Simulation and Randomization

Zack Treisman

Spring 2026

Philosophy

Question: Given a statistical model, how do we evaluate it?

model + computer $\implies$ generate simulated data.

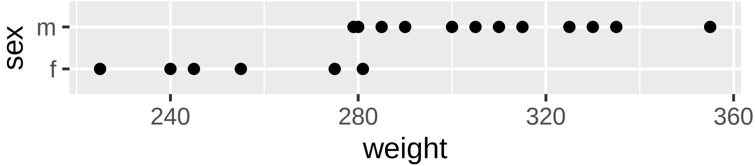
With simulated data we can:

- Compare models to data (hypothesis testing)
- Make predictions (confidence intervals)
- Plan data collection (power analysis)

```r
set.seed(222)
```

A one variable example

A study in Kenward et al. (2004) examined sexual dimorphism in New Caledonian crows. Sex and weight for 18 birds were recorded.



Crow example (cont.)

This is a classic scenario that calls for a *t*-test.

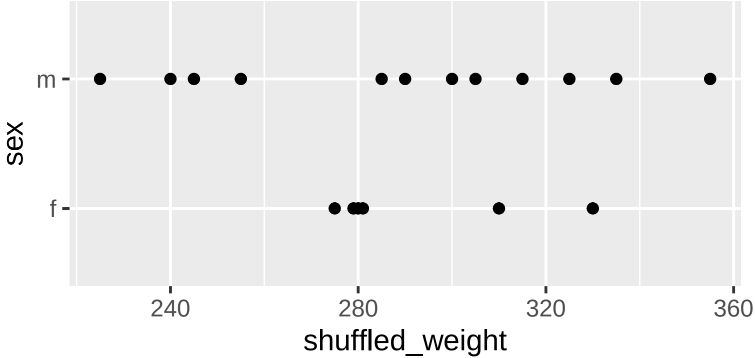
```r
t.test(weight~sex, data = crows)
```

```
##
##  Welch Two Sample t-test
##
## data:  weight by sex
## t = -4.9918, df = 11.221, p-value = 0.000384
## alternative hypothesis: true difference in means between
## 95 percent confidence interval:
##  -80.03218 -31.13449
## sample estimates:
## mean in group f mean in group m
##        253.5000        309.0833
```

Crow example (cont. 2)

Alternatively, we could model the null hypothesis of no difference by **randomizing** the data. What would these data look like if we had the same eighteen weights but randomly reassigned to the birds?

```r
crows$shuffled_weight <- sample(crows$weight, length(crows$weight))
ggplot(crows, aes(shuffled_weight, sex))+geom_point()
```



Crow example (cont 3.)

Repeat the shuffle many times, and see how our data compare. We'll use the same statistic that the *t*-test used: difference in means between groups.

```r
obs <- mean(crows$weight[crows$sex=="f"])-
       mean(crows$weight[crows$sex=="m"])
num_sim <- 10000
diffs <- numeric(num_sim)
for(i in 1:num_sim){
  sim_weight <- sample(crows$weight, length(crows$weight))
  diffs[i] <- mean(sim_weight[crows$sex=="f"])-
              mean(sim_weight[crows$sex=="m"])
}
mean(abs(obs)<=abs(diffs))
```

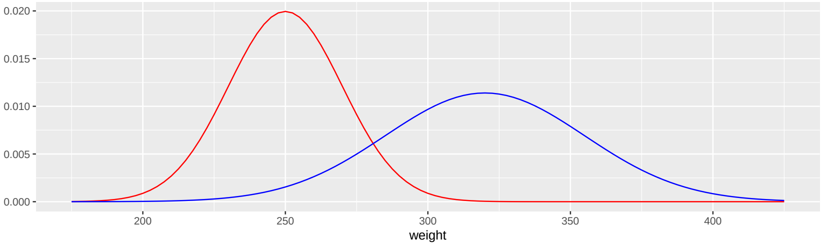
```
## [1] 3e-04
```

We see three examples in 10,000 runs where the observed difference in means is at least as large as the observed difference in absolute value. Note that this matches very closely to the p value of 0.000384 from the *t*-test.

An existing model for crow weights

Randomization allowed us to test our data against the null theory of complete independence of the sex and weight variables.

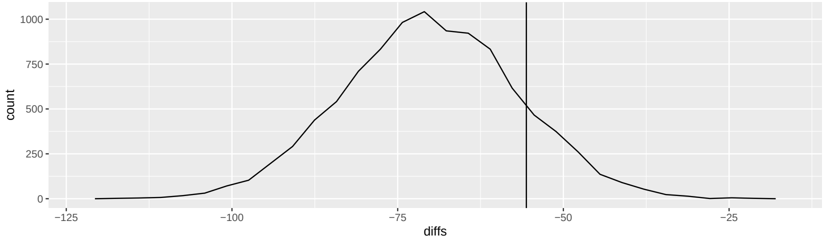
Suppose instead that previous observations lead us to predict that New Caledonian crows will exhibit sexual dimorphism for weight, with female weights distributed as $N(250, 20^2)$ and male weights distributed as $N(320, 35^2)$.



Simulation based on the model

We will **simulate** data based on this model.

```r
diffs <- numeric(num_sim)
for(i in 1:num_sim){
  sim <- data.frame(sex = c(rep("f", 6), rep("m", 12)),
                    weight = c(rnorm(6,250,20), rnorm(12, 320,35)))
  diffs[i] <- mean(sim[sim$sex=="f",]$weight)-
              mean(sim[sim$sex=="m",]$weight)
}
ggplot(data.frame(diffs), aes(diffs))+
  geom_freqpoly()+geom_vline(xintercept = obs)
```



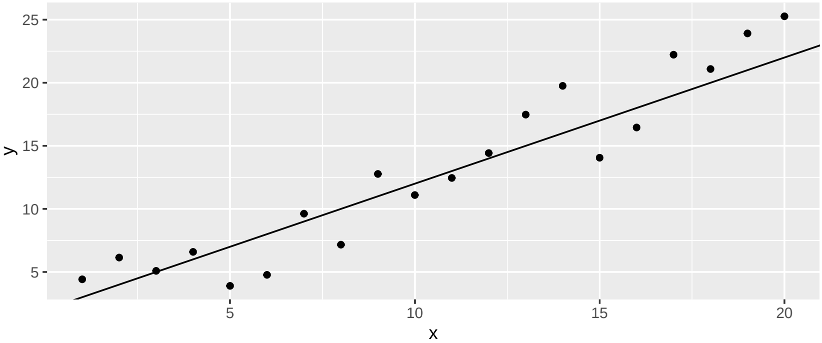
The observed differences in mean weights is not inconsistent with the model.

A simple two variable example

Simulation can be used to create data based on arbitrarily complicated models.

A linear function model with normal errors:

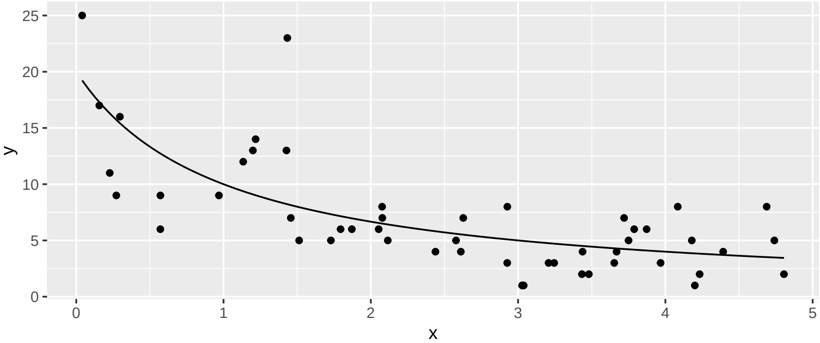
```r
x <- 1:20 # Define inputs
a <- 2; b <- 1 # Set parameters
y_det <- a + b*x # Compute signal part of output
y <- rnorm(20, mean = y_det, sd = 2) # Create noise around the signal
ggplot(data.frame(x=x, y=y), aes(x,y))+
  geom_point() + geom_abline(intercept = a, slope = b)
```



Another example

A hyperbolic function model with negative binomial errors:

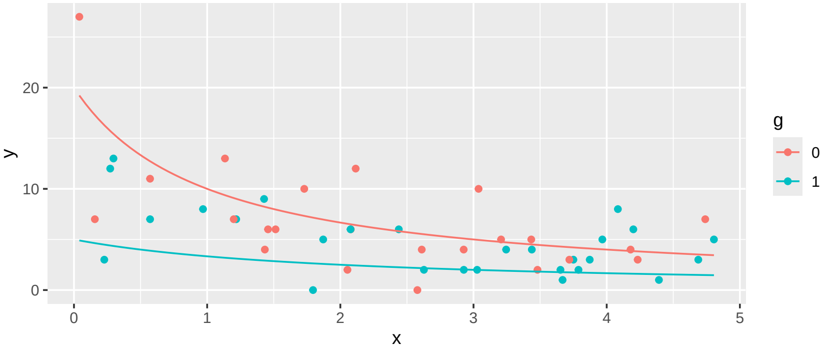
```
x <- runif(50, min = 0, max = 5) # Define inputs
a <- 20; b <- 1; k <- 5 # Set parameters
y_det <- a*b/(b+x) # Compute signal part of output
y <- rnbinom(50, mu = y_det, size = k) # Create noise
ggplot(data.frame(x=x, y=y), aes(x,y))+
  geom_point() + geom_function(fun = function(x) a*b/(b+x) )
```



Adding a categorical predictor

To the previous example, add a grouping factor:

```r
g <- sample(c(0,1), 50, replace = TRUE) # Define new inputs
a <- 20-10*g; b <- 1+g # Redefine parameters to include new input
y_det <- a*b/(b+x) # Compute signal part of output
y <- rnbinom(50, mu = y_det, size = k) # Create noise
ggplot(data.frame(x=x,y=y,g=factor(g)), aes(x,y, color=g))+
  geom_point() +
  geom_function(fun = function(x) (20)/(1+x), aes(color="0")) +
  geom_function(fun = function(x) (10)/(2+x), aes(color="1"))
```



A complicated spatial model

A study of annual weeds in Pacala and Silander (1990), described in Section 5.2.2.2 of Bolker (2008) gives an opportunity to see how complicated models might be built up from simpler components.

The model includes:

- A spatial arrangement of pigweed plants,
- an index of crowding computed from the arrangement,
- an estimate of biomass based on the crowding index, and
- an estimate of seed production based on biomass.

We can model each step of this process and chain them together for the final model.

The spatial arrangement

The spatial arrangement comes from randomly placing parent plants in a rectangular plot, and then generating offspring plants randomly dispersed from the parent plants.

First we set parameters for this process:

```r
L = 30 # consider a 30 by 30 plot
nparents = 50 # and 50 parent plants
offspr_per_parent = 10
noffspr = nparents*offspr_per_parent
dispdist = 2 # mean distance from parent to offspring
```

Then we place the parent plants:

```r
parent_x = runif(nparents,min=0,max=L)
parent_y = runif(nparents,min=0,max=L)
parent_pos <- cbind(parent_x,parent_y)
```

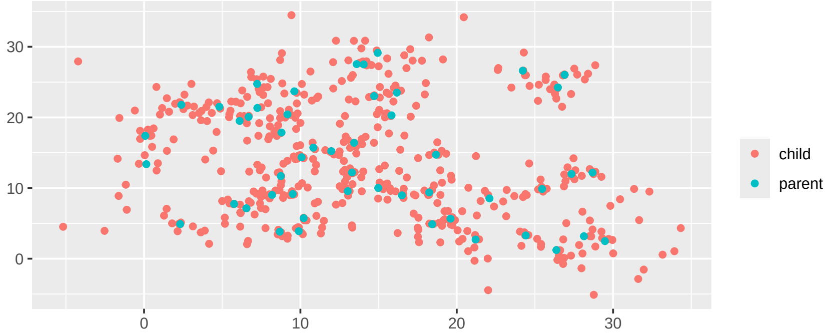
Then the offspring:

```r
angle = runif(noffspr,min=0,max=2*pi)
dist = rexp(noffspr,1/dispdist)
offspr_x = rep(parent_x,each=offspr_per_parent)+cos(angle)*dist
offspr_y = rep(parent_y,each=offspr_per_parent)+sin(angle)*dist
pos <- cbind(offspr_x,offspr_y)
```

The spatial arrangement (cont.)

Here is a view of the resulting arrangement.

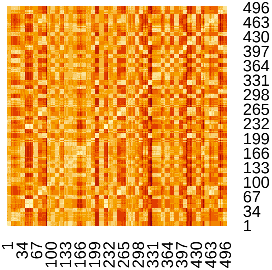
```
ggplot()+
  geom_point(data=as.data.frame(pos),
             aes(offspr_x, offspr_y, color="child"))+
  geom_point(data=as.data.frame(parent_pos),
             aes(parent_x, parent_y, color="parent"))+
  labs(x="", y="", color="")
```



Computing an index of crowding

For the crowding index, first compute distances between plants.

```
ndist <- as.matrix(dist(pos))
heatmap(ndist, Rowv = NA, Colv=NA)
```



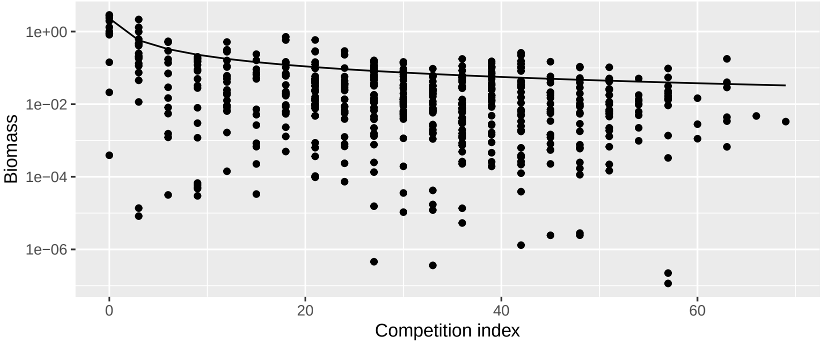
Count the number of plants other the plant being considered within a distance of 2. Call it $N_2(x)$. $CI(x) = 3N_2(x) + 1$ is the crowding index.

```
nbrcrowd = rowSums(ndist<2)-1
ci = nbrcrowd*3
```

Estimating total biomass

The authors of the study determine a model for total biomass. Their estimate is based on the competition index and stochastic noise from a Gamma distribution.

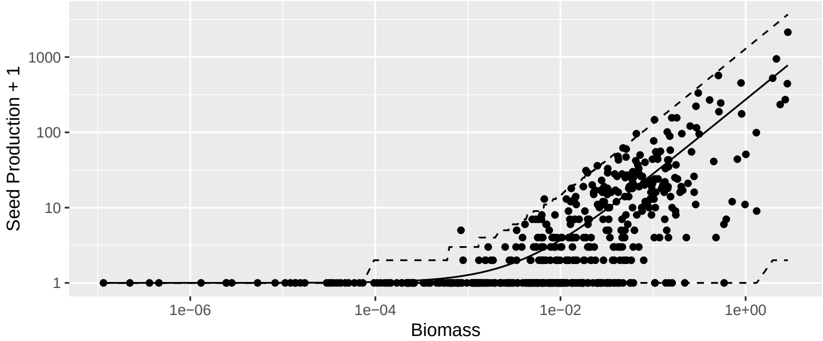
```
M=2.3; alpha=0.49 # parameters from Pacala and Silander
mass_det=M/(1+ci) # deterministic part of the model
mass = rgamma(length(mass_det),scale=mass_det,shape=alpha)
weeds <- data.frame(ci=ci, mass=mass, mass_det=mass_det,
                    x=offspr_x, y=offspr_y)
ggplot(weeds, aes(ci, mass))+geom_point()+geom_line(aes(y=mass_det))+
  scale_y_log10()+labs(x="Competition index", y="Biomass")
```



Estimating seed production

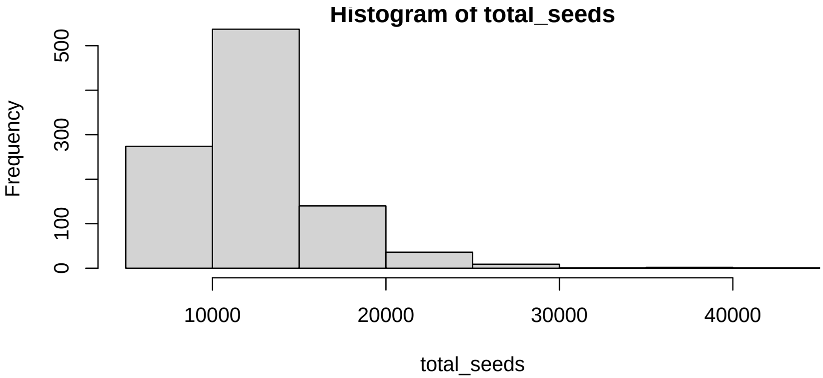
Mean seed production is estimated to be proportional to biomass, with negative binomial noise.

```
b = 271.6; k = 0.569 # parameters from Pacala and Silander
weeds$seed_det <- b*mass # the deterministic part of the model
weeds$seed <- rnbinom(length(weeds$seed_det),mu=weeds$seed_det,size=k)
ggplot(weeds,aes(mass,seed+1))+geom_point()+geom_line(aes(y=seed_det+1))+
  geom_line(aes(y=qnbinom(0.025,mu=seed_det,size=k)+1), linetype=2)+
  geom_line(aes(y=qnbinom(0.975,mu=seed_det,size=k)+1), linetype=2)+
  scale_y_log10()+scale_x_log10()+
  labs(x="Biomass",y="Seed Production + 1")
```



Simulated distribution of total seed production

```
num_sim <- 1000; total_seeds <- numeric(num_sim)
for(i in 1:num_sim){
parent_x = runif(nparents,min=0,max=L); parent_y = runif(nparents,min=0,max=L)
angle = runif(noffspr,min=0,max=2*pi); dist = rexp(noffspr,1/dispdist)
offspr_x = rep(parent_x,each=offspr_per_parent)+cos(angle)*dist
offspr_y = rep(parent_y,each=offspr_per_parent)+sin(angle)*dist
pos <- cbind(offspr_x,offspr_y); ndist <- as.matrix(dist(pos))
nbrcrowd = rowSums(ndist<2)-1; ci = nbrcrowd*3; mass_det=M/(1+ci)
mass = rgamma(length(mass_det),scale=mass_det,shape=alpha)
seed_det <- b*mass ; seed <- rnbinom(length(seed_det),mu=seed_det,size=k)
total_seeds[i] <- sum(seed)
}
hist(total_seeds)
```



Hypothesis testing with a simulated distribution

We saw in the crow example how to do hypothesis testing with a simulated distribution:

- Given an observed value of a statistic, calculate the fraction of the simulated values that are at least as extreme as the observed value. This is the p value.

Suppose that in a 30 × 30 plot with 50 parent plants, each having 10 offspring, you determine that there is a seed set of 23000 seeds. Test the hypothesis that this seed set is consistent with the model.

```r
mean(total_seeds>=23000)
```

```
## [1] 0.023
```

So this data point would be a somewhat unlikely output from the model.

Confidence intervals from a simulated distribution

Computing confidence intervals from a simulated distribution is straightforward and can be done using the `quantile` function.

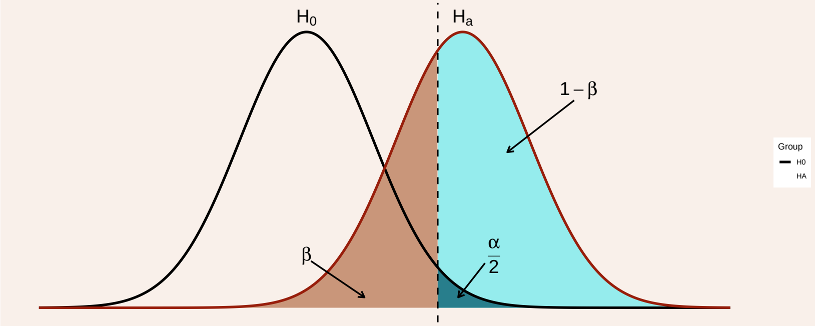
```
quantile(total_seeds, c(0.025, 0.975))
```

```
##     2.5%    97.5%
##  7344.65 22333.55
```

Statistical power

In a hypothesis test based on an assumed distribution of a statistic, there are four important quantities, and and three of them determine the fourth:

- Sample size (n) or other model/ experimental parameters
- Effect size ($\Delta = \mu_a - \mu_0$)
- Significance level (α)
- Power (1-β)


Adapted from http://rpsychologist.com (CCA 4.0)

Power for traditional tests

Power computers for many standard tests are available in R.

- Suppose male and female crow weights are both approximately normally distributed with a pooled standard deviation of 27. How many crows of each sex would we need to weigh if we wanted to have a 90% probability of correctly noting a difference in mean weights of 70?

```
power.t.test(delta=70, sd=27, power=0.9)
```

```
##
##      Two-sample t test power calculation
##
##               n = 4.353678
##           delta = 70
##              sd = 27
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Also see the `pwr` package.

Power computations using simulation

In a situation where you aren't using a standard statistical test, for example if you model your data with `glm` instead of `lm`, you can use simulation to compute power.

This is what we'll do in the lab.

References

Bolker, Benjamin M. 2008. *Ecological Models and Data in R*. Princeton University Press.

Kenward, Benjamin, Christian Rutz, Alex A. S. Weir, Jackie Chappell, and Alex Kacelnik. 2004. "Morphology and Sexual Dimorphism of the New Caledonian Crow Corvus Moneduloides, with Notes on Its Behaviour and Ecology." *Ibis* 146 (4): 652–60. https://doi.org/https://doi.org/10.1111/j.1474-919x.2004.00299.x.

Pacala, Stephen W., and J. A. Silander. 1990. "Field Tests of Neighborhood Population Dynamic Models of Two Annual Weed Species." *Ecological Monographs* 60 (1): 113–34. http://www.jstor.org/stable/1943028.