             family=binomial, data=runoff)
> summary(fit2)
...
Residual deviance: 116.11  on 228  degrees of freedom

> fit2$df.residual
[1] 228
> fit2$deviance
[1] 116.10591             # seems too low
> pchisq(116.11, df=228)  # P(X2 < 116.11)
[1] 5.765843e-11
> curve(dchisq(x,df=228), from=100,to=300, xlab="x2")
> points(116.11, 0, col="red", pch=16)
```

Is there real underdispersion?



Why is the deviance so low? Also, $\hat{\sigma}^2 = 0.65$. Underdispersion?

```
> sum( residuals(fit2, type="pearson")ˆ2 )/228
[1] 0.6524366
```

Simulating the distribution of the residual deviance

"Formula": When *sample sizes are large* and if the model is correct, then the residual deviance is approximately χ^2 distributed, on residual df.

Here: deviance$=$ 116.11 on df$=$ 228. The "formula" gives us a very low p-value and suggests underdispersion. But what can we trust?

To see what the deviance is "just by chance", we will

- simulate new data sets under our hypothesized model: use our estimated model and repeat many experiments *in silico*. Our model will be correct (H_0 true) for these *in silico* experiments.
- apply the treatment we applied to our original data set,
- calculate the deviance of each of these new data sets,
- repeat many times, summarize the deviances

Simulating one new data set

```
> dim(runoff)   # 231 11
> mu=predict(fit2, type="response") # estimated probabilities
> mu                               # of runoff events for each storm
    1     2     3     4     5     6     7     8     9    10
0.140 0.026 0.018 0.028 0.018 0.051 0.021 0.249 0.025 0.927 0.
...
  222   223   224   225   226   227   228   229   230   231
0.675 0.638 0.099 0.031 0.763 0.689 0.899 1.000 0.257 0.993

> sim.events = rbinom(231, size=1, prob=mu)   # One experiment
> sim.events                                  # was simulated
  [1] 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 ..
 [38] 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 ..
...
[223] 1 0 0 1 1 1 1 1 1

> sim.data = data.frame(
+    Precip          = runoff$Precip,
+    MaxIntensity10 = runoff$MaxIntensity10,
+    RunoffEvent     = sim.events
+ )
```

Simulating one new data set

```
> sim.data  # just to check the new data set
    Precip MaxIntensity10 RunoffEvent
1     0.47           0.96           0
2     0.34           0.18           0
3     0.16           0.24           0
...
186   0.11           0.18           1
187   0.16           0.48           0
...
228   1.07           2.22           1
229   2.41           3.12           1
230   0.58           1.20           1
231   1.75           2.70           1

# now apply the same treatment
> sim.fit = glm(RunoffEvent ~ Precip + MaxIntensity10,
+               data=sim.data, family=quasibinomial)
> sim.fit$deviance
[1] 141.73432  # this is random: from one experiment.
```

We got this 141.7 deviance 'just by chance': the model (binomial, Precip + MaxIntensity10) was true.

Simulating many new data sets

First define a function to make and analyze a single data set:

```
> simulate.deviance = function(){
+  sim.data = data.frame(
+     Precip          = runoff$Precip,
+     MaxIntensity10 = runoff$MaxIntensity10,
+     RunoffEvent     = rbinom(231, size=1, prob=mu)
+  )
+  sim.fit = glm(RunoffEvent ~ Precip + MaxIntensity10,
+                   data=sim.data, family=binomial)
+  return( sim.fit$deviance )
+ }

> simulate.deviance() # check that this function works
[1] 87.839458         # this is random
> simulate.deviance()
[1] 111.6603          # new random deviance from new expt
> simulate.deviance()
[1] 102.7747          # from another new experiment
```

Simulating many new data sets

We replicate this simulation many times (1000 usually enough).

`replicate(n, function)` : needs a number n of times and a `function` to be repeated.

```
> sim1000 = replicate(1000, simulate.deviance())
> sim1000
   [1] 109.1 102.3  94.1 108.8  73.3  94.9 115.1  99.2 114.9 1
  [13] 102.9 102.5 105.9 135.1 120.0 150.1 131.5  93.7 100.3 1
  [25] 141.9 109.7 103.8 128.4 113.0 148.5  92.5  72.4 106.6 1
...
 [961] 114.4 117.6 108.0  91.6 103.5 129.5 110.0  96.8 100.6
 [973]  99.3 116.7 109.9 109.5 112.5 107.9 133.3  94.5 117.4 1
 [985] 111.6 107.6 108.7 135.0 114.4 127.7 100.8 106.2 112.5 1
 [997] 109.1 122.4 105.6 143.3
```

Summarizing simulated deviances

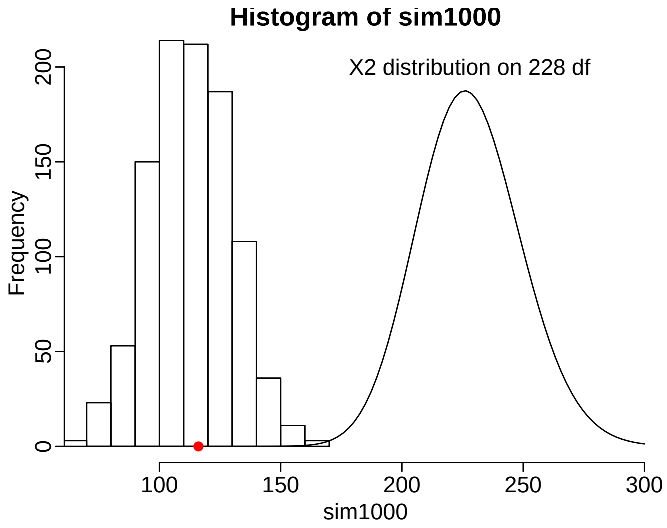
Now we summarize the 1000 deviance values, which were obtained “by chance” under our hypothesized model:

```
> hist(sim100)
```

Overlay the the simulated distribution with the theoretical χ^2 distribution... which we know is a bad approximation because all sample sizes are 1.

```
> hist(sim1000, xlim=c(70,300))
> curve(1000*10*dchisq(x,df=228), add=T)
  # I used 1000 * 10 * chi-square_density   in order to match
  # the area under the curve (1000 * 10 * 1) with
  # the area of the histogram: 1000 points * width 10 of each
> text("X2 distribution on 228 df", x=228, y=200)
> points(116.11, 0, col="red", pch=16)
```

Summarizing simulated deviances



No sign of lack of fit: so far, it looks like our binomial model is adequate.

Testing lack of fit with simulations

Conclusions:

- In the runoff experiment, the distribution of the deviance under H_0: “our model is correct” is very far from a χ^2 distribution on df=residual df.
- In this case (all $n_i = 1$) we should not trust the p-value obtained from comparing the residual deviance to the χ^2 distribution on residual df (was 5.10^{-11}).
- Instead, we should test H_0: “the model is correct” versus lack of fit with simulations.
- How often is the deviance even lower than 116.11 ‘just by chance’? We obtain p-value $= 0.57$: No lack of fit, No evidence of underdispersion.

Testing lack of fit with simulations

```
> sim1000 < 116.11
   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FA
  [13]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FA
...
 [985]  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FA
 [997]  TRUE FALSE  TRUE FALSE

> sum( (sim1000 < 116.11))
 # true=1 false=0, so the ’sum’ of the true/false values
 # will be the number of ’true’s.
[1] 573
> sum( (sim1000 < 116.11)) / 1000
[1] 0.573
```

p-value $=$ 0.57 $=$ probability of observing a deviance of 116.11 or smaller, when the model is really true. Obtained by parametric boostrapping.

Simulation to test for overdispersion: lupine seedlings

We found a dispersion of 1.12, slighly over 1. Is this overdispersion?

```
> fit = glm(lupine/15 ˜ caterpillars+nematodes,
            weight=size, data=lup, family=binomial)
> fit$df.residual
[1] 5
> sum( residuals(fit, type="pearson")ˆ2 ) / 5
[1] 1.121281   # this is the estimated dispersion, sigma2 hat.
```

We will simulate many *in silico* new data sets, for which the model with caterpillars + nematodes and the binomial distribution will be true. Then see what is the distribution of the estimated dispersion until this null hypothesis.

Simulation to test for overdispersion: lupine seedlings

First, write a function to simulate one new experiment, analyze the data, and extract the estimated dispersion.

```
probs = fitted(fit) # estimated probs of lupine survival
probs = predict(fit, type="response")  # equivalent

simulate.dispersion = function(){
 sim.data = data.frame(
  caterpillars = lup$caterpillars,
  nematodes    = lup$nematodes,
  size         = lup$size,
  lupine       = rbinom(8, size=15, prob=probs)
 )
 sim.fit = glm(lupine/15 ˜ caterpillars+nematodes,
          weight=size, data=sim.data, family=binomial)
 sim.s2 = sum( residuals(sim.fit, type="pearson")ˆ2 )/5
 return(sim.s2)
}
```

Simulate many new $\hat{\sigma}^2$ under H_0: no overdispersion

Now apply the function, a few times to check that it works, then replicate many times (and save the values):

```
> simulate.dispersion()
[1] 0.7293403
> simulate.dispersion()
[1] 0.4849162
> simulate.dispersion()
[1] 0.4526491
> simulate.dispersion()
[1] 1.440982
> sim1000s2 = replicate(1000, simulate.dispersion())
> sim1000s2
   [1] 0.611 0.482 0.355 0.639 1.181 0.727 1.612 1.683 1.088 0
...
 [985] 1.068 2.313 0.986 1.028 0.906 1.108 2.506 0.534 1.130 0
 [997] 1.545 0.679 2.502 2.242
```

Summarize the simulated dispersion values

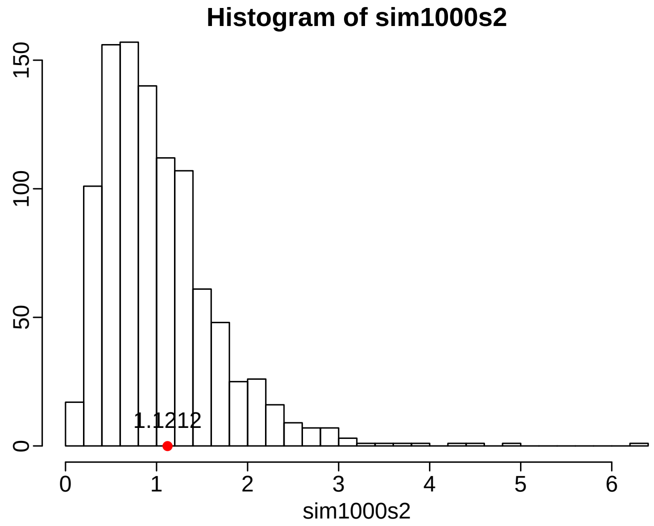
The null distribution of the estimated dispersion is skewed left:

```
> summary(sim1000s2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.09192 0.57320 0.90550 1.02300 1.32800 6.36700
> hist(sim1000s2, breaks=30)
  # I asked for ˜ 30 breakpoints in the histogram
> points(x=1.121281, y=0, col="red", pch=16)
  # placed a point at the dispersion for the original data
> text("1.1212", x=1.121281, y=10)

> sum( sim1000s2 > 1.121281 ) / 1000
[1] 0.357   # proportion of simulated s2 that are > 1.121281
```

p-value $=$ 0.36 to test the null hypothesis that our model is correct with no overdispersion. We accept it.

Visualize the simulated dispersion values


Histogram of sim1000s2
sim1000s2