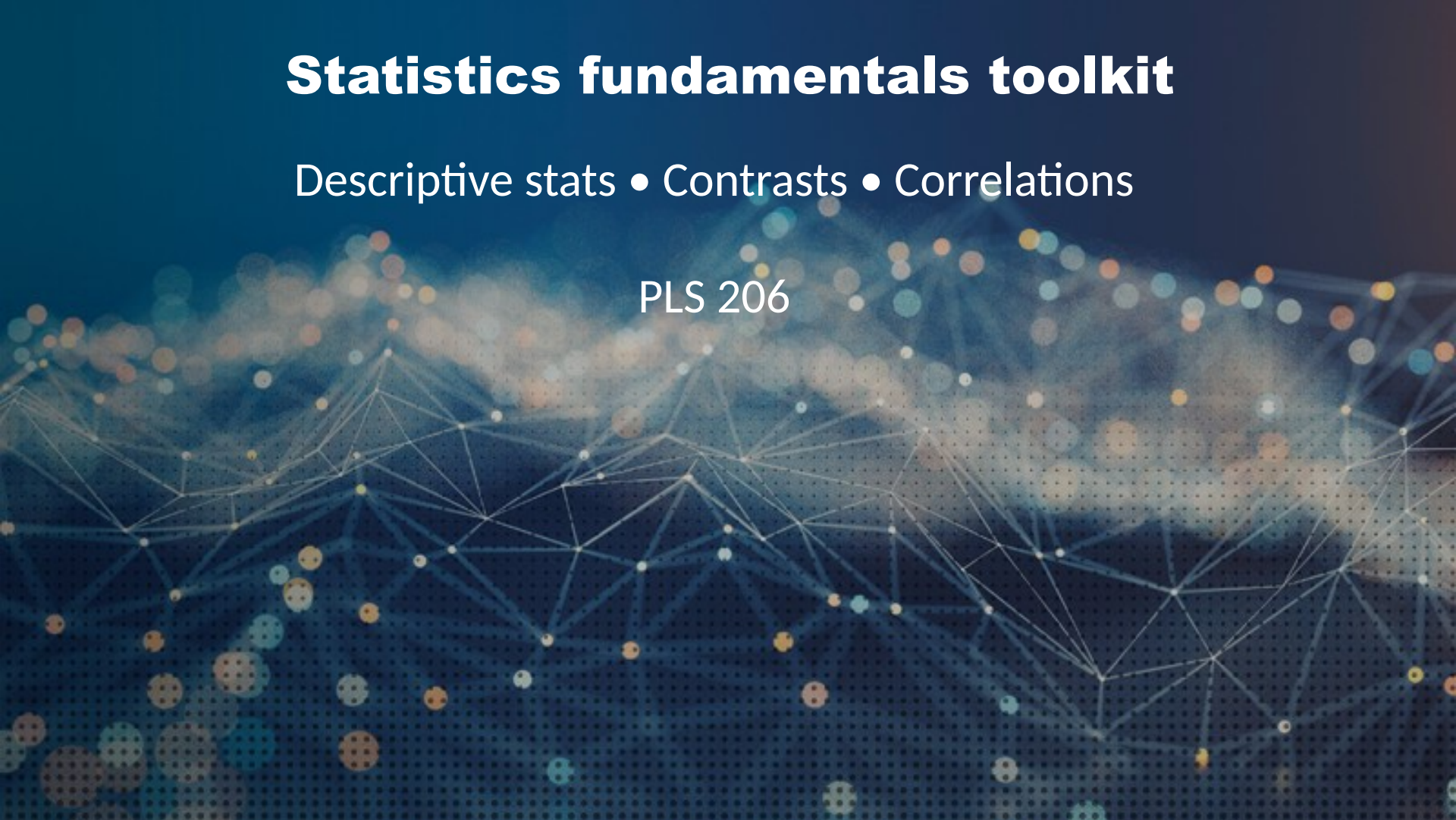
Statistics fundamentals toolkit

Descriptive stats • Contrasts • Correlations

PLS 206



Statistics can't save...

- **a bad design**: no controls, no replication, or pseudoreplication (10 plants in one pot are one replicate, not ten)
- **a biased sample**: measuring only the plants at the easy-to-reach field edge
- **confounding**: the fertilized plots were also the irrigated ones
- **bad logic**: correlation isn't causation; no evidence of an effect isn't evidence of no effect; testing until something is significant
- **a false hypothesis**: no p-value makes a wrong idea right

> “To consult the statistician after an experiment is finished is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of.”

R. A. Fisher, Presidential Address to the first Indian Statistical Congress, 1938

What statistics is for

- **Quantifying uncertainty**: how sure are we? (standard errors, confidence intervals)
- **Separating signal from noise**: is a difference bigger than chance alone would give? (tests)
- **Estimating effects**: how big, in real units (effect sizes)
- **Planning**: how many plants do we need? (power)
- **Seeing structure** in many variables at once (PCA, clustering)
- **Predicting**, with an honest measure of the error (cross-validation)

Good statistics: honest uncertainty about a well-designed question.

Goals for Today

- Distributions: normal, gamma, lognormal, Poisson, binomial
- Checking normality and transforming data
- Uncertainty: SD, SE, confidence intervals
- Test statistics, p-values, errors, and power
- Group comparisons (t-tests, ANOVA; non-parametric)
- Covariance, correlation, and categorical tests
- Effect sizes and multiple testing

Pop quiz!

Kahoot-style warm-up

Warm-up: 17 quick questions from your undergrad stats course.

Not graded.

Use pseudonym if you want.

Fast and correct wins.

Scan with your phone

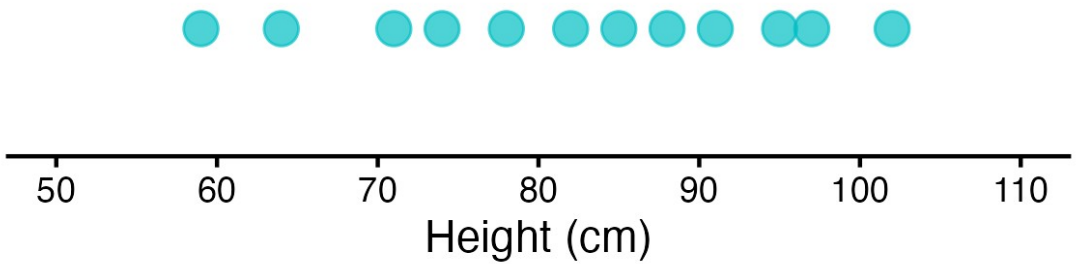
Join on your phone or laptop: greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w1-warmup

host screen

Part 1: Describing data

Means, distributions, and spread

$$$$$$$$$$ make a bet $$$$$$$$$

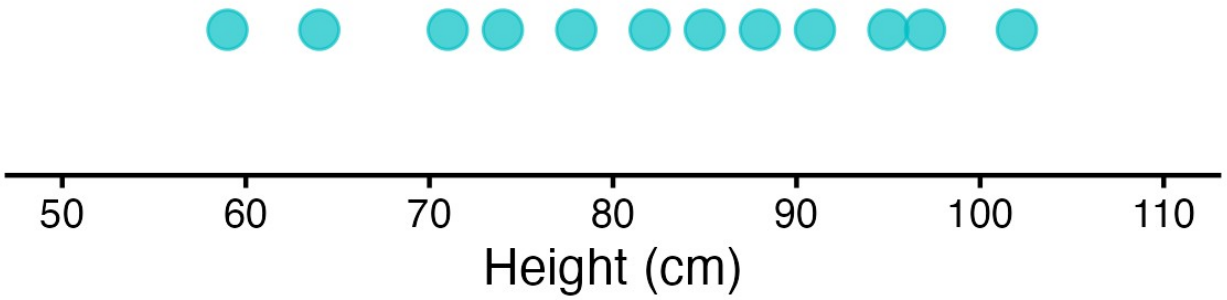


12 wheat plants, measured in cm

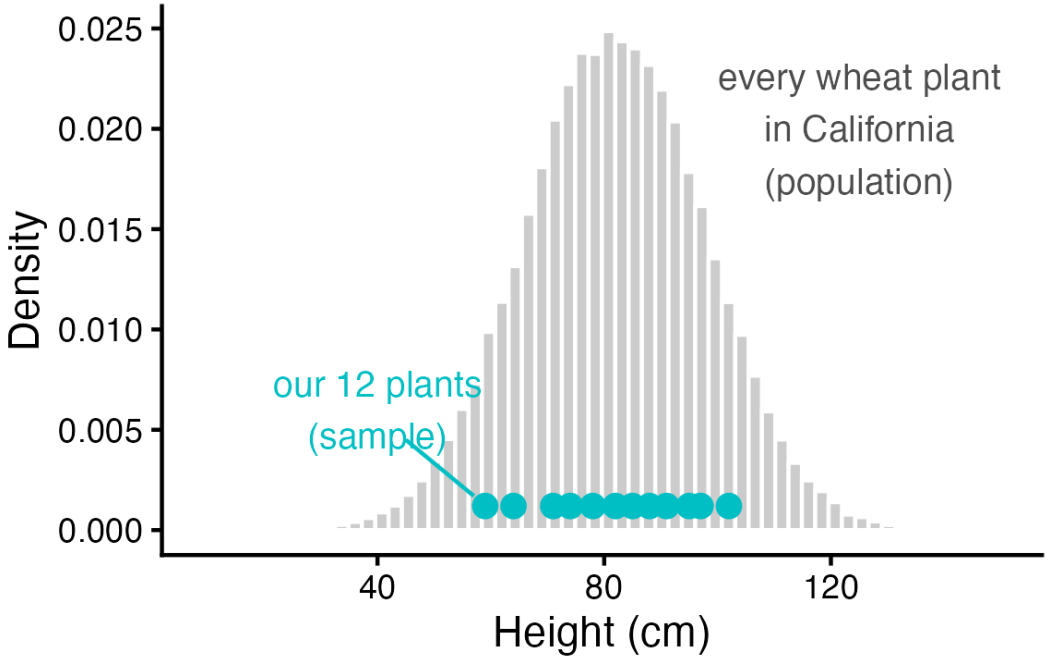
- Guess what the 13th plant height will be. Closest bet wins
- How do you decide what to bet?
- Statistics makes that bet precise …
- … and tells us **how sure** we should be (i.e how much to wager)

12 wheat plants

```
> height <- c(78, 95, 64, 88, 102, 71, 85, 91, 59, 97, 82, 74)
> height
 [1]  78  95  64  88 102  71  85  91  59  97  82  74
> length(height)
[1] 12
```



Population vs. sample



	Population	Sample
mean	μ	$\bar{x}$
SD	σ	s
size	all plants	$n = 12$

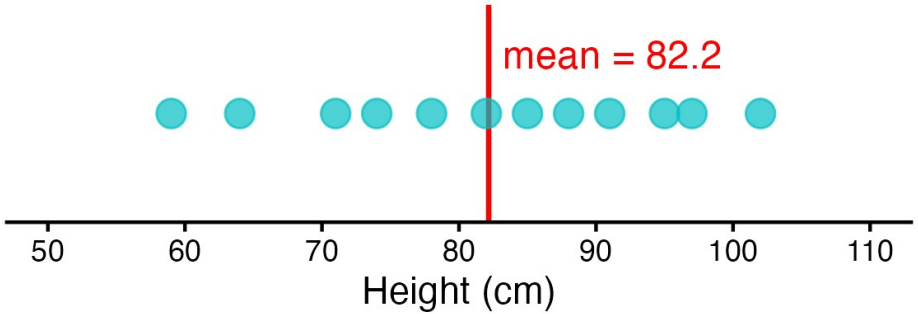
We use the **sample** to estimate the **population**.

The mean

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\bar{x} = \frac{78 + 95 + 64 + \cdots + 74}{12} = \frac{986}{12} = 82.2 \text{ cm}$$

```
> sum(height)
[1] 986
> sum(height) / length(height)
[1] 82.16667
> mean(height)
[1] 82.16667
```



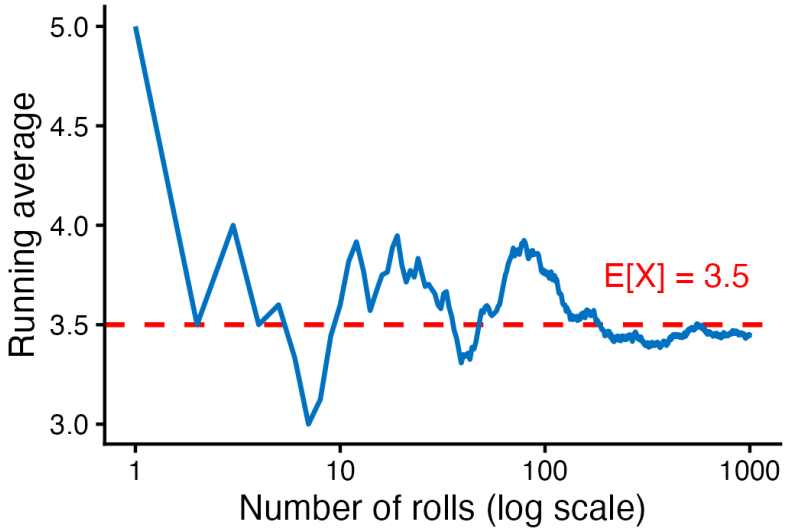
Best single bet for a new plant: **82.2 cm**

Expected value: the long-run average

$$E[X] = \sum_x x \cdot P(X = x)$$

$$E[\text{die}] = 1 \cdot \tfrac{1}{6} + 2 \cdot \tfrac{1}{6} + \cdots + 6 \cdot \tfrac{1}{6} = 3.5$$

```
> sum(1:6 * 1/6)                    # E[X] for one die
[1] 3.5
> set.seed(206)
> rolls <- sample(1:6, 1000, replace = TRUE)
> mean(rolls[1:10]); mean(rolls)
[1] 3.6
[1] 3.447
```



- You never roll a 3.5, but that's what the rolls average out to
- A bet's **expected value** = what it pays per play, on average

The mean is an expected value

$$\bar{x} = \sum_{i=1}^{n} x_i \cdot \frac{1}{n} = 78 \cdot \tfrac{1}{12} + 95 \cdot \tfrac{1}{12} + \cdots + 74 \cdot \tfrac{1}{12} = 82.2$$

Pick one of our 12 plants at random: each has probability **1/12**. The mean is that bet's expected value.

```
> height <- c(78, 95, 64, 88, 102, 71, 85, 91, 59, 97, 82, 74)
> sum(height * 1/12)   # each plant equally likely
[1] 82.16667
> mean(height)
[1] 82.16667
```

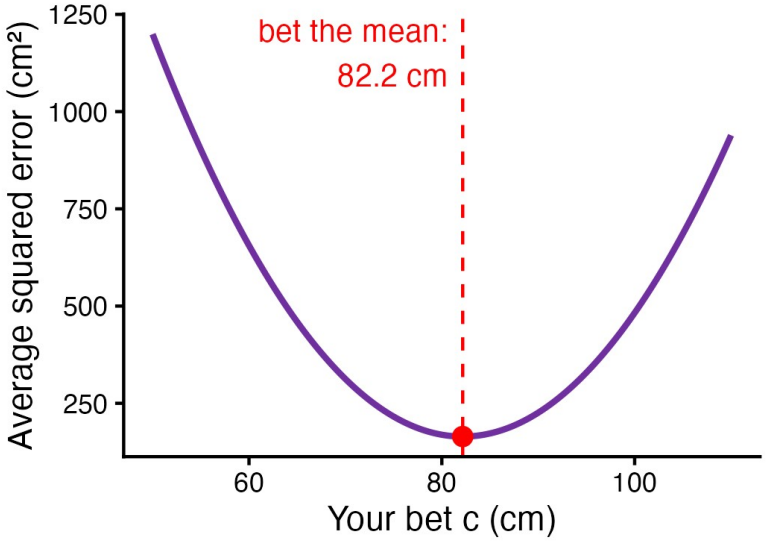
Discrete	$E[X] = \sum_x x \cdot P(X = x)$
Continuous	$E[X] = \int x\, f(x)\, dx$
Population	$E[X] = \mu$ (normal: the μ in $N(\mu, \sigma)$)
Sample	$\bar{x}$ estimates $E[X]$

- **E[X]**: a property of the population (**μ**)
- **x̄**: our estimate of it from a sample

Why the mean is the best bet

$$E\left[(X-c)^2\right] \text{ is smallest when } c = E[X]$$

```
> bets <- 50:110
> sq_err <- sapply(bets, function(c) mean((height - c)^2))
> bets[which.min(sq_err)]     # best whole-cm bet
[1] 82
> optimize(function(c) mean((height - c)^2), c(50, 110))$minimum
[1] 82.16667
```



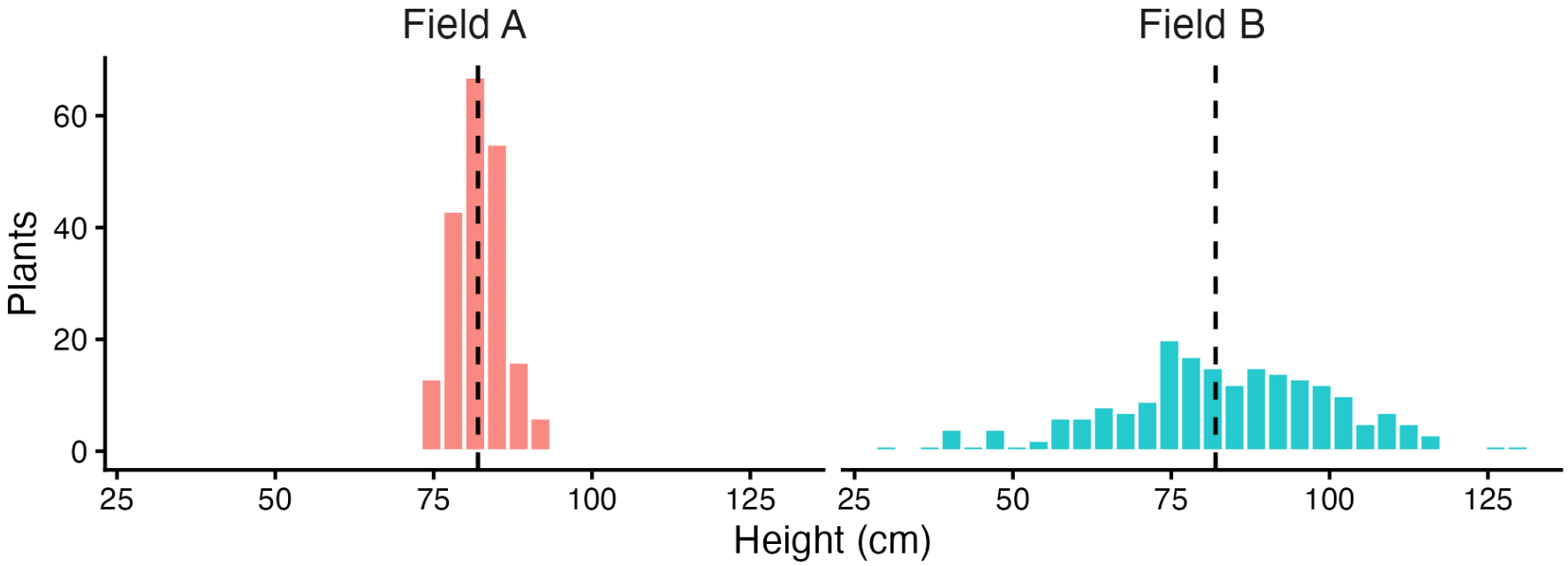
- Score each bet by its squared error: the bet with the lowest expected penalty is the **mean**
- Score by absolute error instead: the best bet is the **median** (more on squared vs. absolute later)

Mean vs. median

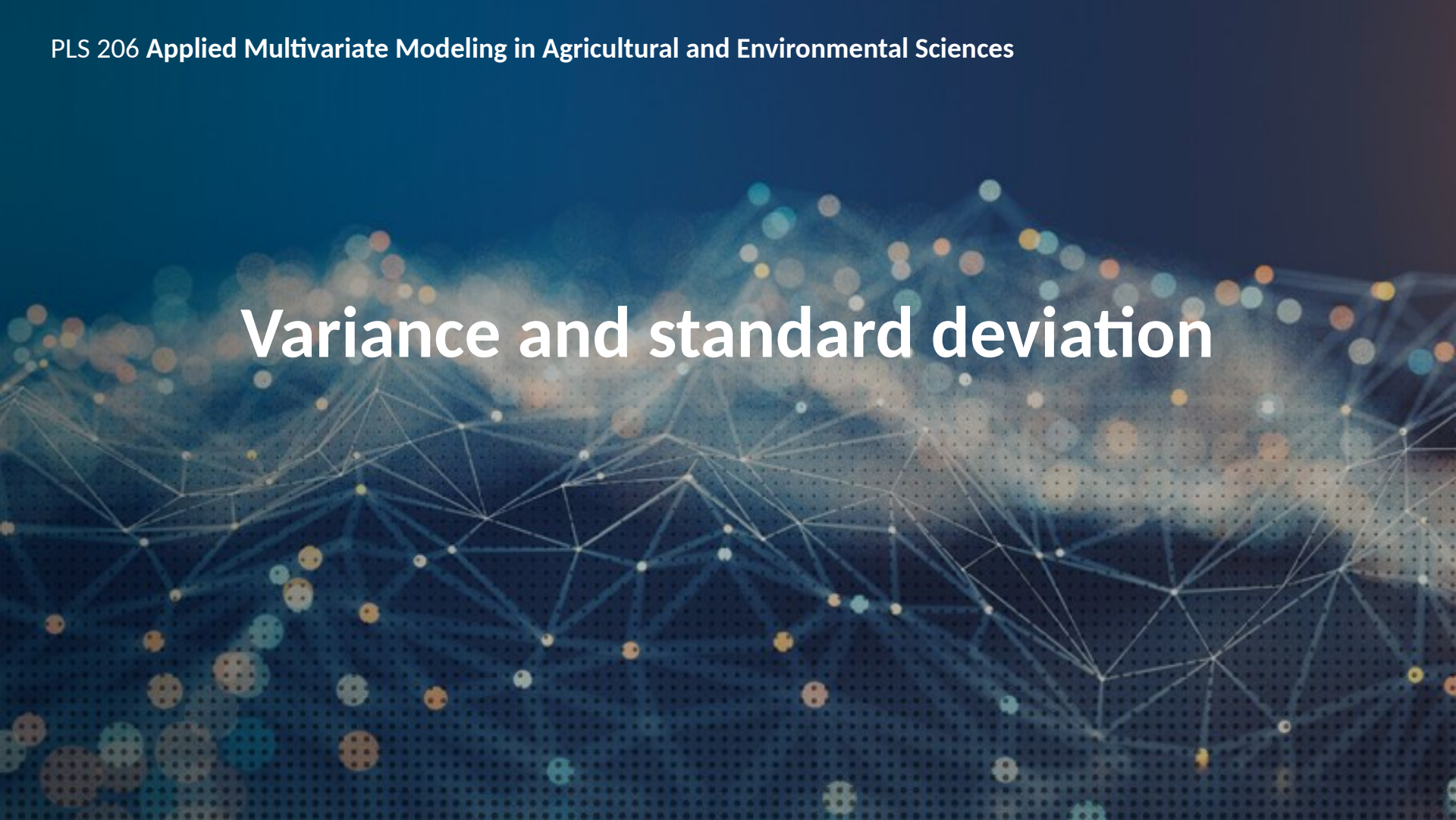
```
> median(height)
[1] 83.5
> # oops: someone typed 820 instead of 82
> typo <- c(height, 820)
> mean(typo)
[1] 138.9231
> median(typo)
[1] 85
```

- **Median** = middle value (50% above, 50% below)
- One typo moves the mean from 82 to **139**; the median barely moves
- The median is robust to outliers; the mean is not

Same mean, different spread

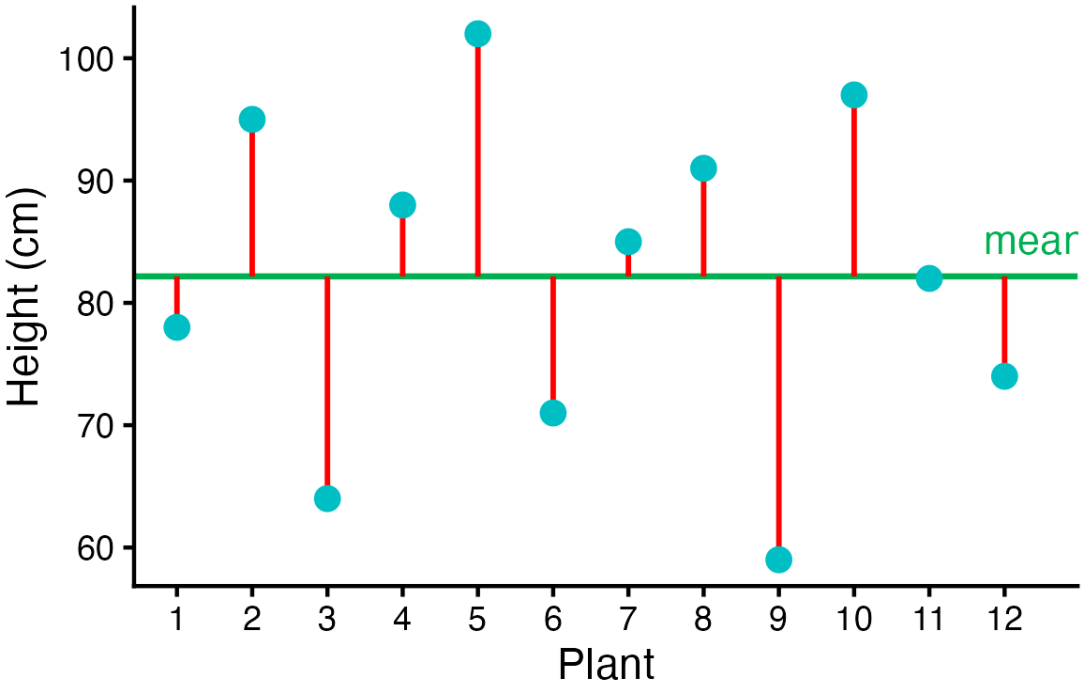


$\bar{x}$ = 82 cm in both fields. How do we put a number on spread?



Variance and standard deviation

Deviations from the mean



$$d_i = x_i - \bar{x}$$

Each **red line** is one plant's deviation: how far it is from the **mean**

Deviations sum to zero

```
> deviations <- height - mean(height)
> round(deviations, 1)
 [1]  -4.2  12.8 -18.2   5.8  19.8 -11.2   2.8   8.8 -23.2  14.8
[11]  -0.2  -8.2
> sum(deviations)
[1] -5.684342e-14
```

$$\sum_{i=1}^{n} (x_i - \bar{x}) = 0 \quad \text{always}$$

So we can't just average them: the positives and negatives cancel.

Absolute vs. squared

$$\frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$$

mean absolute deviation

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

mean squared deviation

```
> mean(abs(deviations))    # mean |deviation|
[1] 10.83333
> mean(deviations^2)       # mean deviation^2
[1] 164.4722
```

Squares win: they are smooth (differentiable) and **add up**. Sums of squares are the backbone of ANOVA, regression, and PCA.

Variance, step by step

```
> tab <- data.frame(height,
+   dev  = round(deviations, 1),
+   dev2 = round(deviations^2, 1))
> tab
   height   dev  dev2
1      78  -4.2  17.4
2      95  12.8 164.7
3      64 -18.2 330.0
4      88   5.8  34.0
5     102  19.8 393.4
6      71 -11.2 124.7
7      85   2.8   8.0
8      91   8.8  78.0
9      59 -23.2 536.7
10     97  14.8 220.0
11     82  -0.2   0.0
12     74  -8.2  66.7
> sum(deviations^2)
[1] 1973.667
```

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

$$s^2 = \frac{1973.7}{12-1} = 179.4 \text{ cm}^2$$

```
> n <- length(height)
> sum(deviations^2) / (n - 1)
[1] 179.4242
> var(height)
[1] 179.4242
```

Standard deviation

$$s = \sqrt{s^2}$$

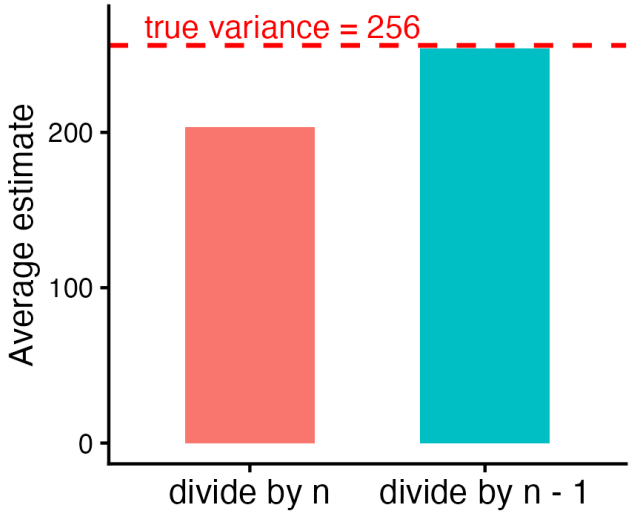
$$s = \sqrt{179.4} = 13.4 \text{ cm}$$

```
> sqrt(var(height))
[1] 13.39493
> sd(height)
[1] 13.39493
```

- **Variance:** cm^2 SD: back in cm
- SD = typical distance of a plant from the mean

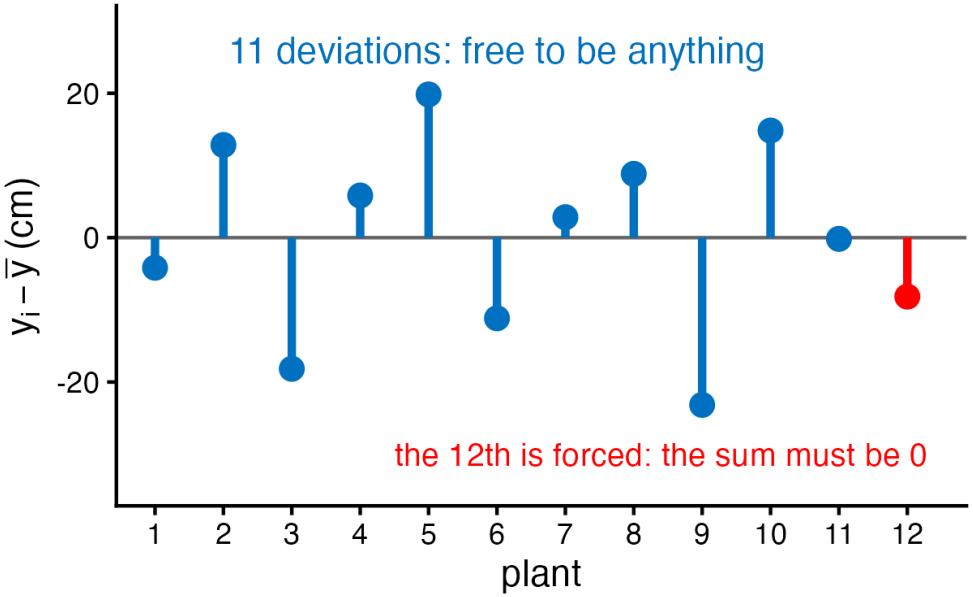
Why n − 1?

```
> set.seed(206)
> sims <- replicate(20000, {
+   s <- rnorm(5, mean = 82, sd = 16)  # 5 plants
+   d <- s - mean(s)
+   c(over_n = sum(d^2) / 5, over_n1 = sum(d^2) / 4)
+ })
> rowMeans(sims)   # truth: 16^2 = 256
  over_n  over_n1
203.2391 254.0488
```



- $\bar{x}$ is fit from the same data, so deviations from $\bar{x}$ are a bit **too small**
- Dividing by n underestimates the variance, worst at small n

Degrees of freedom: the intuition



Degrees of freedom = how many values are still free to vary once you've estimated something from the data

- 12 plants = 12 pieces of information
- Estimating $\bar{x}$ **spends one**
- 11 are left to estimate the spread

```
> dev <- height - mean(height)
> round(sum(dev), 10)     # the constraint: always 0
[1] 0
> -sum(dev[1:11])         # so 11 deviations fix the 12th
[1] -8.166667
> dev[12]
[1] -8.166667
> length(height) - 1      # df = 11
[1] 11
```

That's the n − 1 in the variance.

Degrees of freedom: the definition

$$df = n - (\text{number of quantities estimated from the data})$$

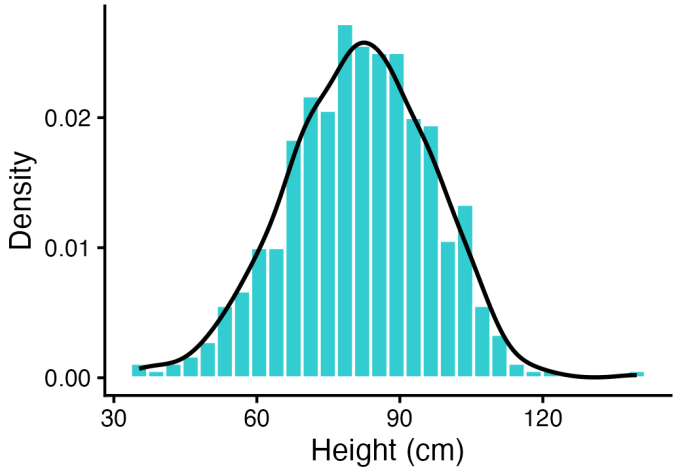
$$\sum_{i=1}^{n} (y_i - \bar{y}) = 0 \quad \Rightarrow \quad n - 1 \text{ free deviations}$$

Where it shows up	Estimated first	df	Ours
variance, SD, one-sample t	$\bar{y}$	$n - 1$	11
two-sample t (equal variances)	$\bar{y}_1,\ \bar{y}_2$	$n_1 + n_2 - 2$	28
test of a correlation r	a line: intercept, slope	$n - 2$	58
ANOVA within (residuals)	k group means	$N - k$	27
ANOVA between (groups)	the grand mean	$k - 1$	2
regression (next week)	intercept + p slopes	$n - p - 1$	

Fewer df → s is less certain → fatter-tailed t → wider CIs and larger p-values

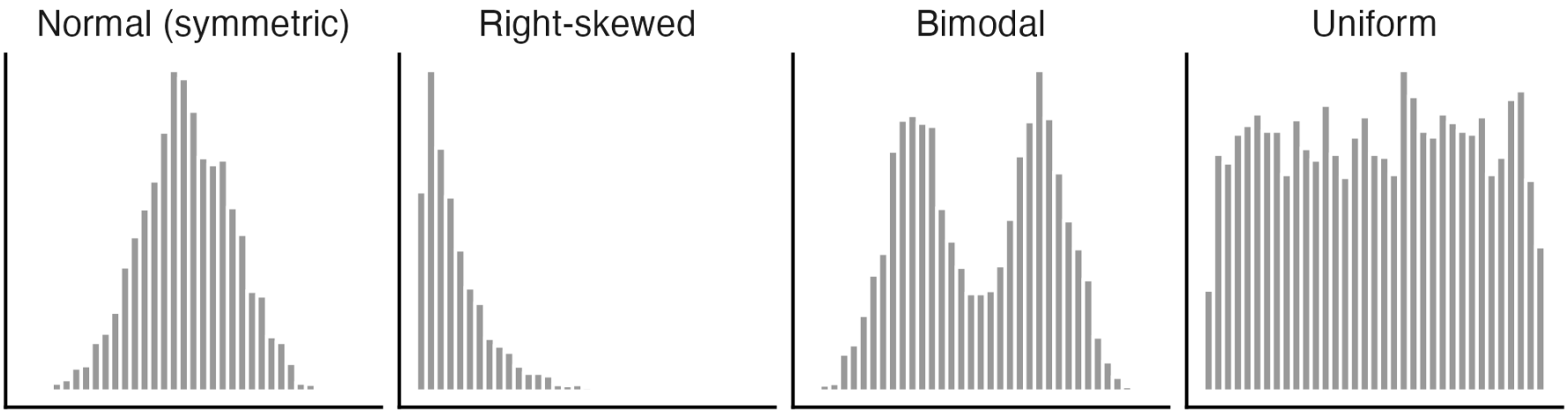
Distributions

```
> set.seed(206)
> # simulate a field of 500 plants
> field <- rnorm(500, mean = 82, sd = 16)
> head(round(field, 1))
[1] 99.3 98.3 52.8 79.4 77.4 58.6
> summary(field)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  35.30   71.05   81.55   81.48   92.36  139.70
```



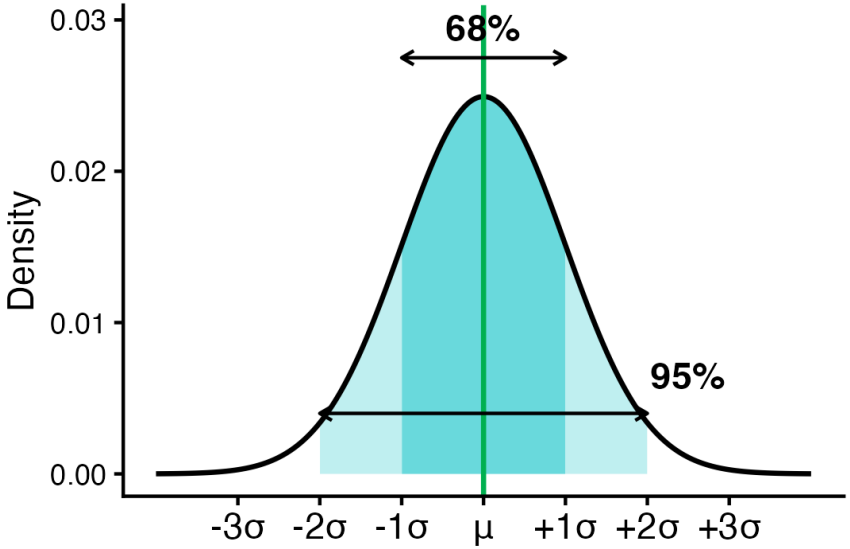
A distribution describes **how often** each value occurs: center, spread, and shape.

Shapes of distributions



- Always look at the shape before choosing a statistic or a test
- Skew and multiple peaks make the mean a **misleading** summary
- in R: **hist()** function!

The normal distribution



$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Two numbers describe it: **μ** (center) and **σ** (spread)
- 68% within 1 SD, 95% within ~2 SD
- Many tests assume it

Heights by hand (μ = 82 cm, σ = 16 cm):

$$x = 82\ (\mu): \quad f(82) = \frac{1}{16\sqrt{2\pi}} e^{0} = 0.025$$

$$x = 98\ (+1\sigma): \quad f(98) = 0.025 \times e^{-\frac{(98-82)^2}{2 \cdot 16^2}} = 0.025 \times e^{-0.5} = 0.015$$

$$x = 114\ (+2\sigma): \quad f(114) = 0.025 \times e^{-2} = 0.003$$

dnorm() does the same:

```
> round(dnorm(c(82, 98, 114), 82, 16), 3)
[1] 0.025 0.015 0.003
```

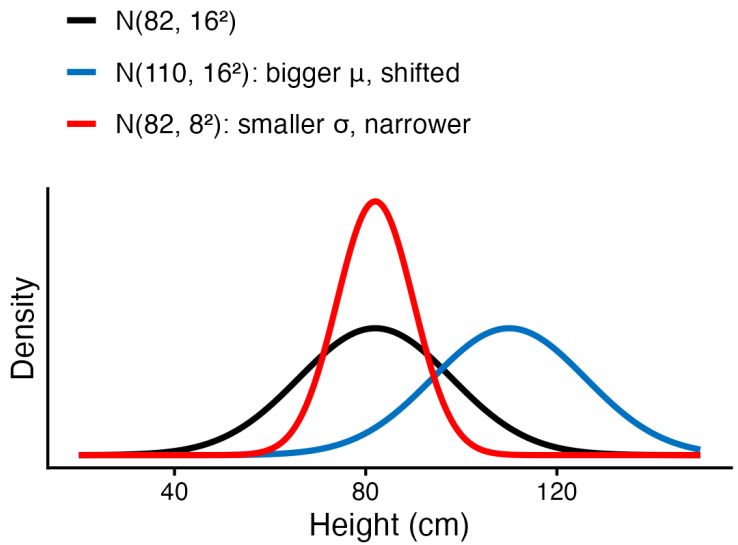
Reading the notation

$$\underbrace{X}_{\text{the variable}} \quad \underbrace{\sim}_{\text{is distributed as}} \quad \underbrace{N}_{\text{normal}} \left(\underbrace{\mu}_{\text{mean}}, \underbrace{\sigma^2}_{\text{variance}} \right)$$

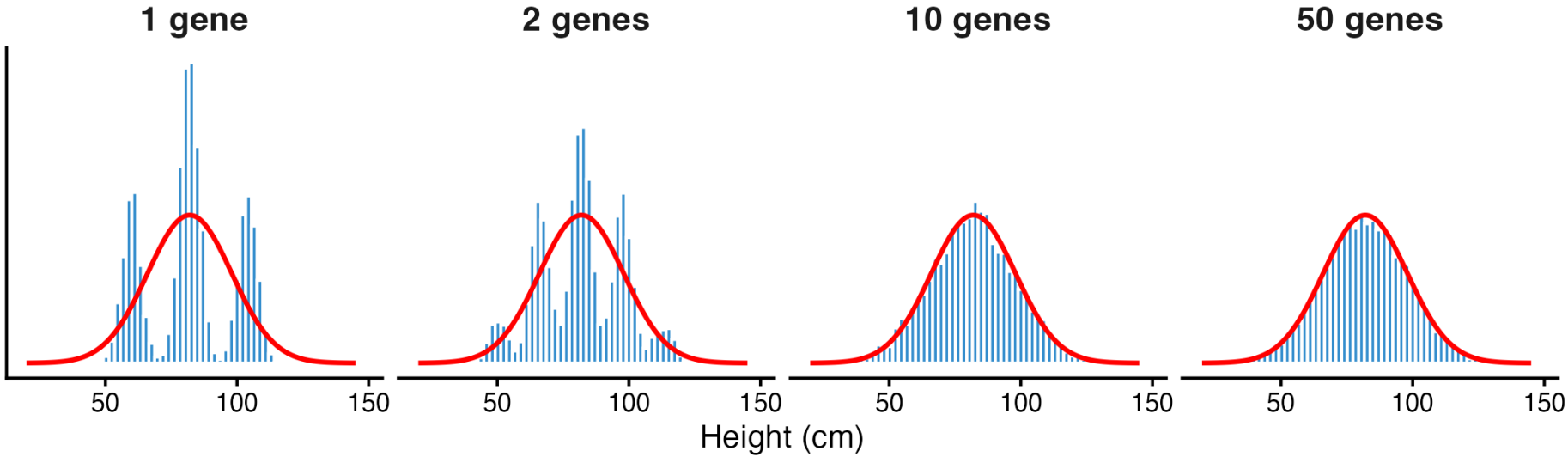
$$\text{height} \sim N(82,\ 16^2)$$

"Height is normally distributed with **mean 82 cm** and **SD 16 cm** (variance 256 cm²)"

```
> set.seed(206)
> x <- rnorm(5000, mean = 82, sd = 16)  # sd, not variance!
> c(mean = mean(x), sd = sd(x), var = var(x))
     mean        sd       var
 81.86517  15.83895 250.87243
```



Normal: many small effects add up



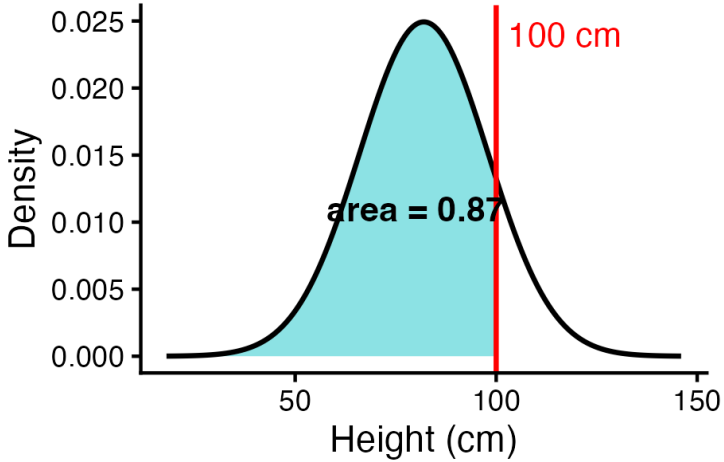

```
> set.seed(1)
> loci <- 50                    # genes for height
> geno <- matrix(rbinom(5000 * loci, 2, 0.5), ncol = loci)
> dose <- rowSums(geno) - loci  # tall alleles vs. average
> height <- 82 + 3.12 * dose +
+           rnorm(5000, sd = 3.6)    # environment
> c(mean(height), sd(height))
[1] 81.88343 16.04766
```

- Many small, independent effects that **add up**
- Height, yield, leaf length; measurement error

Distributions in R: d p q r

```
> dnorm(82, mean = 82, sd = 16)      # curve height
[1] 0.02493389
> pnorm(100, mean = 82, sd = 16)     # P(< 100 cm)
[1] 0.8697055
> 1 - pnorm(100, mean = 82, sd = 16) # P(> 100 cm)
[1] 0.1302945
> qnorm(0.975, mean = 82, sd = 16)   # 97.5% shorter
[1] 113.3594
> rnorm(5, mean = 82, sd = 16)       # 5 plants
[1] 83.86110 69.13416 78.78999 87.42395 87.41154
```

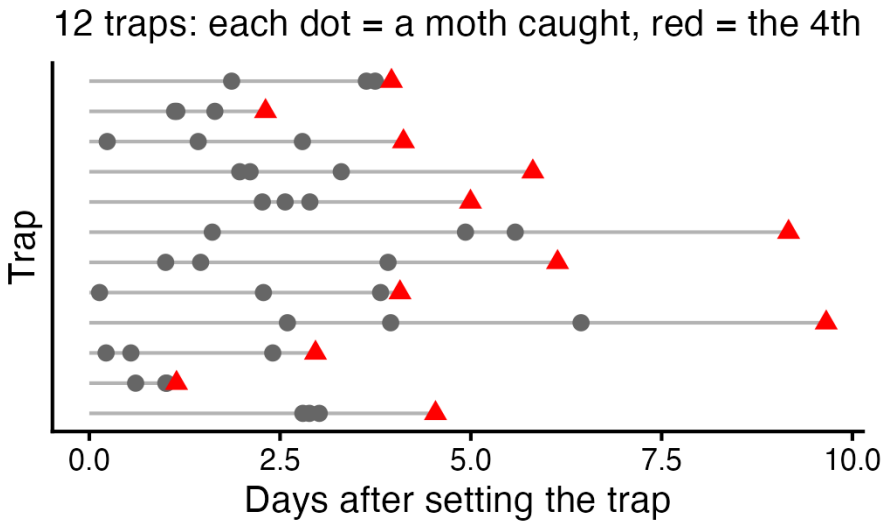
dnorm density (height of the curve)
pnorm probability below a value (area)
qnorm quantile: value for a given probability
rnorm random draws

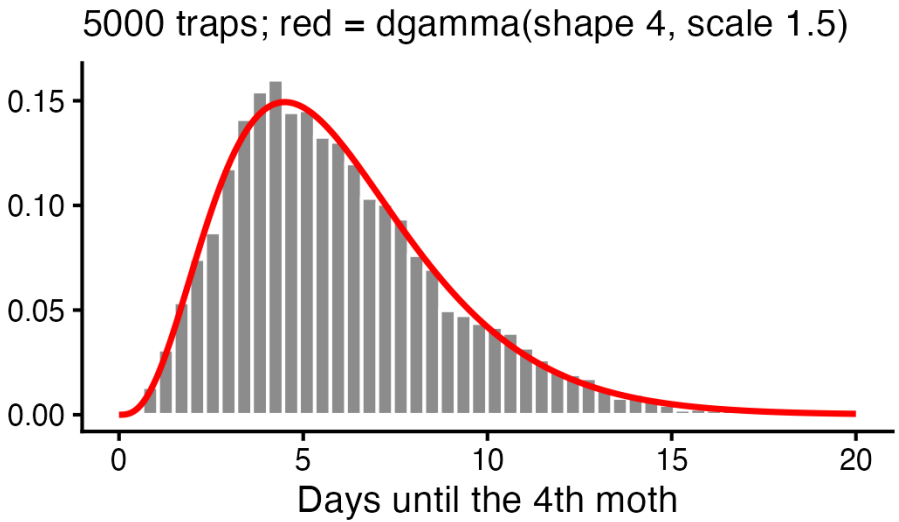


Beyond the normal distribution

Gamma, lognormal, Poisson, binomial

Waiting for 4 moths in a trap






```
> set.seed(2)
> # each moth: a random wait, mean 1.5 days
> waits <- matrix(rexp(5000 * 4, rate = 1/1.5), ncol = 4)
> days  <- rowSums(waits)   # days until the 4th moth
> c(mean(days), var(days))  # gamma(4, 1.5): 6 and 9
[1] 6.011149 9.086912
```

- Each moth arrives after a **random wait** (1.5 days on average)
- Days until the 4th = **sum of 4 waits** → gamma

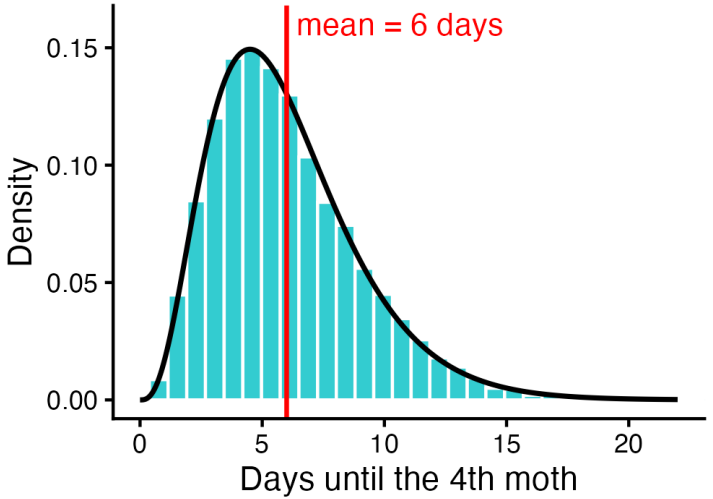
Gamma distribution

Positive, right-skewed: waiting times, rainfall amounts, days to flowering

$$\mu = k\,\theta \qquad \sigma^2 = k\,\theta^2$$

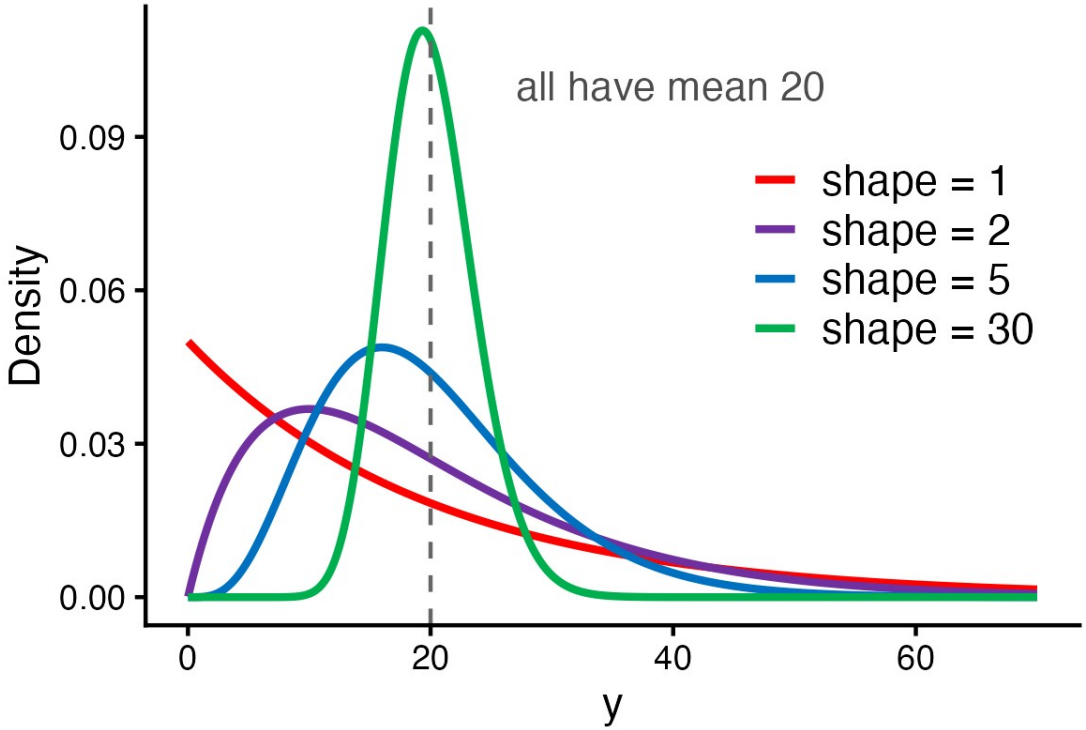
k = shape: how many events (4 moths)
θ = scale: average wait per event (1.5 days)

```
> # days until the trap has caught 4 moths
> set.seed(1)
> days <- rgamma(10000, shape = 4, scale = 1.5)
> mean(days)    # shape * scale   = 6
[1] 6.005883
> var(days)     # shape * scale^2 = 9
[1] 8.97683
> pgamma(10, shape = 4, scale = 1.5)   # P(4 moths by day 10)
[1] 0.8991163
```



Shape 4 × scale 1.5 = **mean 6 days**. Same d/p/q/r pattern: **dgamma, pgamma, qgamma, rgamma**

Shape controls the skew



- **shape = 1:** exponential (waiting for one event)
- Small shape: strongly right-skewed
- **Large shape:** nearly normal
- **rgamma(5000, shape = 30)** comes back on the normality-test slide

Live poll

Multiple choice

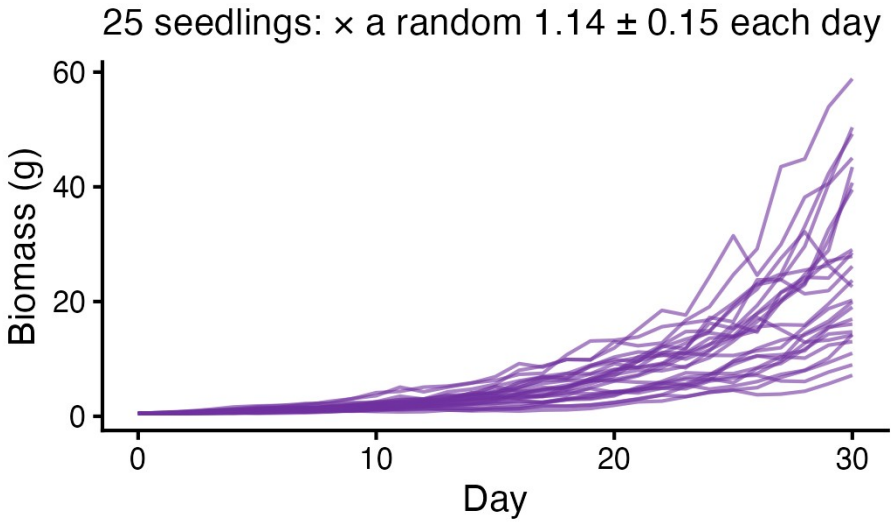
Shoot biomass has a long right tail. Which is bigger: the mean or the median?

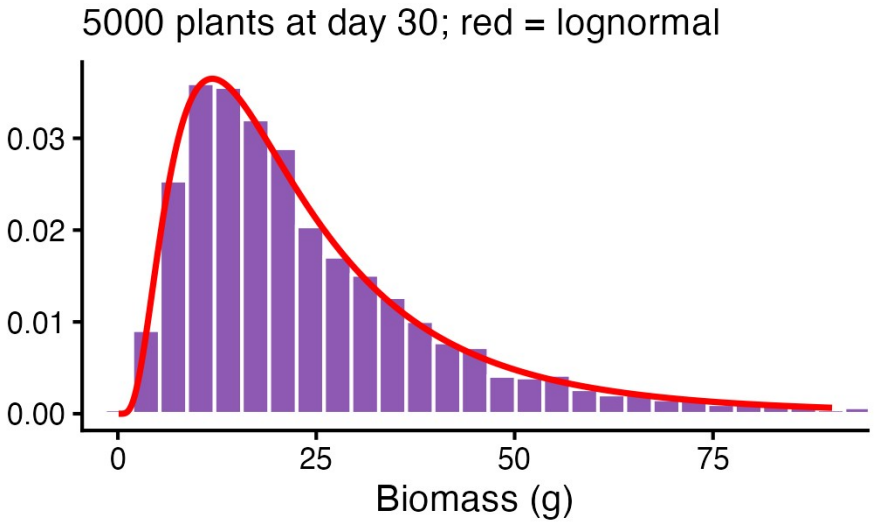
Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-skew

host screen

Seedlings that grow by a percentage each day





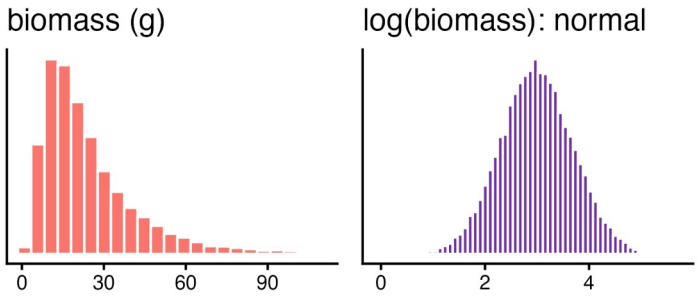

```
> set.seed(3)
> growth <- matrix(rnorm(5000 * 30, 1.14, 0.145), ncol = 30)
> biomass <- 0.5 * apply(growth, 1, prod)  # 0.5 g x 30 days
> c(median(biomass), mean(biomass))
[1] 19.78098 25.46055
```

- Many random factors that **multiply**: logs turn products into sums
- Biomass, population size, concentrations

Lognormal distribution

$$\log(y) \sim N(\mu, \sigma^2) \qquad \text{median} = e^{\mu} \qquad \text{mean} = e^{\mu+\sigma^2/2}$$

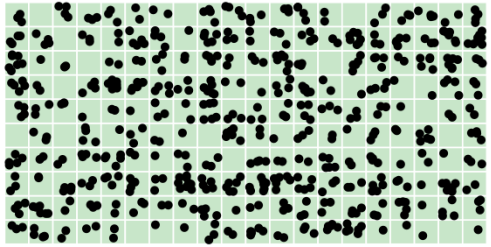
```
> set.seed(1)
> b <- rlnorm(100000, meanlog = log(20), sdlog = 0.7)
> median(b)     # exp(meanlog)
[1] 20.01122
> mean(b)       # exp(meanlog + sdlog^2/2)
[1] 25.54492
> exp(log(20) + 0.7^2 / 2)
[1] 25.55243
```

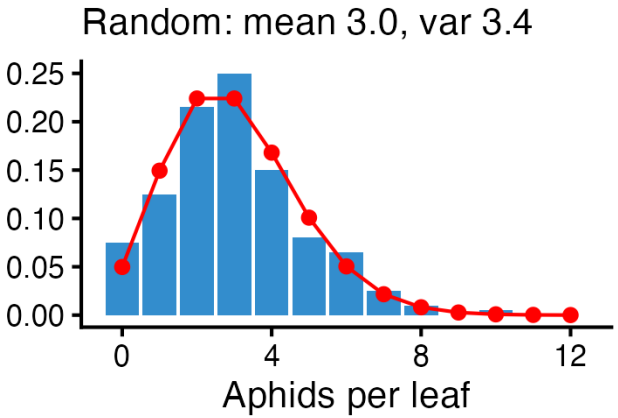


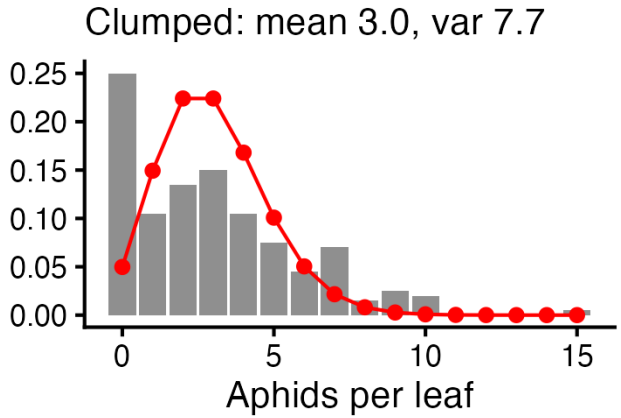
- **log(y) is normal**: things that grow multiplicatively (biomass, concentrations)
- Median 20 g, mean **25.5 g**: the long tail pulls the mean up

Aphids landing on leaves

600 aphids land at random







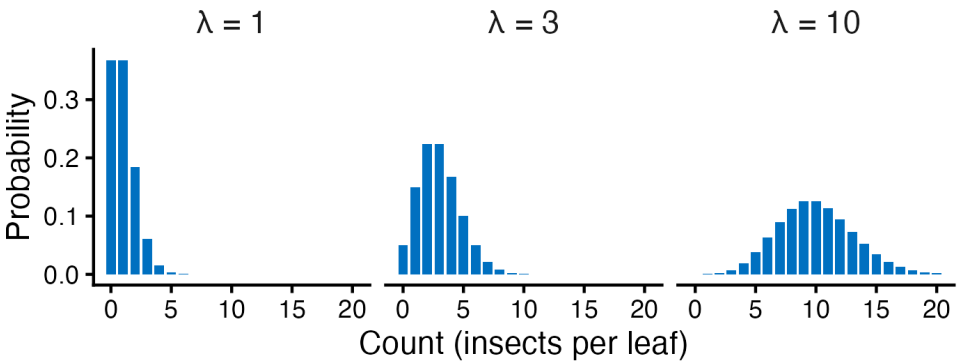

```
> set.seed(4)
> leaf <- sample(1:200, 600, replace = TRUE)  # 600 aphids
> counts <- tabulate(leaf, nbins = 200)
> c(mean(counts), var(counts))  # mean = variance
[1] 3.000000 3.356784
```

- Independent events, each **rare** on any one unit
- Clumped events: variance > mean (**overdispersed**; GLMs, Week 3)

Poisson distribution (counts)

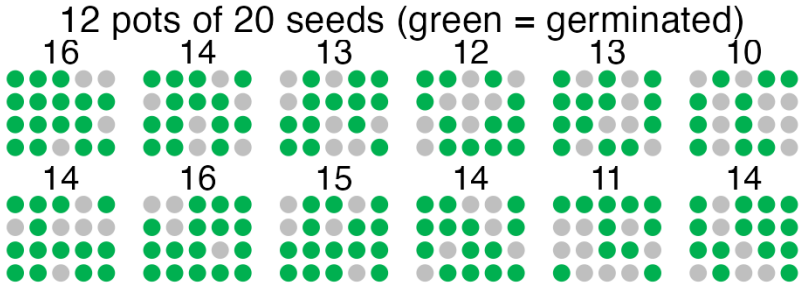
$$P(Y = k) = \frac{\lambda^k e^{-\lambda}}{k!} \qquad \mu = \sigma^2 = \lambda$$

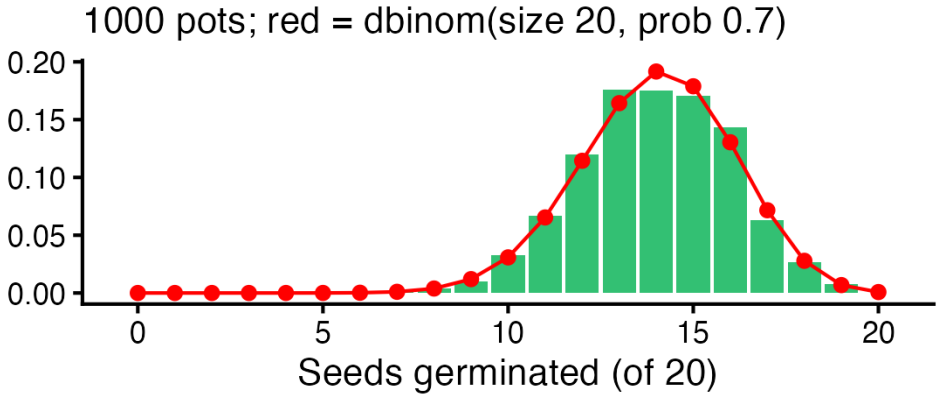
```
> set.seed(1)
> # insects per leaf
> insects <- rpois(10000, lambda = 3)
> mean(insects)
[1] 3.0051
> var(insects)              # = mean
[1] 3.040178
> dpois(0, lambda = 3)    # P(no insects)
[1] 0.04978707
```



Counts: insects per leaf, seeds per pod. **Mean = variance**, one parameter λ.

Seeds in a pot: each germinates or not





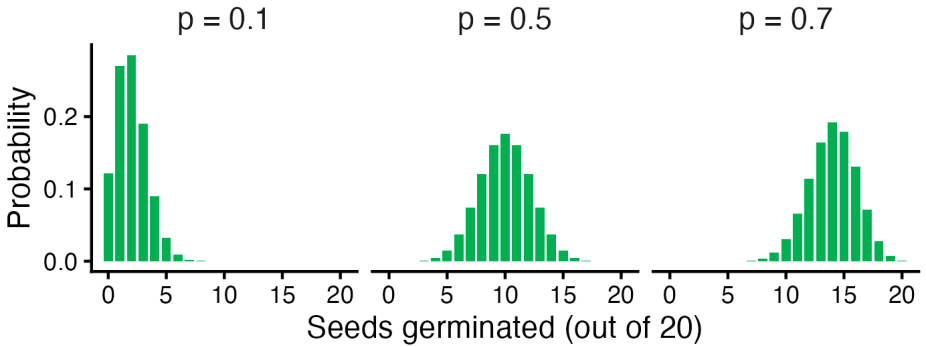

```
> set.seed(5)
> seeds <- matrix(runif(1000 * 20) < 0.7, ncol = 20)  # 70% coin
> germinated <- rowSums(seeds)                        # per pot of 20
> c(mean(germinated), var(germinated))  # 14 and 4.2
[1] 13.9480  4.3036
```

- n independent yes/no trials, **same p**, count the yeses
- Germination, survival, infection; alleles in a genotype

Binomial distribution

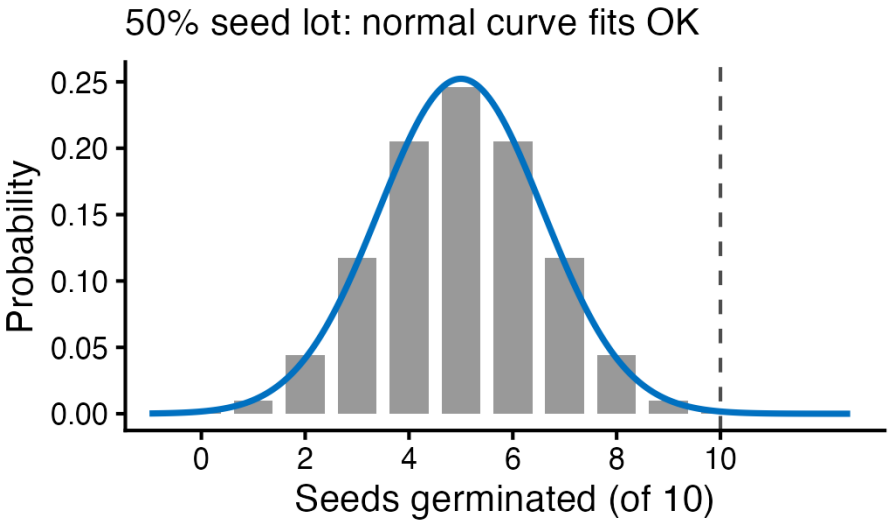
$$P(Y = k) = \binom{n}{k} p^k (1-p)^{n-k} \qquad \mu = np \quad \sigma^2 = np(1-p)$$

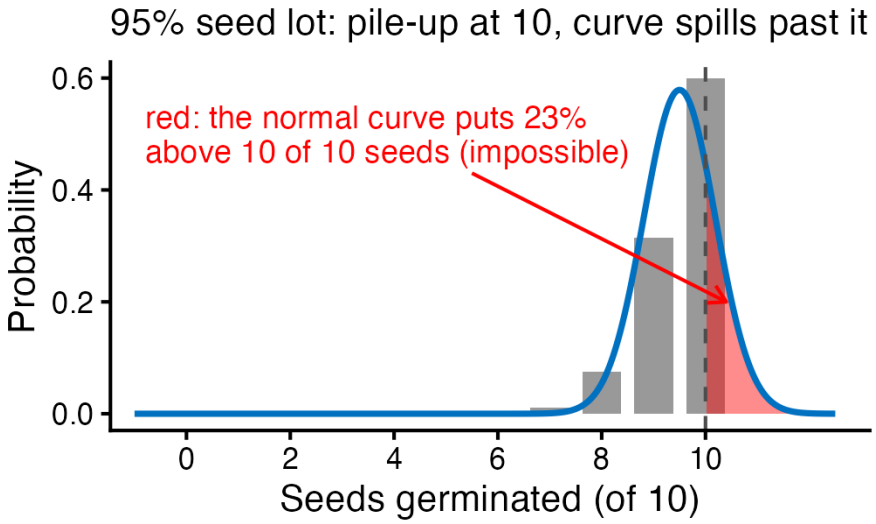
```
> # 20 seeds per pot, p = 0.7
> dbinom(14, size = 20, prob = 0.7)  # P(= 14)
[1] 0.191639
> pbinom(10, size = 20, prob = 0.7)  # P(<= 10)
[1] 0.0479619
> set.seed(1)
> rbinom(8, size = 20, prob = 0.7)   # 8 pots
[1] 15 15 14 11 16 11 11 13
```



Successes out of n trials: seeds germinated, plants surviving, infected leaves. Parameters: **size** (n) and **prob** (p).

Binomial: why not just normal?






```
> dbinom(10, size = 10, prob = 0.95)   # all 10: 60% of pots
[1] 0.5987369
> pnorm(10, mean = 9.5, sd = sqrt(10 * .95 * .05),
+       lower.tail = FALSE)            # normal: 23% above 10
[1] 0.23408
> 10 * c(0.5, 0.95) * (1 - c(0.5, 0.95))  # variance = np(1 - p)
[1] 2.500 0.475
```

- Whole numbers, stuck between **0 and n**
- Variance **np(1 − p)** is set by p: no separate σ
- Near 0% or 100%: skewed. Hence logistic regression (Week 3)

Distributions in R: cheat sheet

Distribution	R name	Parameters	Mean	Example
Normal	`norm`	mean, sd	μ	plant height
Gamma	`gamma`	shape, scale	$k\theta$	days to germination
Lognormal	`lnorm`	meanlog, sdlog	$e^{\mu+\sigma^2/2}$	shoot biomass
Exponential	`exp`	rate	$1/\text{rate}$	time to first event
Poisson	`pois`	lambda	λ	insects per leaf
Binomial	`binom`	size, prob	np	seeds germinated of 20
Uniform	`unif`	min, max	$(a+b)/2$	random location in a plot
t, χ^2, F	`t`, `chisq`, `f`	df		test statistics

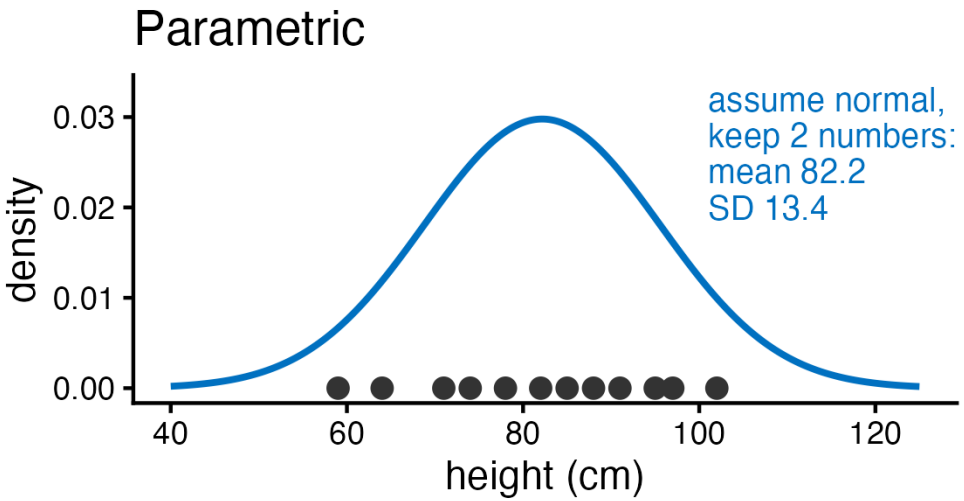
Every one has **d p q r**: dgamma(), pgamma(), qgamma(), rgamma() ...

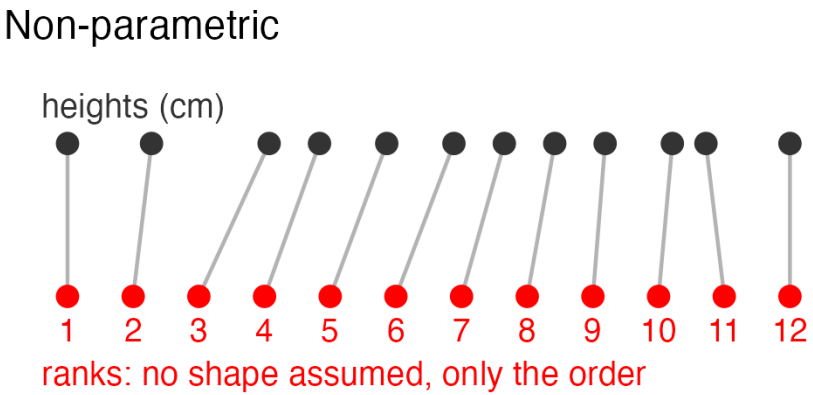
Notation, mean, and variance

Distribution	Notation	Mean $E[X]$	Variance
Normal	$X \sim N(\mu, \sigma^2)$	μ	σ^2
Lognormal	$X \sim \text{LN}(\mu, \sigma^2)$	$e^{\mu+\sigma^2/2}$	$(e^{\sigma^2} - 1)\, e^{2\mu+\sigma^2}$
Gamma	$X \sim \text{Gamma}(k, \theta)$	$k\theta$	$k\theta^2$
Poisson	$X \sim \text{Pois}(\lambda)$	λ	λ
Binomial	$X \sim \text{Bin}(n, p)$	np	$np(1-p)$

- Each has an **expected value** and a **variance** written in terms of its parameters
- Only the normal has separate knobs for center and spread. Poisson: variance = mean.

Parametric vs. non-parametric: the idea





- **Parametric**: assume a distribution (normal, Poisson…), then work with its few **parameters** (μ, σ, λ, p)
- **Non-parametric**: assume no shape; work with **ranks** or reshuffles of the data

```
> rank(height)                 # non-parametric: only the order
 [1]  5 10  2  8 12  3  7  9  1 11  6  4
> typo <- c(height, 820)       # one typo
> c(mean(height), mean(typo))  # the mean jumps
[1]  82.16667 138.92308
> rank(typo)[13]               # the typo is just rank 13
[1] 13
```

Parametric vs. non-parametric: the math

Parametric $y_i \sim N(\mu, \sigma^2) \qquad H_0 : \ \mu_1 = \mu_2$

Non-parametric $y_i \sim F \text{ (any shape)} \qquad H_0 : \ F_1 = F_2$

Parametric	Non-parametric
one-sample or paired t-test	Wilcoxon signed-rank
two-sample t-test	Wilcoxon rank-sum; permutation test
Pearson r	Spearman ρ (Pearson on ranks)
one-way ANOVA	Kruskal–Wallis (ANOVA on ranks)

- Parametric: more power when the assumptions hold, answers in real units (a difference in cm)
- Non-parametric: robust to skew and outliers, ~95% of the t-test's power on normal data

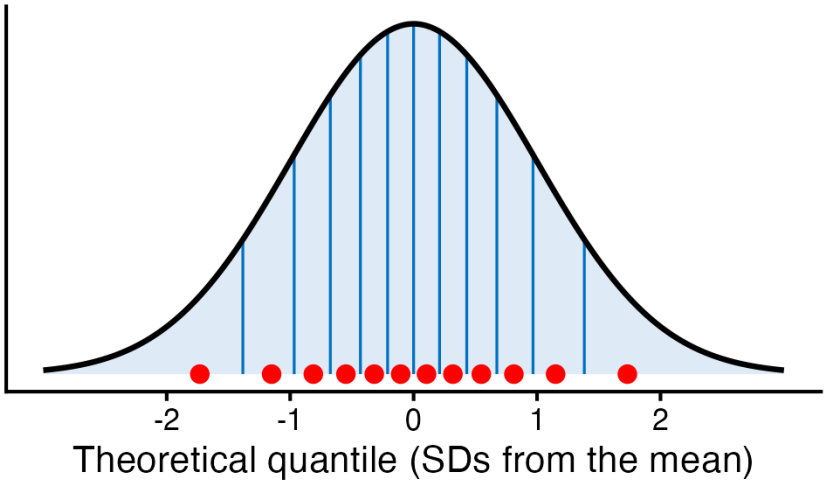
Non-parametric ≠ assumption-free: observations still have to be independent.

Checking normality

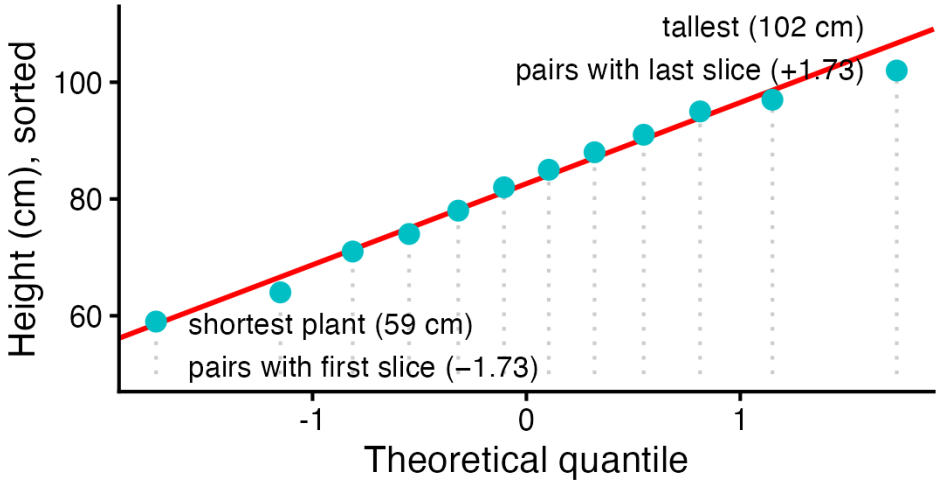
and transforming data

What are theoretical quantiles?

Normal curve cut into 12 equal-area slices;
red = the middle of each slice



Q–Q plot: 12 sorted plants vs. the 12 red points



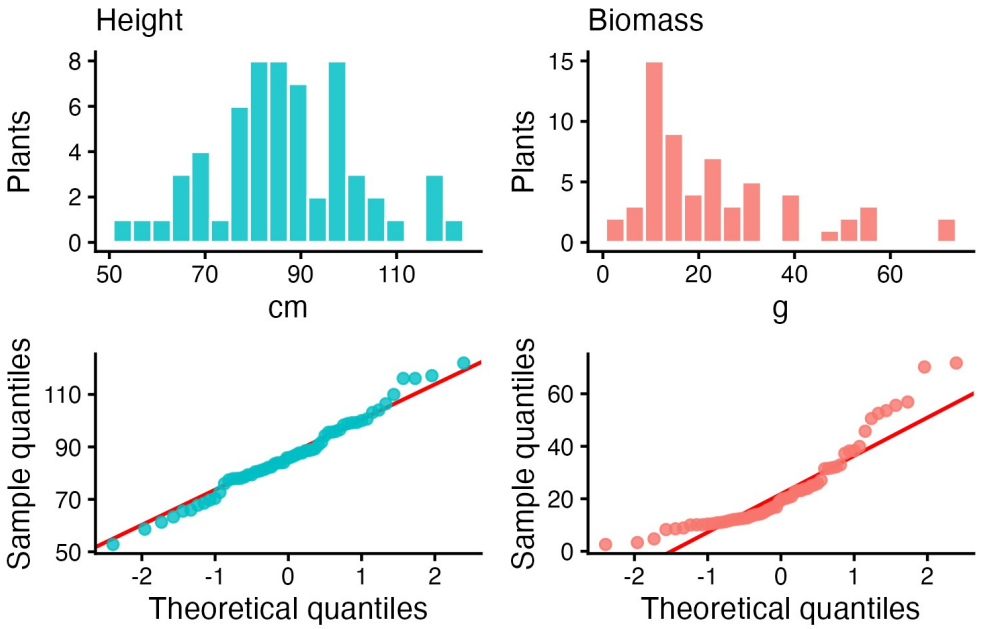

```
> height <- c(78, 95, 64, 88, 102, 71, 85, 91, 59, 97, 82, 74)
> sort(height)                    # sample quantiles
 [1]  59  64  71  74  78  82  85  88  91  95  97 102
> round(qnorm(ppoints(12)), 2)    # theoretical quantiles
 [1] -1.73 -1.15 -0.81 -0.55 -0.32 -0.10  0.10  0.32  0.55  0.81
[11]  1.15  1.73
> qqnorm(height); qqline(height, col = "red")
```

- **Theoretical quantile**: where the k-th smallest of n values should sit if the data were perfectly normal (in SDs)
- Q–Q plot: sorted data vs. these. Normal → **straight line**

Is it normal? Look first

```
> set.seed(206)
> height  <- rnorm(60, mean = 82, sd = 16)    # cm
> biomass <- rlnorm(60, meanlog = log(20),
+                   sdlog = 0.7)              # grams
> summary(biomass)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.593  11.975  19.474  23.675  31.622  71.719
```

```
> qqnorm(height);  qqline(height, col = "red")
> qqnorm(biomass); qqline(biomass, col = "red")
```



Height: symmetric, points on the line = **normal**. **Biomass**: long right tail, points **curve up** at the right = right skew.
S-shape (both ends off the line) = heavy tails.

Shapiro–Wilk test

H_0 : the data come from a normal distribution

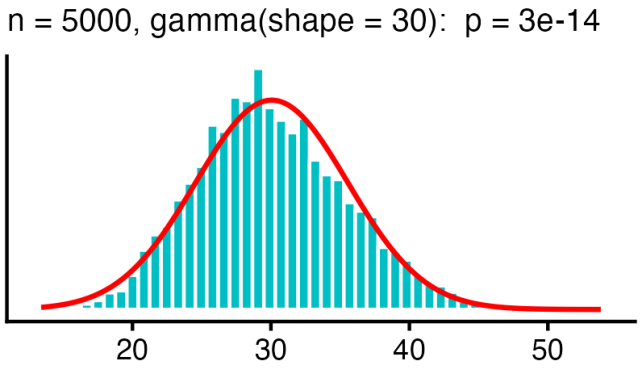
```
> shapiro.test(height)

        Shapiro-Wilk normality test

data:  height
W = 0.98744, p-value = 0.7946

> shapiro.test(biomass)

        Shapiro-Wilk normality test

data:  biomass
W = 0.87259, p-value = 1.515e-05
```

Height: p = 0.79 → no evidence against normality

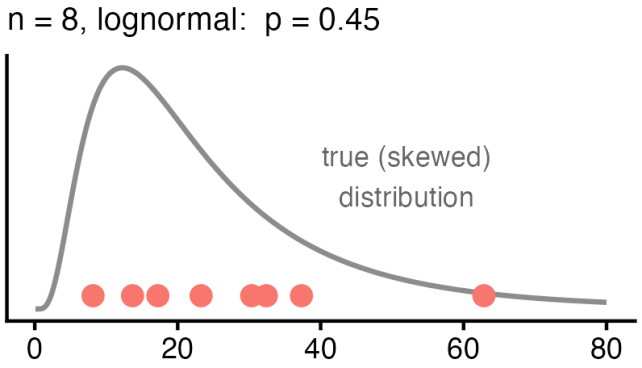
Biomass: p = 0.00002 → not normal

Careful with normality tests

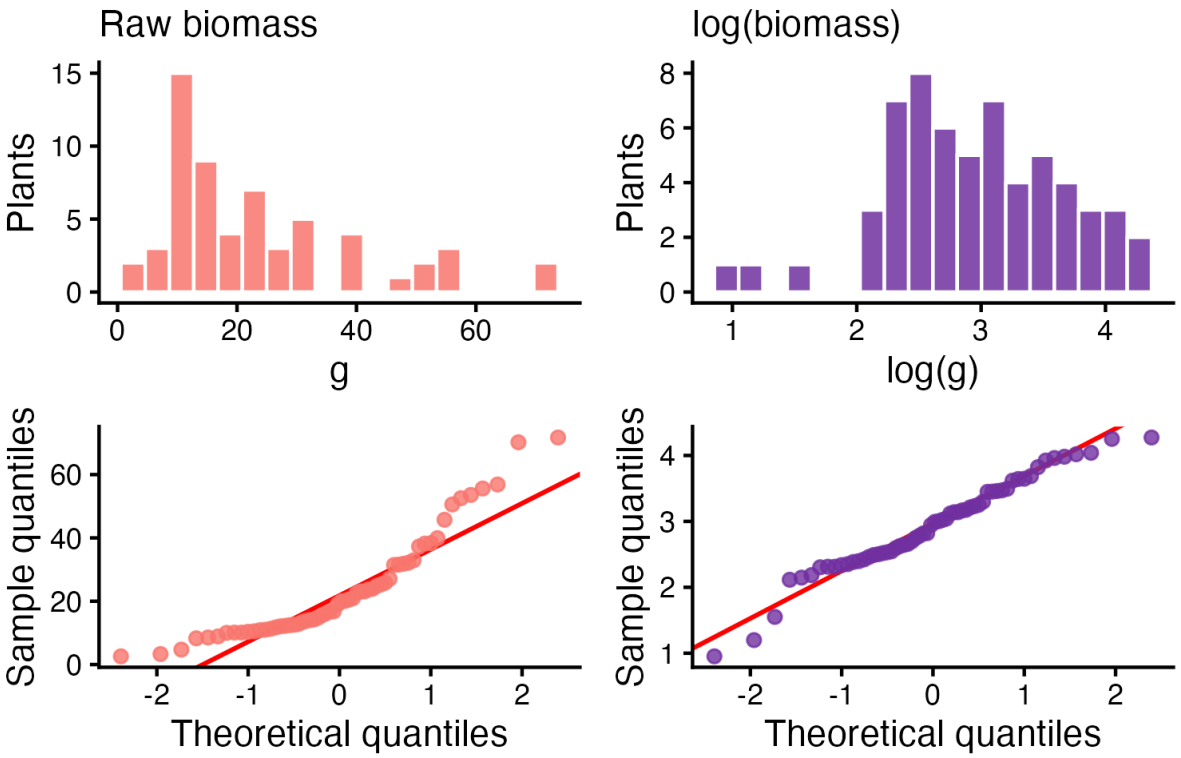
```
> set.seed(1)
> big <- rgamma(5000, shape = 30)   # barely skewed, n = 5000
> shapiro.test(big)$p.value
[1] 2.839524e-14
> set.seed(4)
> small <- rlnorm(8, meanlog = log(20), sdlog = 0.7)
> shapiro.test(small)$p.value       # truly skewed, n = 8
[1] 0.4505372
```



- **Large n:** flags trivial departures (top: barely skewed, p = 3e-14)
- **Small n:** misses real skew (bottom: truly skewed, p = 0.45)
- **Look at the Q–Q plot.** The test is a supplement, not a verdict.



Log transformation

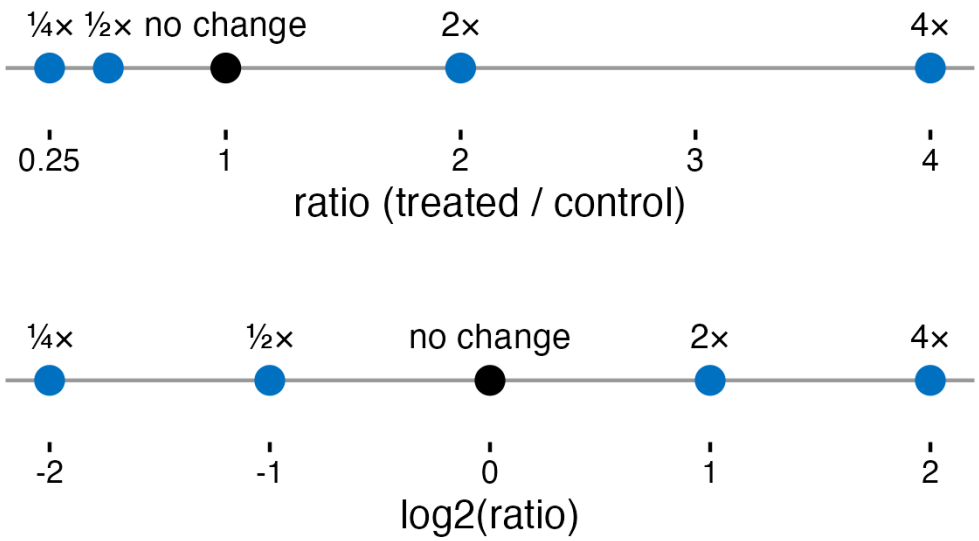


$$y' = \log(y)$$

```
> log_biomass <- log(biomass)
> shapiro.test(log_biomass)$p.value
[1] 0.2002304
```

After logging, biomass looks normal: **p = 0.20**

Logs and ratios



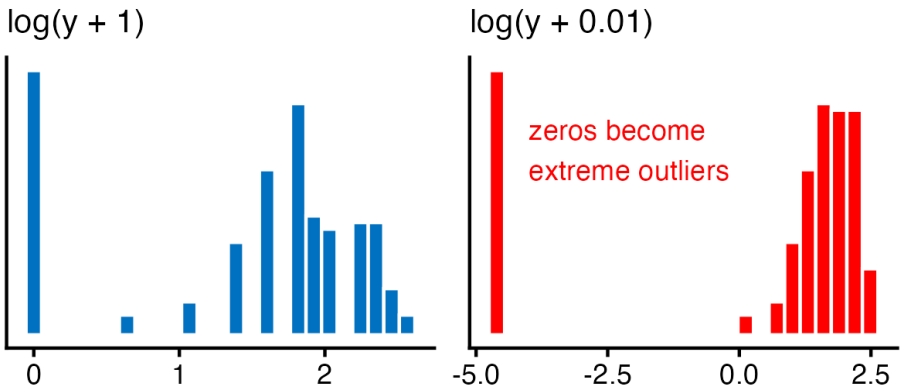
$$\log_2(2) = +1 \qquad \log_2(1) = 0 \qquad \log_2(\tfrac{1}{2}) = -1$$

```
> # treated / control yield
> r <- c(0.25, 0.5, 1, 2, 4)
> log2(r)
[1] -2 -1  0  1  2
> two <- c(0.5, 2)      # halved, doubled
> mean(two)             # arithmetic mean
[1] 1.25
> 2^mean(log2(two))     # geometric mean
[1] 1
```

- Raw ratios: halving sits 0.5 below 1, doubling sits 1 above: **skewed**
- log2: halving = −1, doubling = +1, no change = 0: **symmetric**. Average ratios on the log scale.
- **Arithmetic mean** 1.25: total yield up 25%. **Geometric mean** 1: the typical proportional effect; a halving and a doubling cancel.

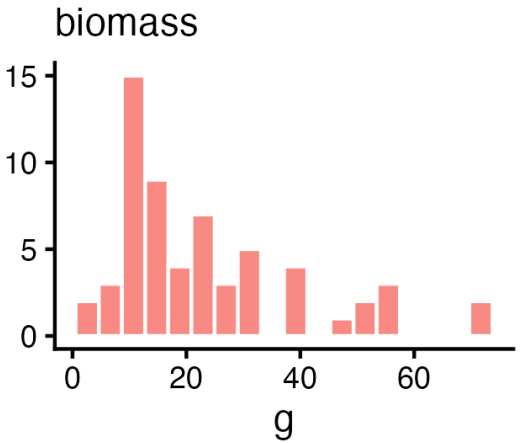
Zeros and negatives

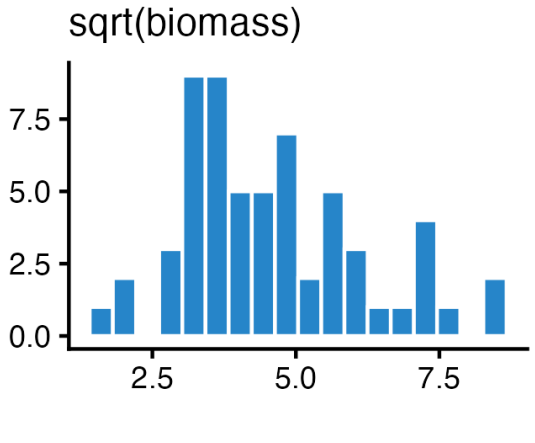
```
> log(0)
[1] -Inf
> log(-1)
[1] NaN
> counts <- c(0, 0, 1, 2, 3, 5, 8, 13)
> log(counts + 1)      # = log1p(counts)
[1] 0.0000000 0.0000000 0.6931472 1.0986123 1.3862944 1.7917595
[7] 2.1972246 2.6390573
> log(counts + 0.01)   # different c, different answer
[1] -4.605170186 -4.605170186  0.009950331  0.698134722
[5]  1.101940079  1.611435915  2.080690761  2.565718293
```

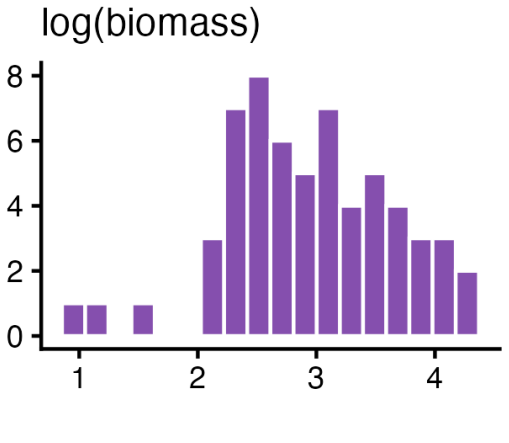


- log(0) = −Inf, log(negative) = NaN
- log(y + c): the answer **depends on c**; zeros can become outliers
- Many zeros or negatives: use INT, or a GLM (Week 3)

Common transformations

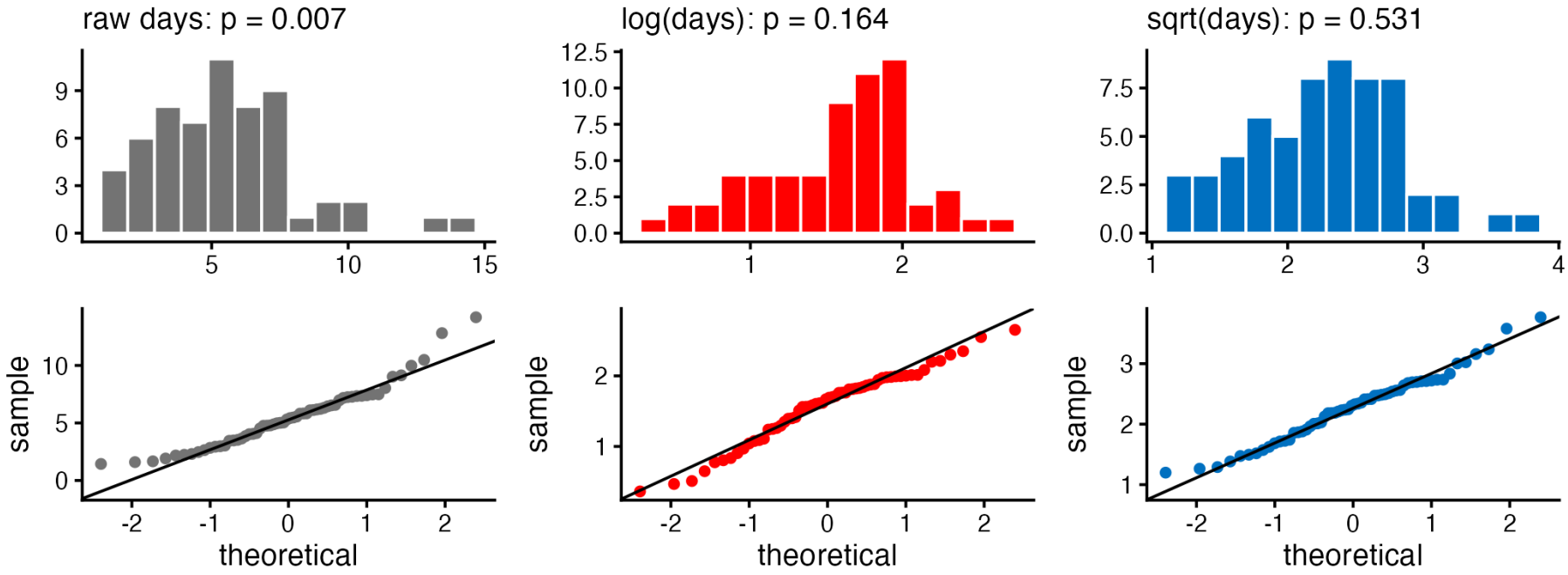






Transformation	Typical use
$\sqrt{y}$	counts (insects per leaf)
$\log(y)$	right-skewed, multiplicative (biomass)
$\log(y+1)$	right-skewed with zeros
$\log \frac{p}{1-p}$	proportions (logit)

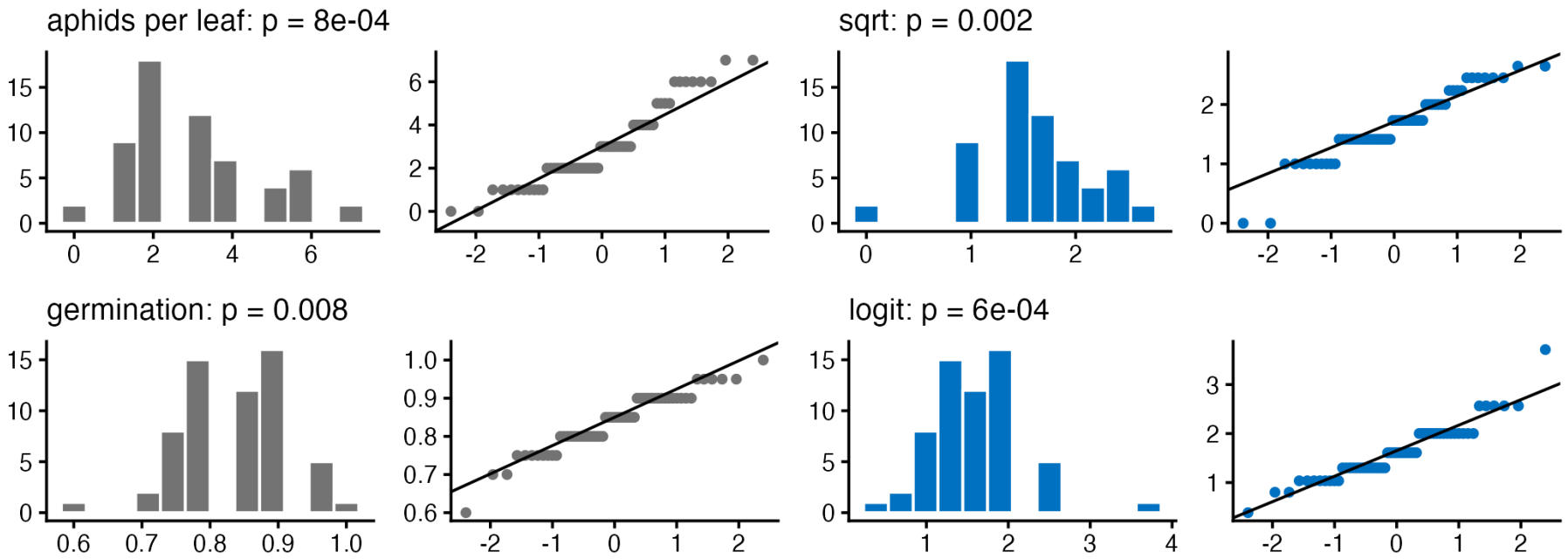
Checking the moth-trap data




```
> set.seed(53)
> days <- rgamma(60, shape = 4, scale = 1.5)
> shapiro.test(days)$p.value
[1] 0.006995453
> shapiro.test(log(days))$p.value   # too strong
[1] 0.1637573
> shapiro.test(sqrt(days))$p.value  # about right
[1] 0.5312969
```

log passes the test but overshoots (bends the other way). **sqrt** is the right strength.

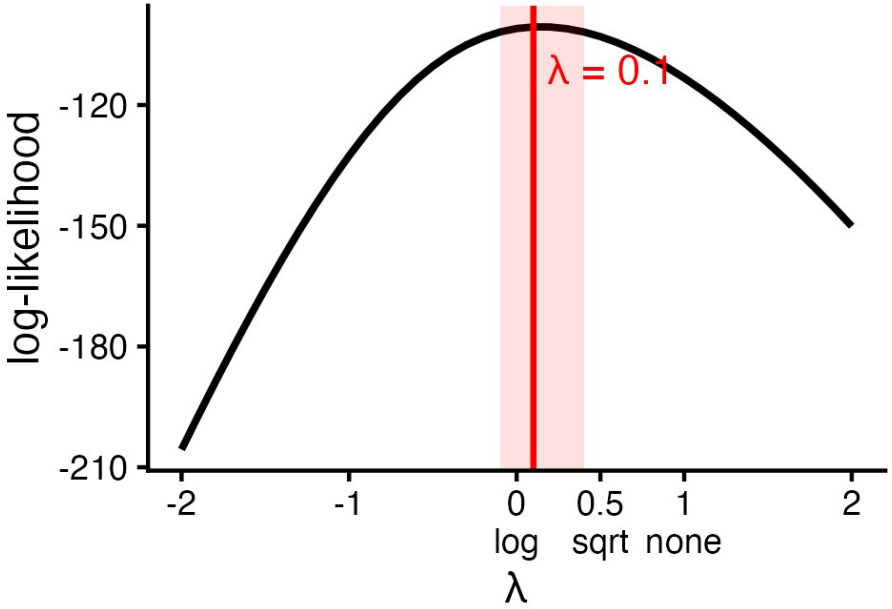
Counts: transforming isn't enough



- Only a handful of possible values: the Q-Q plot is a **staircase**, and no transformation removes the steps
- Model them as what they are: Poisson and binomial **GLMs** (Week 3), or use a rank-based test

Box–Cox: let the data pick λ

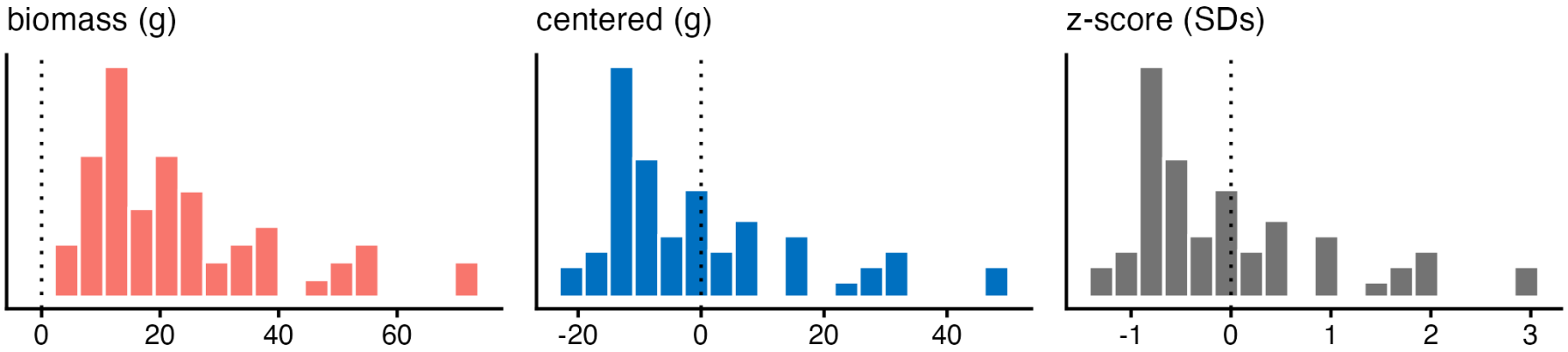
$$y^{(\lambda)} = \frac{y^{\lambda} - 1}{\lambda} \qquad (\lambda = 0 : \ \log y)$$




```
> library(MASS)
> bc <- boxcox(biomass ~ 1, plotit = FALSE)
> bc$x[which.max(bc$y)]   # best lambda
[1] 0.1
```

- Biomass: λ ≈ 0 → **log**
- Needs y > 0
- **Yeo–Johnson** extends it to zeros and negatives (**bestNormalize** package)

Standardizing is not normalizing



Centering

$$y - \bar{y}$$

mean 0, units and SD kept

Standardizing = z-score

$$z = \frac{y - \bar{y}}{s}$$

mean 0, SD 1: "how many SDs from the mean"

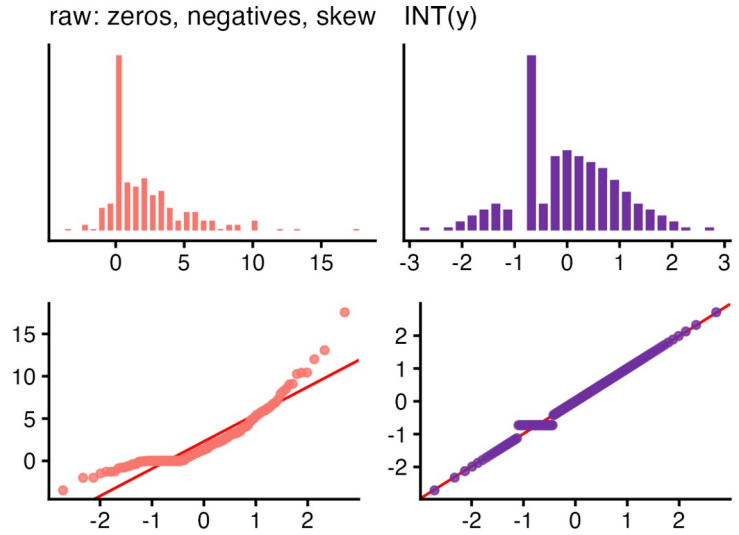
```
> centered <- scale(biomass, scale = FALSE)  # y - mean
> z        <- scale(biomass)                 # (y - mean) / sd
> round(c(mean(centered), sd(centered)), 2)  # mean 0, SD kept
[1]  0.00 16.25
> round(c(mean(z), sd(z)), 2)                # mean 0, SD 1
[1] 0 1
> shapiro.test(z)$p.value                    # same as raw
[1] 1.514571e-05
```

Neither changes the **shape**: same skew, same Shapiro p. Use them to put variables on a common scale (PCA, regularization), not to fix skew.

Rank-based inverse normal (INT)

$$\mathrm{INT}(y) = \Phi^{-1}\left(\frac{\mathrm{rank}(y) - 0.5}{n}\right)$$

```
> int <- function(y) qnorm((rank(y) - 0.5) / length(y))
> set.seed(5)
> # zeros, negatives, and skew
> y <- c(rep(0, 30), -rexp(20, 1), rexp(100, 0.3))
> round(head(sort(y), 3), 2)
[1] -3.48 -2.00 -1.99
> shapiro.test(y)$p.value
[1] 7.355401e-11
> shapiro.test(int(y))$p.value
[1] 0.5896974
```



- **Works on anything:** zeros, negatives, any shape. Forces a normal distribution.
- **Costs:** original units are gone (only ranks remain) and ties stay tied (the flat step).

Transformations change the question

```
> mean(biomass)              # arithmetic mean
[1] 23.67545
> exp(mean(log(biomass)))    # geometric mean
[1] 18.89643
> median(biomass)
[1] 19.47377
```

- Mean of the logs, back-transformed = **geometric mean** (18.9 g), close to the median
- The arithmetic mean (23.7 g) is pulled up by the long tail
- Differences on the log scale are **ratios**: "droughted plants were 30% lighter"

When do my data need to be normal?

What tests assume normality?

Test	What has to be normal	If it isn't
one-sample t-test	the data	Wilcoxon signed-rank
paired t-test	the differences	Wilcoxon signed-rank
two-sample t-test	each group	Wilcoxon rank-sum, permutation
ANOVA	the residuals	Kruskal–Wallis
linear regression	the residuals	transform, or a GLM
Pearson's r (the test)	both variables, roughly	Spearman's ρ

- **No normality needed:** rank tests, permutation tests, binomial / Fisher / chi-squared, GLMs (they use their own distribution)
- Even when assumed: it's for the **p-value and CI**, and matters less as n grows

Monday recap

Kahoot-style warm-up

14 quick questions on Monday's material.

Not graded.

Use pseudonym if you want.

Fast and correct wins.

Scan with your phone

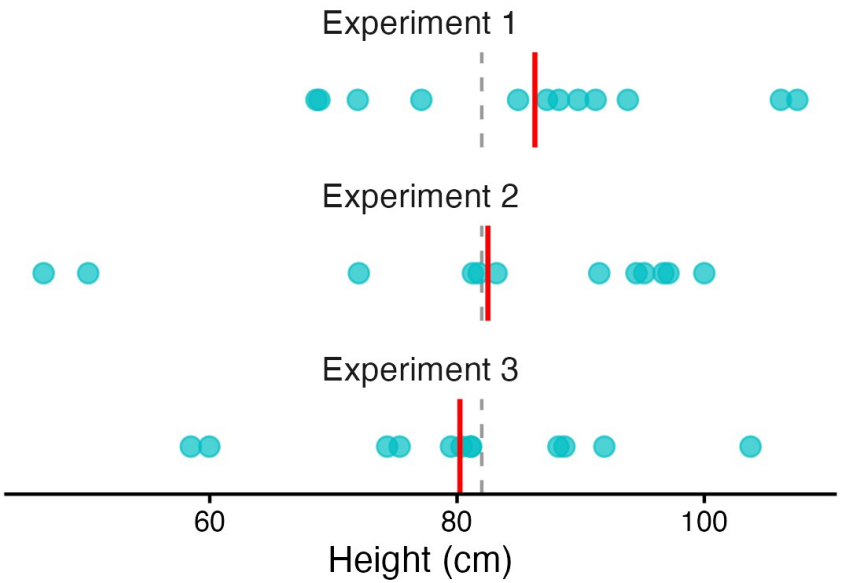
[Join on your phone or laptop: greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w1-recap](greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w1-recap)

[host screen](greymonroe.github.io/live-poll-test/quiz-host.html?quiz=pls206-w1-recap)

Standard error and confidence intervals

One experiment = one sample

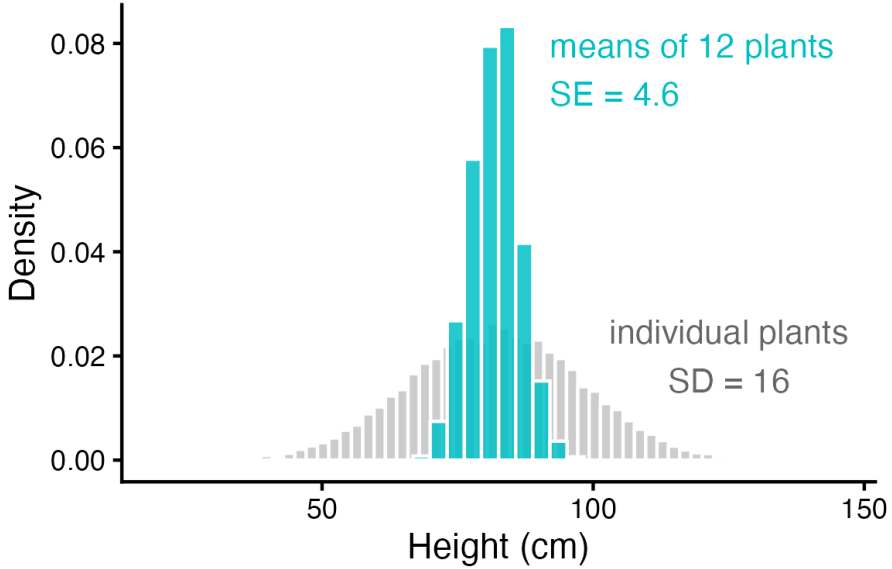
```
> set.seed(1)
> mean(rnorm(12, mean = 82, sd = 16))  # exp. 1
[1] 86.2982
> mean(rnorm(12, mean = 82, sd = 16))  # exp. 2
[1] 82.49754
> mean(rnorm(12, mean = 82, sd = 16))  # exp. 3
[1] 80.23286
```



Repeat the same experiment and the mean **changes every time**.

Sampling distribution of the mean

```
> set.seed(206)
> means <- replicate(2000,
+                    mean(rnorm(12, mean = 82, sd = 16)))
> sd(means)       # SD of the 2000 means
[1] 4.518642
> 16 / sqrt(12)   # the formula
[1] 4.618802
```



The spread of the means is much **narrower** than the spread of the plants

Standard error

$$SE = \frac{s}{\sqrt{n}}$$

$$SE = \frac{13.4}{\sqrt{12}} = 3.87 \text{ cm}$$

```
> se <- sd(height) / sqrt(length(height))
> se
[1] 3.866784
```

SE = how much the mean would bounce around if we repeated the experiment. It measures the **precision of our estimate**.

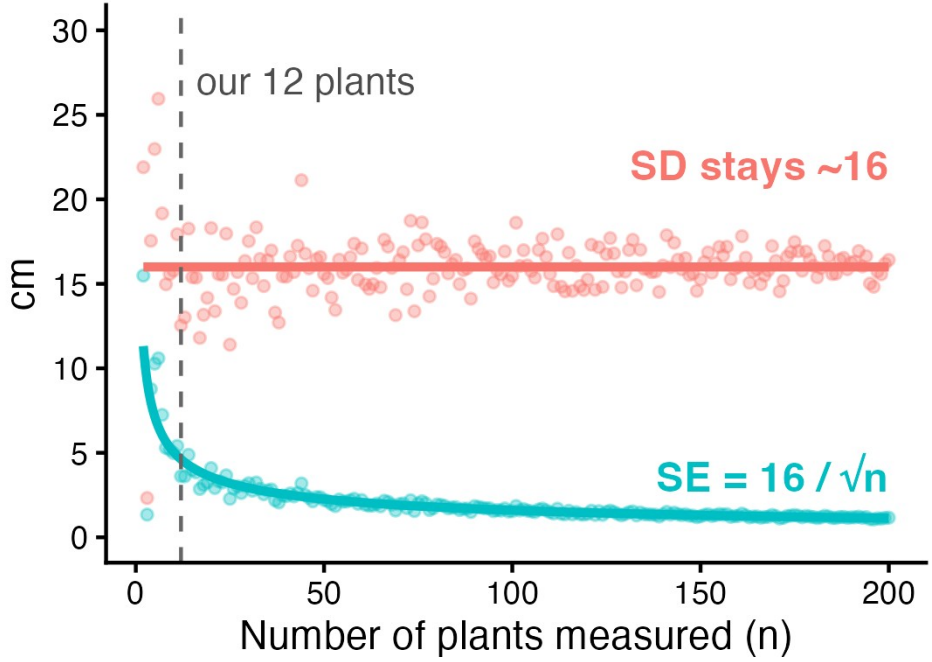
SD vs. SE

SD

Spread of individual plants
Property of the population
Does not shrink with n

SE

Spread of the sample mean
Property of the experiment
Shrinks with √n



Live poll

Number: the class builds the central limit theorem

Everyone: simulate 50 plants' biomass (skewed!) in R. No set.seed().

```
> b <- rlnorm(50, meanlog = log(20), sdlog = 0.7)
> hist(b)     # 50 plants: long right tail
> mean(b)     # submit this number
```

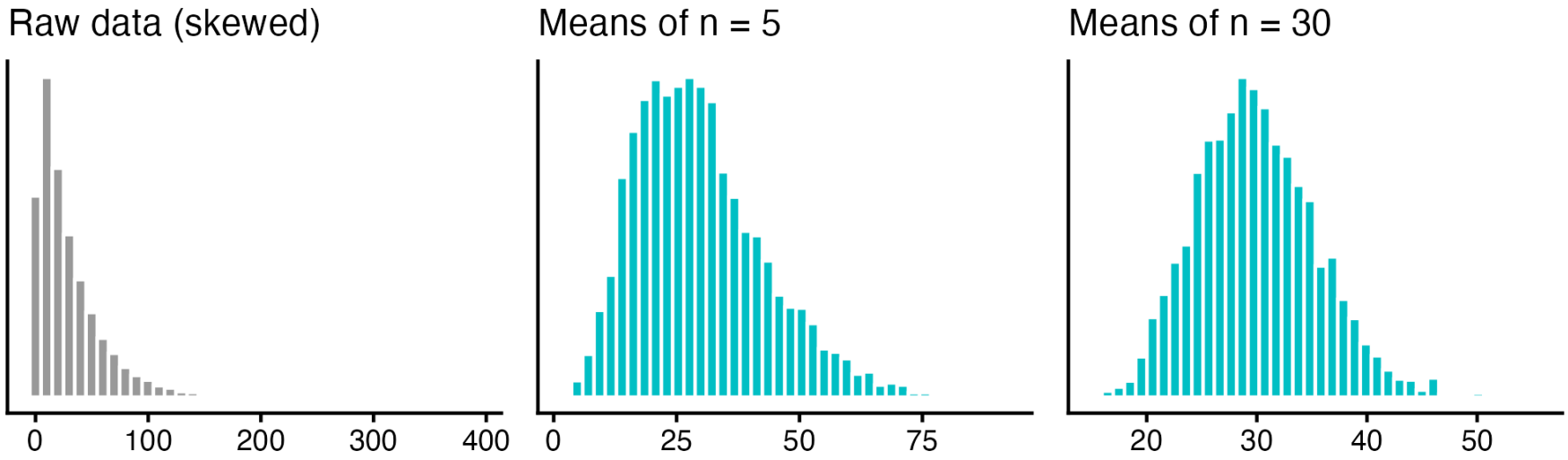
Look at your histogram, then submit **mean(b)** in grams.

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-means

host screen

Central limit theorem



Means of samples are approximately **normal** even when the data are not, and more so as n grows.

Confidence intervals

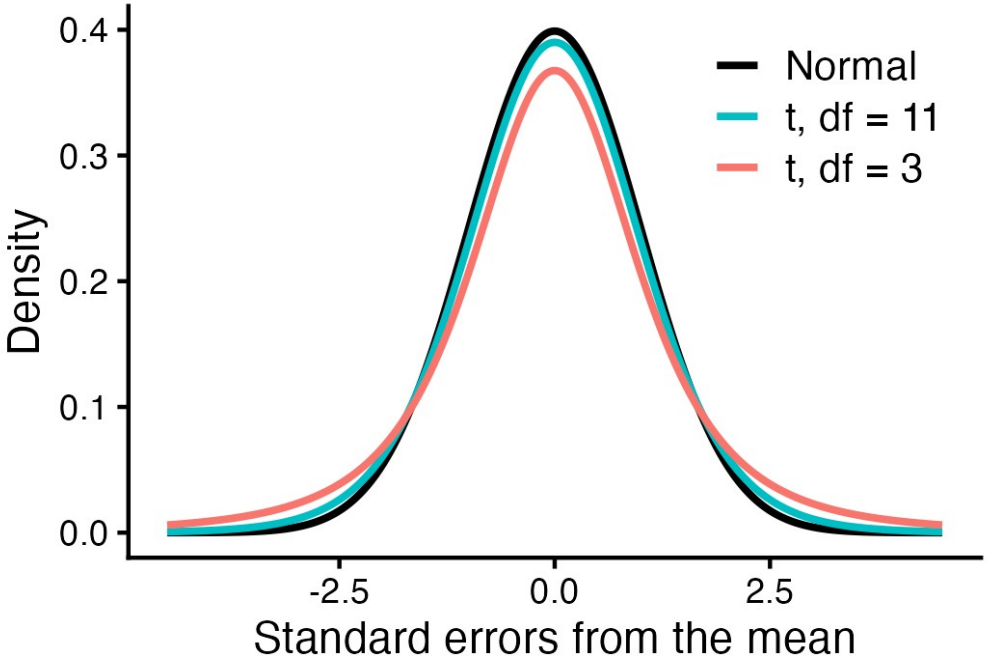
$$\bar{x} \pm t_{0.975,\, n-1} \times SE$$

$$82.2 \pm 2.20 \times 3.87 = [73.7,\ 90.7]\ \text{cm}$$

```
> qnorm(0.975)              # normal
[1] 1.959964
> qt(0.975, df = n - 1)     # t with 11 df
[1] 2.200985
> xbar  <- mean(height)
> tcrit <- qt(0.975, df = n - 1)
> c(lower = xbar - tcrit * se,
+   upper = xbar + tcrit * se)
   lower    upper
73.65593 90.67740
```

t instead of 1.96 because we **estimated** s from 12 plants

The t distribution



- Like the normal, but heavier tails
- Extra uncertainty from estimating s
- Approaches the normal as **df = n − 1** grows

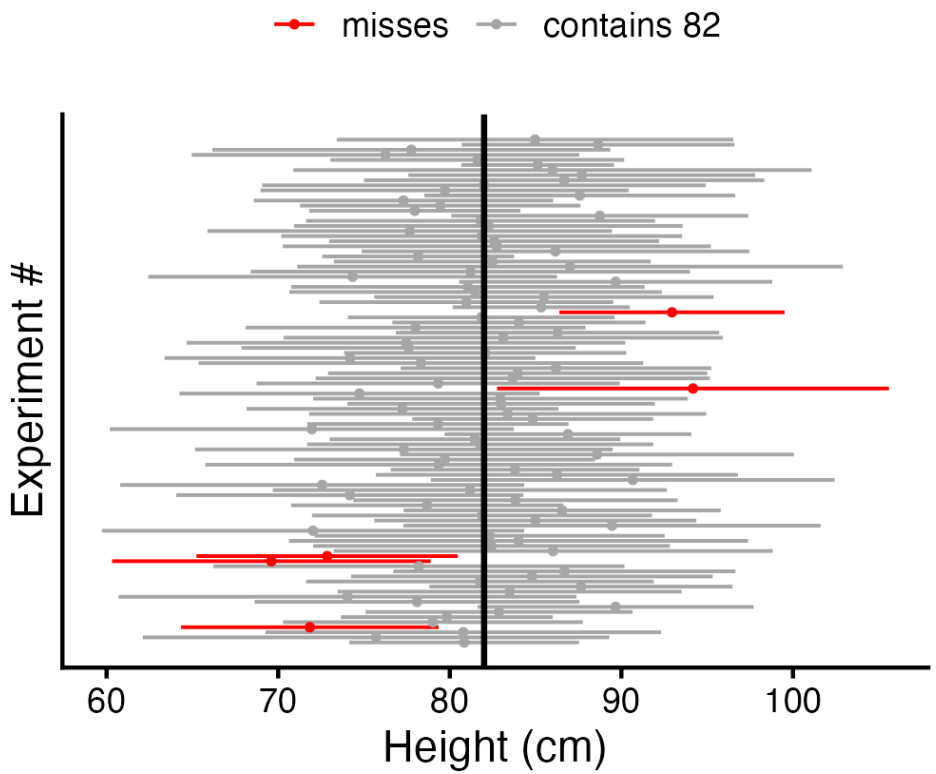
t.test() gives the CI

```
> t.test(height)

        One Sample t-test

data:  height
t = 21.249, df = 11, p-value = 2.787e-10
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 73.65593 90.67740
sample estimates:
mean of x
 82.16667
```

95 percent confidence interval: **73.66 to 90.68** — same as by hand

100 experiments, 100 intervals

5 of 100 intervals miss the true mean



- Each line: one experiment of 12 plants and its 95% CI
- About **5 in 100** miss the true mean
- We never know whether ours is one of them

Live poll

Multiple choice

Our 95% CI for mean height is 73.7 to 90.7 cm. Which statement is correct?

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-ci

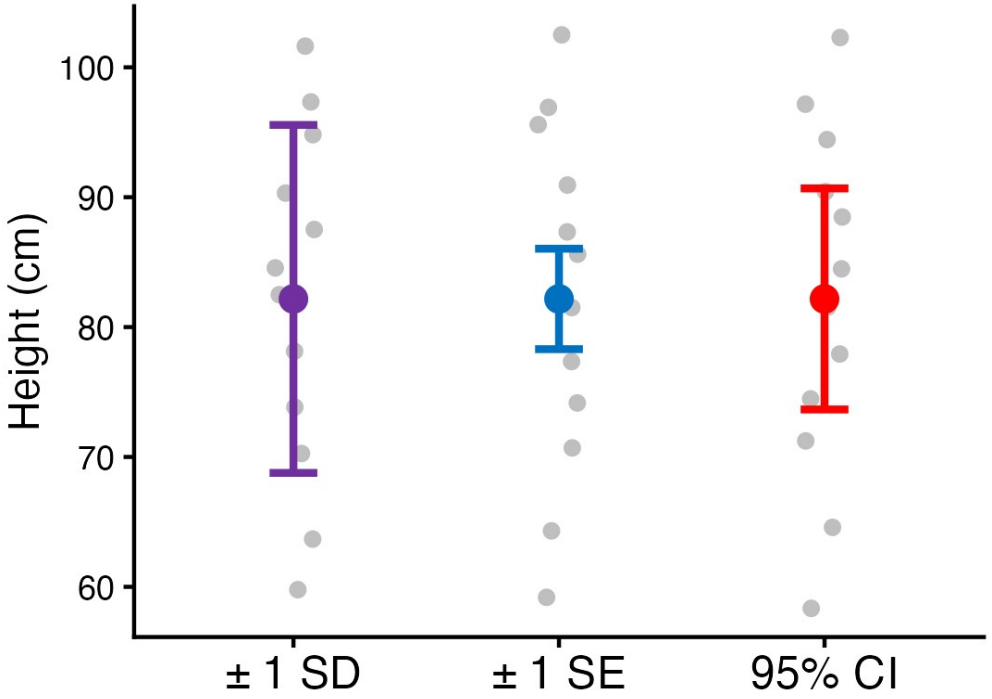
host screen

What 95% means

✗ There is a 95% probability the true mean is in this interval

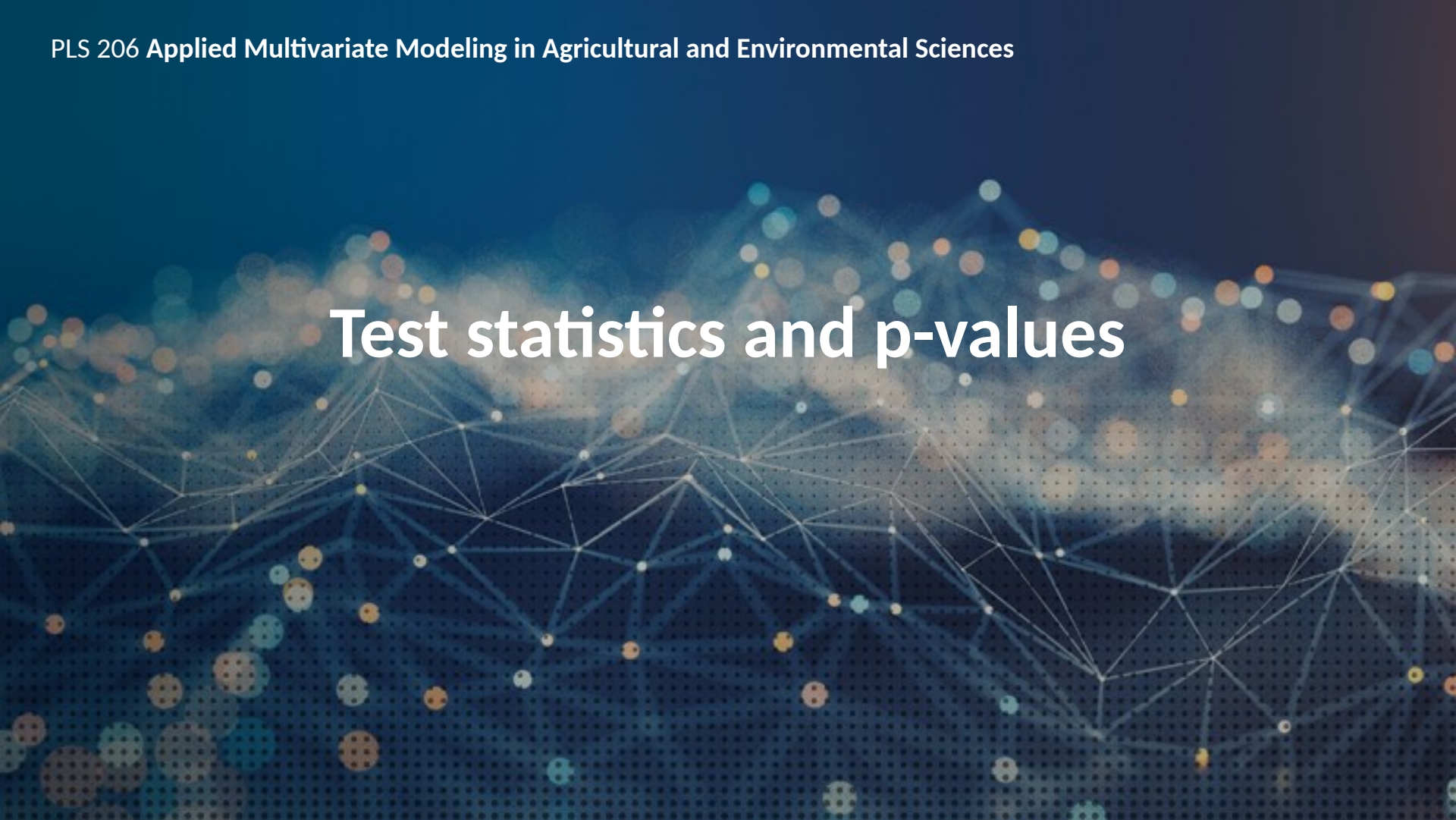
✓ 95% of intervals built this way contain the true mean

Which error bar?



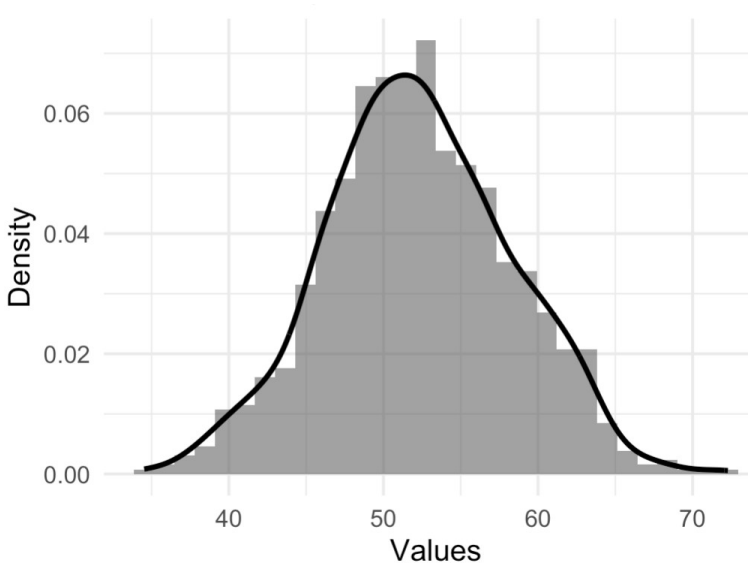
± 1 SD spread of plants
± 1 SE precision of the mean
95% CI plausible range for μ

Always say which one in your figure legend.

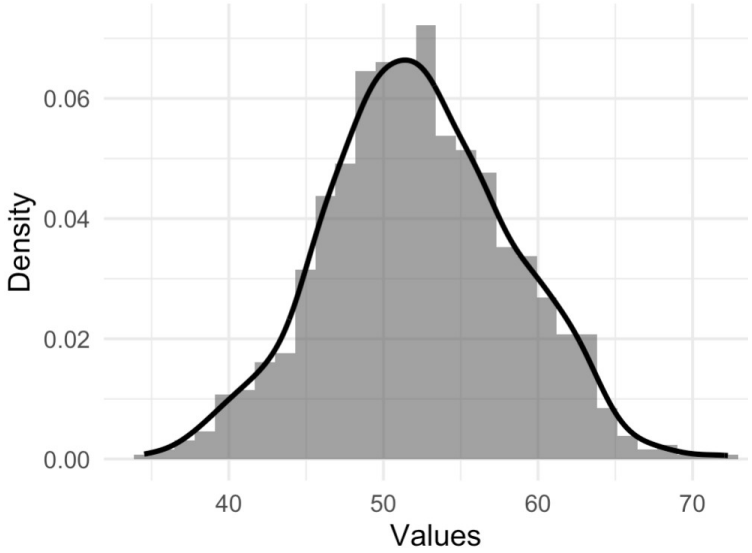


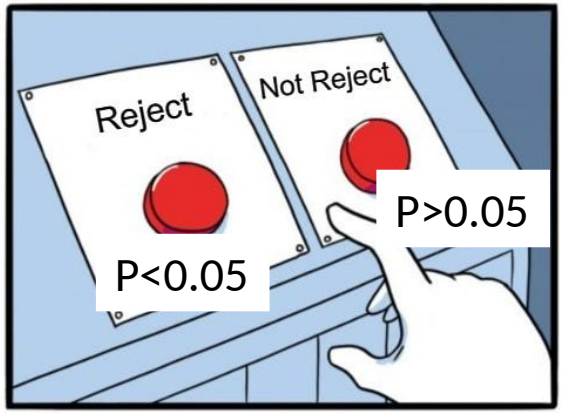
Test statistics and p-values

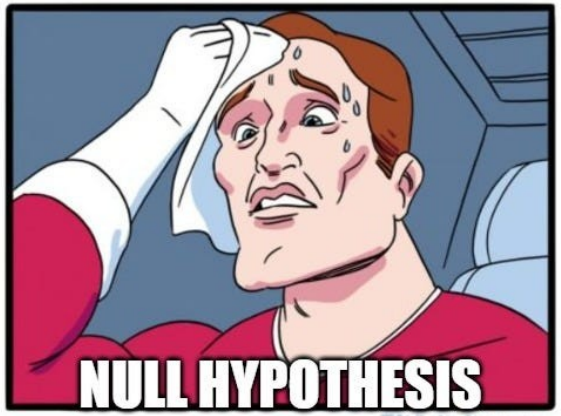
```
> x
 [1] 48.63715 50.61894 61.35225 52.42305 52.77573 62.29039 54.76550
 [8] 44.40963 47.87888 49.32603 59.34449 54.15888 54.40463 52.66410
[15] 48.66495 62.72148 54.98710 40.20030 56.20814 49.16325 45.59306
[22] 50.69215 45.84397 47.62665 48.24976 41.87984 57.02672 52.92024
[29] 45.17118 59.52289 54.55879 50.22957 57.37075 57.26880 56.92949
[36] 56.13184 55.32351 51.62853 50.16422 49.71717 47.83176 50.75250
[43] 44.40762 65.01374 59.24777 45.26135 49.58269 49.20007 56.67979
[50] 51.49979 53.51991 51.82872 51.74278 60.21161 50.64537 61.09882
[57] 42.70748 55.50768 52.74313 53.29565 54.27784 48.98606 50.00076
[64] 45.88855 45.56925 53.82117 54.68926 52.31803 57.53360 64.30051
[71] 49.05381 38.14499 58.03443 47.74480 47.87195 58.15343 50.29136
[78] 44.67569 53.08782 51.16665 52.03459 54.31168 49.77604 55.86626
[85] 50.67708 53.99069 58.58103 54.61109 50.04441 58.89285 57.96102
[92] 55.29038 53.43239 48.23256 60.16391 48.39844 65.12400 61.19566
[99] 50.58580 45.84147
```



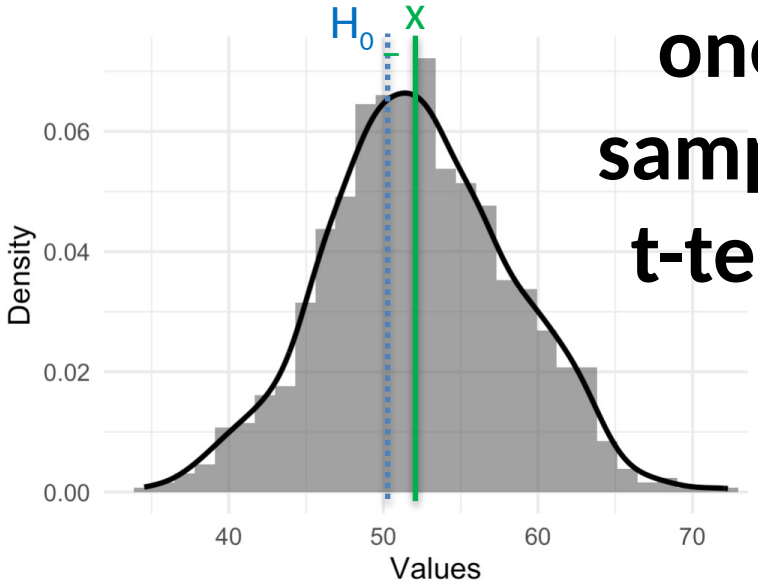
```
> x
 [1] 48.63715 50.61894 61.35225 52.42305 52.77573 62.29039 54.76550
 [8] 44.40963 47.87888 49.32603 59.34449 54.15888 54.40463 52.66410
[15] 48.66495 62.72148 54.98710 40.20030 56.20814 49.16325 45.59306
[22] 50.69215 45.84397 47.62665 48.24976 41.87984 57.02672 52.92024
[29] 45.17118 59.52289 54.55879 50.22957 57.37075 57.26880 56.92949
[36] 56.13184 55.32351 51.62853 50.16422 49.71717 47.83176 50.75250
[43] 44.40762 65.01374 59.24777 45.26135 49.58269 49.20007 56.67979
[50] 51.49979 53.51991 51.82872 51.74278 60.21161 50.64537 61.09882
[57] 42.70748 55.50768 52.74313 53.29565 54.27784 48.98606 50.00076
[64] 45.88855 45.56925 53.82117 54.68926 52.31803 57.53360 64.30051
[71] 49.05381 38.14499 58.03443 47.74480 47.87195 58.15343 50.29136
[78] 44.67569 53.08782 51.16665 52.03459 54.31168 49.77604 55.86626
[85] 50.67708 53.99069 58.58103 54.61109 50.04441 58.89285 57.96102
[92] 55.29038 53.43239 48.23256 60.16391 48.39844 65.12400 61.19566
[99] 50.58580 45.84147
```







```
> x
 [1] 48.63715 50.61894 61.35225 52.42305 52.77573 62.29039 54.76550
 [8] 44.40963 47.87888 49.32603 59.34449 54.15888 54.40463 52.66410
[15] 48.66495 62.72148 54.98710 40.20030 56.20814 49.16325 45.59306
[22] 50.69215 45.84397 47.62665 48.24976 41.87984 57.02672 52.92024
[29] 45.17118 59.52289 54.55879 50.22957 57.37075 57.26880 56.92949
[36] 56.13184 55.32351 51.62853 50.16422 49.71717 47.83176 50.75250
[43] 44.40762 65.01374 59.24777 45.26135 49.58269 49.20007 56.67979
[50] 51.49979 53.51991 51.82872 51.74278 60.21161 50.64537 61.09882
[57] 42.70748 55.50768 52.74313 53.29565 54.27784 48.98606 50.00076
[64] 45.88855 45.56925 53.82117 54.68926 52.31803 57.53360 64.30051
[71] 49.05381 38.14499 58.03443 47.74480 47.87195 58.15343 50.29136
[78] 44.67569 53.08782 51.16665 52.03459 54.31168 49.77604 55.86626
[85] 50.67708 53.99069 58.58103 54.61109 50.04441 58.89285 57.96102
[92] 55.29038 53.43239 48.23256 60.16391 48.39844 65.12400 61.19566
[99] 50.58580 45.84147
```



one sample t-test

Null hypothesis (H_0): The population mean = some value (μ_0)

Alternative hypothesis (H_1): The population mean ≠ μ_0 (two-sided), or > / < (one-sided)

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$$

$$df = n - 1$$

```
> t.test(x, mu = 50)

        One Sample t-test

data:  x
t = 4.8026, df = 99, p-value = 5.569e-06
alternative hypothesis: true mean is not equal to 50
95 percent confidence interval:
 51.74424 54.20026
sample estimates:
mean of x
 52.97225
```

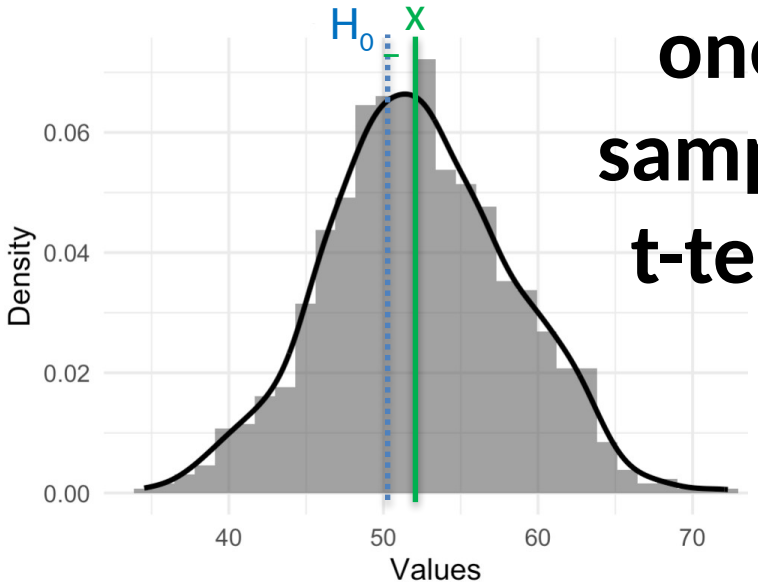
$\bar{x}$ = sample mean
μ_0 = hypothesized mean
s = sample standard deviation
n = sample size

```
> t<-(mean(x)-50)/(sd(x)/sqrt(length(x))) ##t
> t
[1] 4.802567
> df<-length(x)-1
> pval <- 2 * (1 - pt(abs(t), df))
> pval
[1] 5.568592e-06
```

```
> x
 [1] 48.63715 50.61894 61.35225 52.42305 52.77573 62.29039 54.76550
 [8] 44.40963 47.87888 49.32603 59.34449 54.15888 54.40463 52.66410
[15] 48.66495 62.72148 54.98710 40.20030 56.20814 49.16325 45.59306
[22] 50.69215 45.84397 47.62665 48.24976 41.87984 57.02672 52.92024
[29] 45.17118 59.52289 54.55879 50.22957 57.37075 57.26880 56.92949
[36] 56.13184 55.32351 51.62853 50.16422 49.71717 47.83176 50.75250
[43] 44.40762 65.01374 59.24777 45.26135 49.58269 49.20007 56.67979
[50] 51.49979 53.51991 51.82872 51.74278 60.21161 50.64537 61.09882
[57] 42.70748 55.50768 52.74313 53.29565 54.27784 48.98606 50.00076
[64] 45.88855 45.56925 53.82117 54.68926 52.31803 57.53360 64.30051
[71] 49.05381 38.14499 58.03443 47.74480 47.87195 58.15343 50.29136
[78] 44.67569 53.08782 51.16665 52.03459 54.31168 49.77604 55.86626
[85] 50.67708 53.99069 58.58103 54.61109 50.04441 58.89285 57.96102
[92] 55.29038 53.43239 48.23256 60.16391 48.39844 65.12400 61.19566
[99] 50.58580 45.84147
```

Null hypothesis (H_0): The population mean = some value (μ_0)

Alternative hypothesis (H_1): The population mean $\neq \mu_0$ (two-sided), or > / < (one-sided)



one sample t-test

```
> t.test(x, mu = 50)

        One Sample t-test

data:  x
t = 4.8026, df = 99, p-value = 5.569e-06
alternative hypothesis: true mean is not equal to 50
95 percent confidence interval:
 51.74424 54.20026
sample estimates:
mean of x
 52.97225
```

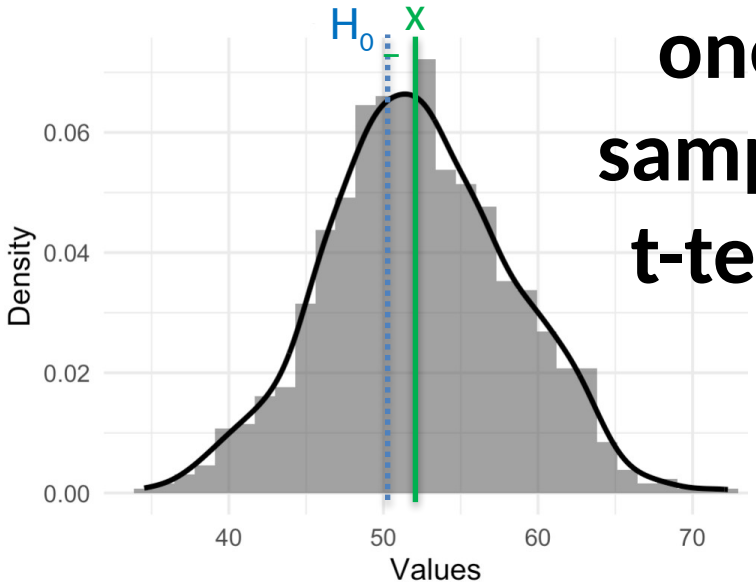
$$CI = \bar{x} \pm t^* \times SE$$

```
> alpha <- 0.05
> df <- length(x) - 1
> t_star <- qt(1 - alpha/2, df)   # two-tailed critical value
> mean(x)-sd(x)/sqrt(length(x))*t_star
[1] 51.74424
> mean(x)+sd(x)/sqrt(length(x))*t_star
[1] 54.20026
```

```
> x
 [1] 48.63715 50.61894 61.35225 52.42305 52.77573 62.29039 54.76550
 [8] 44.40963 47.87888 49.32603 59.34449 54.15888 54.40463 52.66410
[15] 48.66495 62.72148 54.98710 40.20030 56.20814 49.16325 45.59306
[22] 50.69215 45.84397 47.62665 48.24976 41.87984 57.02672 52.92024
[29] 45.17118 59.52289 54.55879 50.22957 57.37075 57.26880 56.92949
[36] 56.13184 55.32351 51.62853 50.16422 49.71717 47.83176 50.75250
[43] 44.40762 65.01374 59.24777 45.26135 49.58269 49.20007 56.67979
[50] 51.49979 53.51991 51.82872 51.74278 60.21161 50.64537 61.09882
[57] 42.70748 55.50768 52.74313 53.29565 54.27784 48.98606 50.00076
[64] 45.88855 45.56925 53.82117 54.68926 52.31803 57.53360 64.30051
[71] 49.05381 38.14499 58.03443 47.74480 47.87195 58.15343 50.29136
[78] 44.67569 53.08782 51.16665 52.03459 54.31168 49.77604 55.86626
[85] 50.67708 53.99069 58.58103 54.61109 50.04441 58.89285 57.96102
[92] 55.29038 53.43239 48.23256 60.16391 48.39844 65.12400 61.19566
[99] 50.58580 45.84147
```

Null hypothesis (H_0): The population mean = some value (μ_0)

Alternative hypothesis (H_1): The population mean $\neq \mu_0$ (two-sided), or > / < (one-sided)



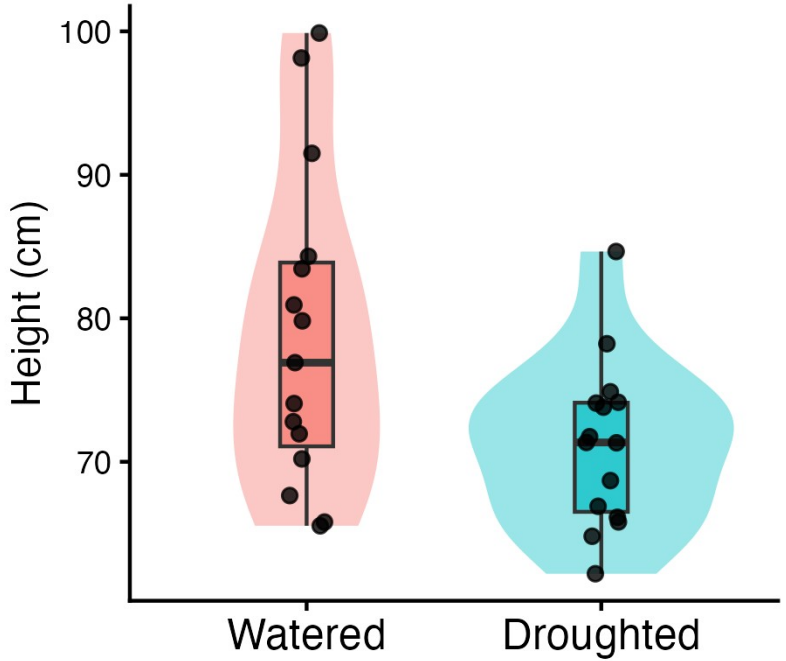
one sample t-test

```
> t.test(x, mu = 50)

        One Sample t-test

data:  x
t = 4.8026, df = 99, p-value = 5.569e-06
alternative hypothesis: true mean is not equal to 50
95 percent confidence interval:
 51.74424 54.20026
sample estimates:
mean of x
 52.97225
```

Watered vs. droughted

```
> set.seed(11)
> trial <- data.frame(
+   height = c(rnorm(15, 84, 12),
+              rnorm(15, 74, 12)),
+   treatment = rep(c("Watered", "Droughted"),
+                   each = 15))
> head(trial, 4)
    height treatment
1 76.90763   Watered
2 84.31913   Watered
3 65.80136   Watered
4 67.64816   Watered
```

Is the difference real, or could it be **chance** from sampling 15 plants each?



Test statistic = signal / noise

$$t = \frac{\text{signal}}{\text{noise}} = \frac{\bar{x}_1 - \bar{x}_2}{SE_{\text{diff}}}$$

$$SE_{\text{diff}} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

t by hand

```
> w <- trial$height[trial$treatment == "Watered"]
> d <- trial$height[trial$treatment == "Droughted"]
> signal <- mean(w) - mean(d)
> noise  <- sqrt(var(w)/15 + var(d)/15)
> signal
[1] 7.612456
> noise
[1] 3.20656
> signal / noise
[1] 2.374026
```

$$t = \frac{78.86 - 71.25}{3.21} = 2.37$$

The difference is **2.4 SEs** from zero

t.test()

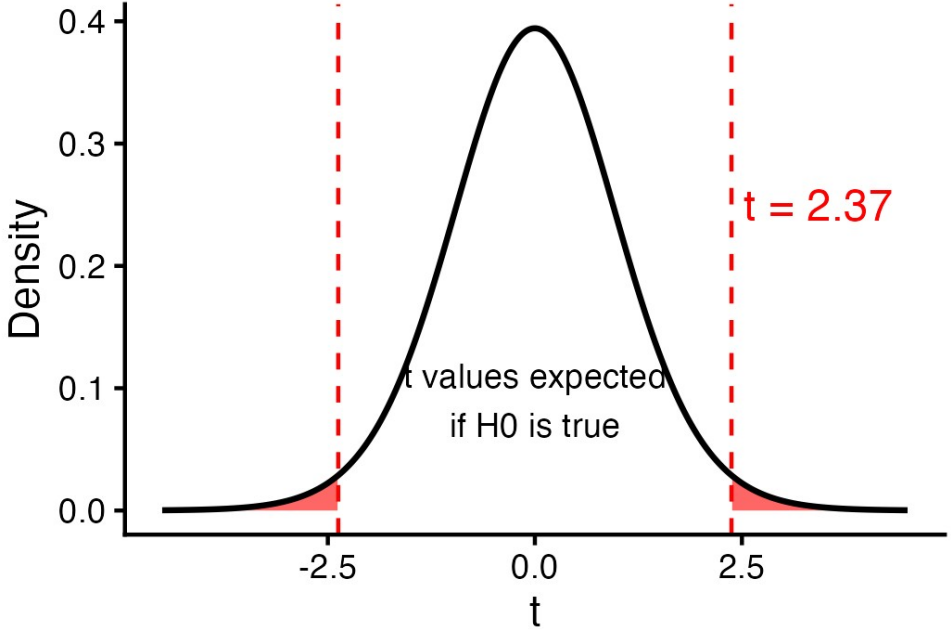
```
> t.test(height ~ treatment, data = trial)

        Welch Two Sample t-test

data:  height by treatment
t = 2.374, df = 21.223, p-value = 0.0271
alternative hypothesis: true difference in means between group
Watered and group Droughted is not equal to 0
95 percent confidence interval:
  0.9483139 14.2765983
sample estimates:
  mean in group Watered mean in group Droughted
               78.86303                71.25057
```

Same t = 2.374. Welch's test (the default) does not assume equal variances; df = 21.2.

Alternate syntax: two vectors; Student's t

```
> t.test(w, d)$statistic
       t
2.374026
> t.test(w, d, var.equal = TRUE)$p.value
[1] 0.02468965
```

From t to p



$$p = P\left(|T| \geq |t_{\text{obs}}| \mid H_0 \text{ true}\right)$$

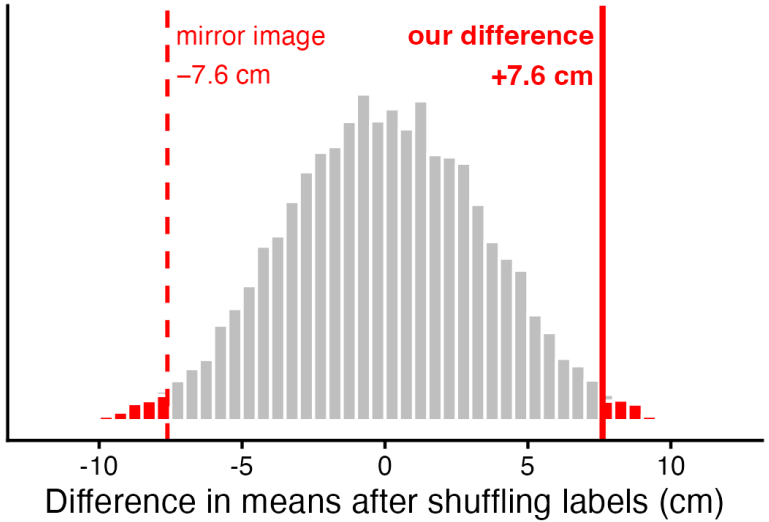
```
> t_obs <- signal / noise
> df    <- t.test(height ~ treatment,
+                 data = trial)$parameter
> 2 * pt(-abs(t_obs), df)   # both tails
[1] 0.02710185
```

The p-value is the **red area**: how often chance alone gives a t this extreme.

A p-value by shuffling labels

a permutation test

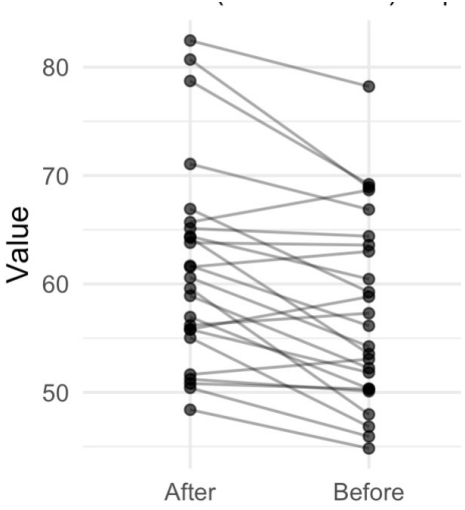
```
> set.seed(3)
> observed <- mean(w) - mean(d)
> null_diffs <- replicate(10000, {
+   shuffled <- sample(trial$treatment)   # no replacement
+   mean(trial$height[shuffled == "Watered"]) -
+     mean(trial$height[shuffled == "Droughted"])
+ })
> mean(abs(null_diffs) >= abs(observed))  # either direction
[1] 0.0258
```



- If drought did nothing, the labels are arbitrary. Shuffle them 10,000 times: **red** = shuffles at least as extreme as ours, **either direction**: p = **0.026** (t-test: 0.027)
- Not a bootstrap: shuffling (**no replacement**) builds the null for a p-value; a bootstrap resamples **with replacement** to get an SE or CI

Paired t-test

```
> df
      after   before
1  65.70457 68.65839
2  56.94951 50.33579
3  59.58114 47.96439
4  56.18829 57.29763
5  63.80180 63.58607
6  55.03589 46.83742
7  82.45742 78.21918
8  55.79342 51.82677
9  66.91447 59.24740
10 50.41310 45.92759
11 61.68081 56.14247
12 61.56274 63.00493
13 64.39138 60.44929
14 55.81577 58.82193
15 51.21642 50.10190
```




```
> df<-data.frame(after, before)
> tt_paired <- t.test(df$after, df$before, paired = TRUE)
> tt_paired

        Paired t-test

data:  df$after and df$before
t = 4.5693, df = 24, p-value = 0.0001242
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 2.231368 5.907615
sample estimates:
mean difference
       4.069491
```

(Same as one sample t-test on difference)

```
> t.test(df$after- df$before)

        One Sample t-test

data:  df$after - df$before
t = 4.5693, df = 24, p-value = 0.0001242
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 2.231368 5.907615
sample estimates:
mean of x
 4.069491
```

Live poll

Multiple choice

Watered vs. droughted: p = 0.027. What does that mean?

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-pvalue

host screen

What a p-value is (and isn't)

✓ P(a difference at least this large | H_0 true)

✗ the probability the hypothesis is true
✗ the probability the result was luck
✗ a measure of how big the effect is

Type I and Type II errors

	$\mathbf{H_0}$ **true** (no effect)	$\mathbf{H_0}$ **false** (real effect)
Reject $\mathbf{H_0}$	Type I error: false positive (α)	Correct: power $= 1 - \beta$
Don't reject	Correct	Type II error: false negative (β)

- **α = 0.05** is the false-positive rate we agree to accept
- **Power** = the chance of detecting an effect that is really there

Live poll

Number: live histogram

Guess: plants per group needed to detect a 10 cm drought effect 80% of the time (SD = 12 cm)?

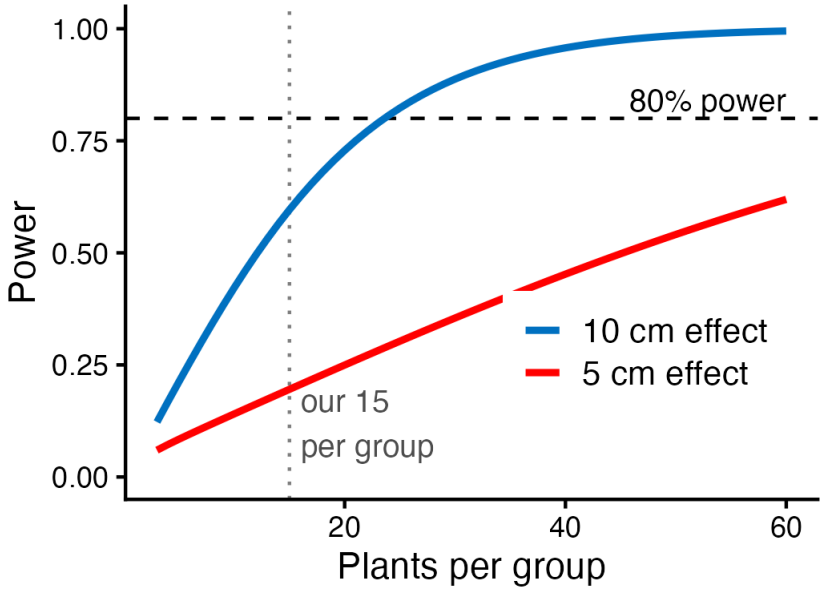
Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-power

host screen

Power and sample size

```
> # n per group for a 10 cm effect?
> power.t.test(delta = 10, sd = 12, power = 0.8)$n
[1] 23.60472
> # power with 15 per group
> power.t.test(n = 15, delta = 10, sd = 12)$power
[1] 0.596191
```

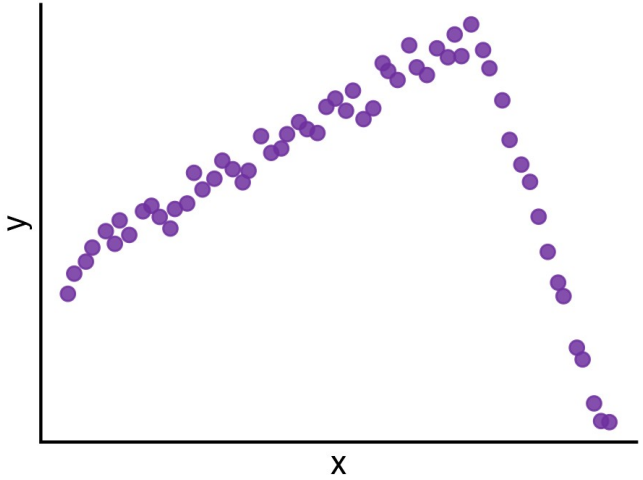


- To detect 10 cm with 80% power: **24 plants per group**
- Our 15 per group: only **60%** power
- Halve the effect: you need ~4× the plants

Live poll

Number: live histogram

Guess r for this scatterplot



Scan with your phone

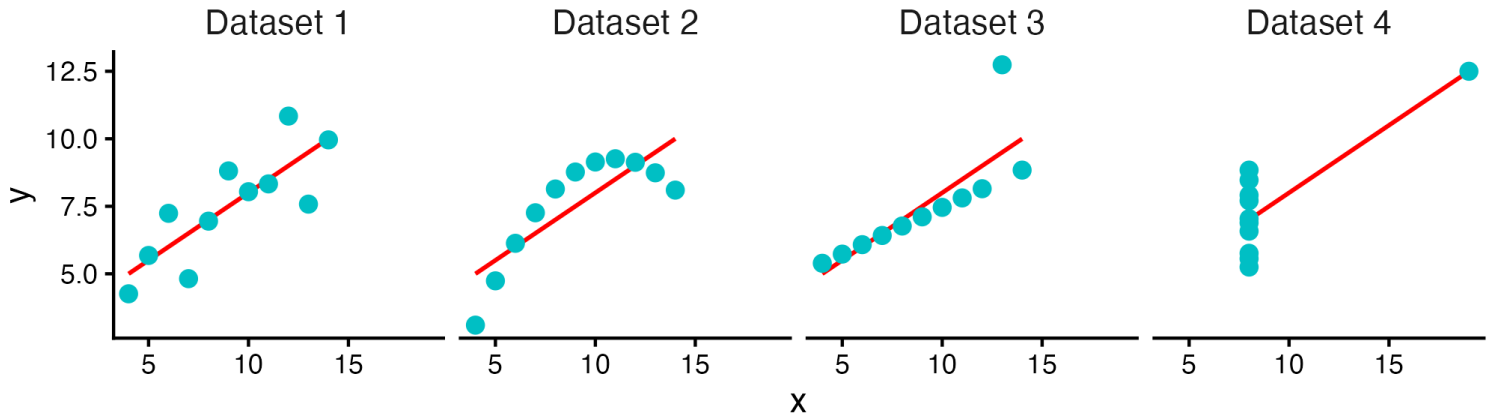
Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-guessr

host screen

Always plot your data

```
> sapply(1:4, function(i) {
+   x <- anscombe[, i]
+   y <- anscombe[, i + 4]
+   round(c(mean_y = mean(y), sd_y = sd(y),
+           r = cor(x, y)), 2)
+ })
       [,1] [,2] [,3] [,4]
mean_y 7.50 7.50 7.50 7.50
sd_y   2.03 2.03 2.03 2.03
r      0.82 0.82 0.82 0.82
```

Four datasets, identical mean, SD, and correlation …



Kahoot-style showdown

End-of-lecture game

12 questions on today's lecture. Fast and correct wins.

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w1-showdown

host screen

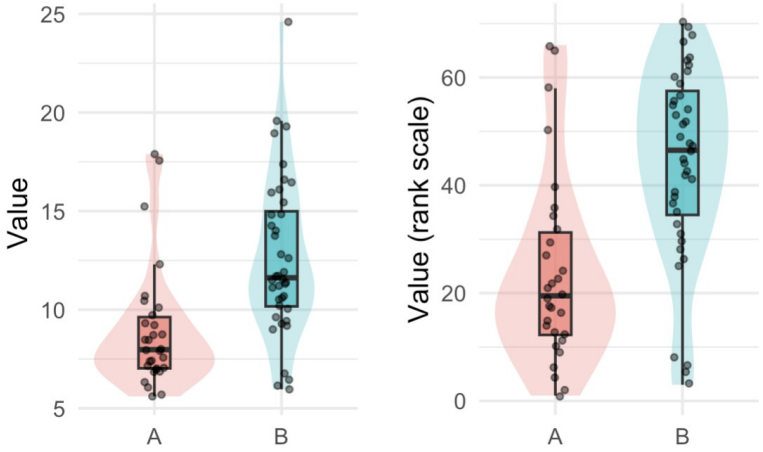
Part 2: The toolkit of tests

Wednesday, September 30

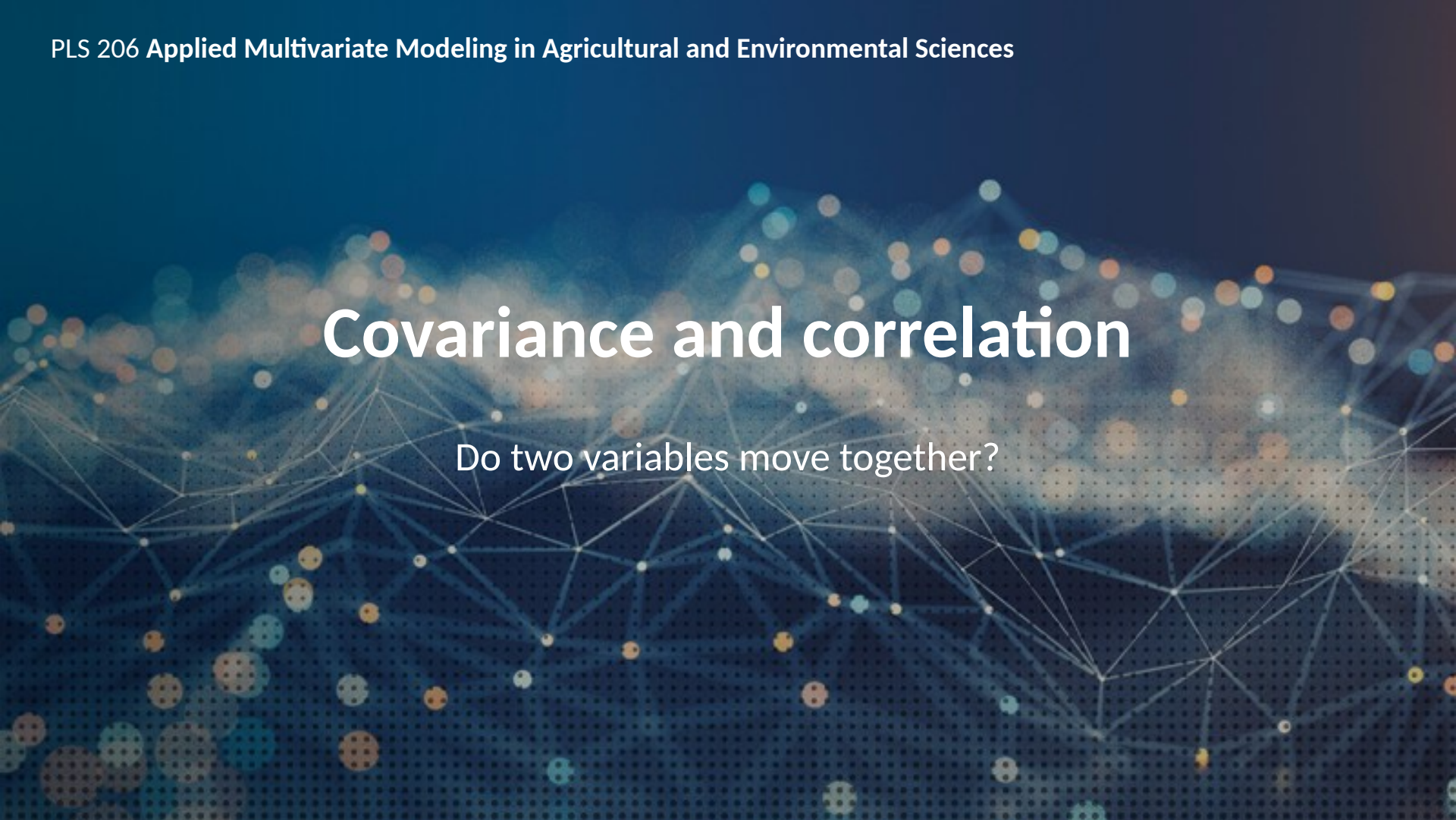
Wilcoxon test

(“*non parametric t-test*”)

```
> df_wilx
   group      value
1      B   5.960864
2      A   5.607224
3      B   6.446158
4      B   9.428198
5      A   7.946408
6      B 11.691042
7      A   9.315445
8      B 16.458500
9      B 10.593346
10     A   8.711629
11     A   6.969304
12     B 10.044467
13     B 16.091317
14     A   8.001500
```




```
> wx_rs <- wilcox.test(value ~ group, data = df_wilx)
> wx_rs

        Wilcoxon rank sum exact test

data:  value by group
W = 249, p-value = 1.596e-05
alternative hypothesis: true location shift is not equal to 0
```

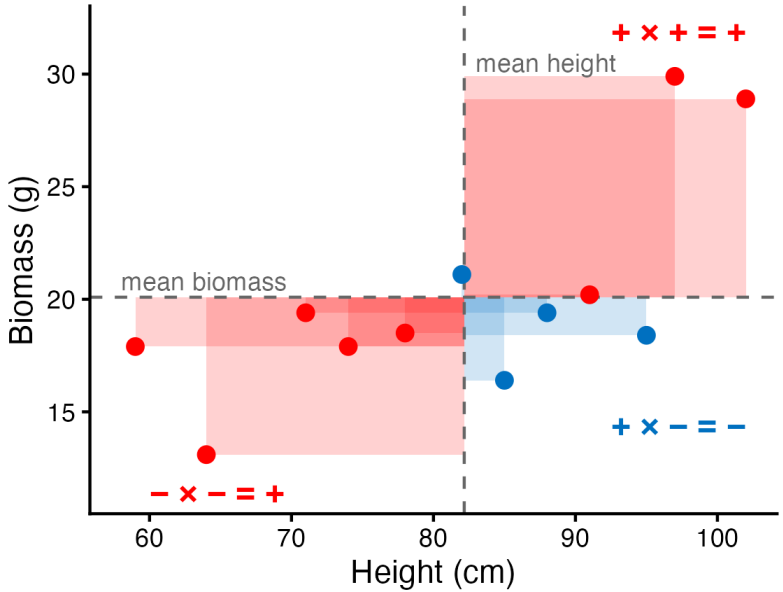


Covariance and correlation

Do two variables move together?

Do height and biomass move together?

```
> plants <- data.frame(   # the same 12 plants, now with biomass
+   height  = c(78, 95, 64, 88, 102, 71,
+              85, 91, 59, 97, 82, 74),          # cm
+   biomass = c(18.5, 18.4, 13.1, 19.4, 28.9, 19.4, 16.4, 20.2,
+              17.9, 29.9, 21.1, 17.9))          # g
```



- Each rectangle: **$(x - \bar{x})(y - \bar{y})$**
- **Red**: both above or both below the mean → positive
- **Blue**: one above, one below → negative
- **Covariance** ≈ the average rectangle.

Mostly red here, so positive.

Covariance, step by step

$$\operatorname{cov}(x, y) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})\,(y_i - \bar{y})$$

$$\operatorname{cov}(x, x) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 = s_x^2$$

```
> dx <- plants$height  - mean(plants$height)
> dy <- plants$biomass - mean(plants$biomass)
> sum(dx * dy) / (12 - 1)     # covariance by hand
[1] 44.98333
> cov(plants$height, plants$biomass)
[1] 44.98333
> cov(plants$height, plants$height)   # = var(height)
[1] 179.4242
> var(plants$height)
[1] 179.4242
```

- Variance is the covariance of a variable **with itself**
- **+** together, **−** opposite, **≈ 0** no linear trend
- **Units: cm × g**

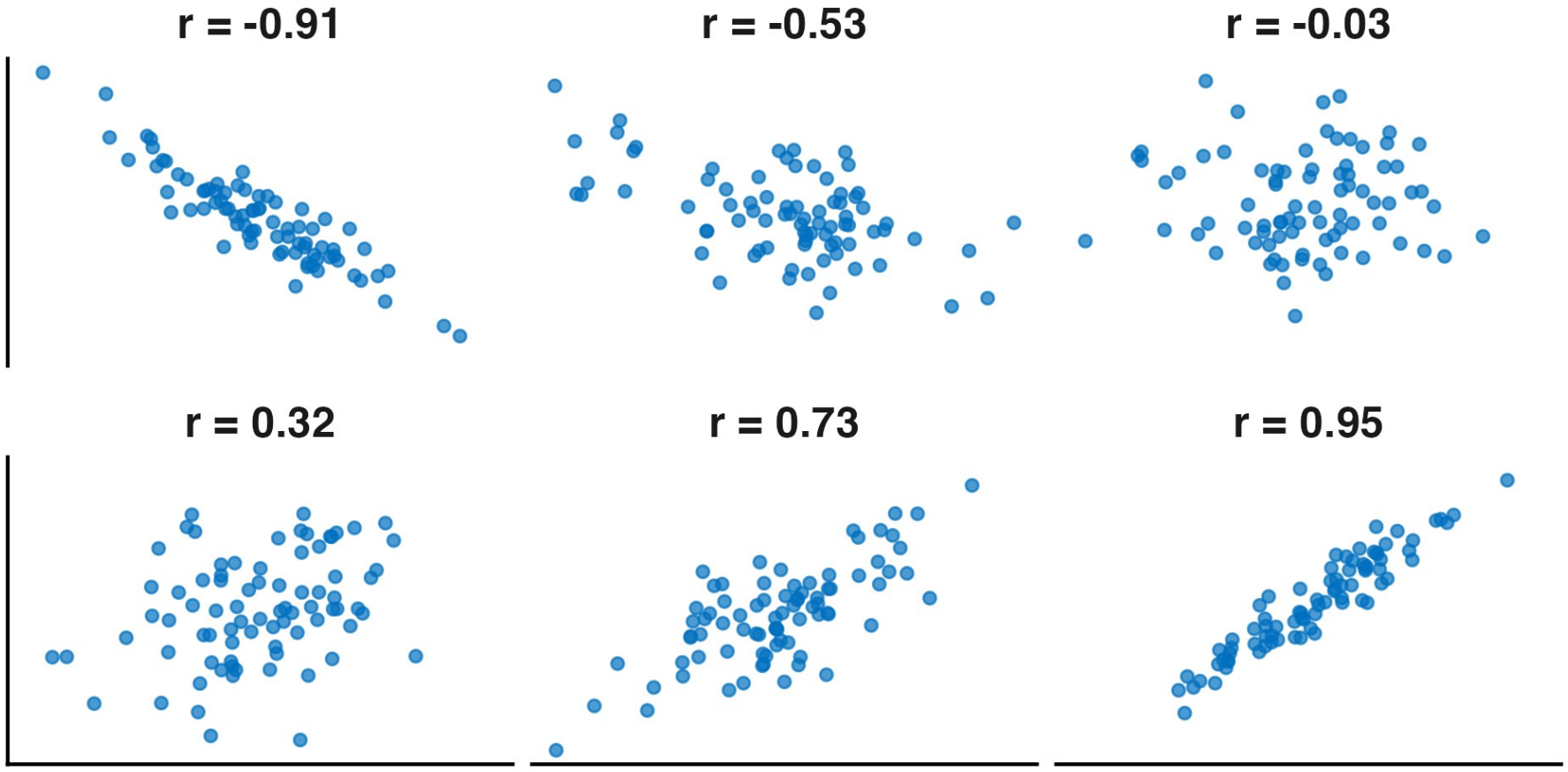
From covariance to correlation

```
> cov(plants$height, plants$biomass)        # cm x g
[1] 44.98333
> cov(plants$height / 100, plants$biomass)  # m x g: 100x less
[1] 0.4498333
> cor(plants$height, plants$biomass)
[1] 0.6998406
> cor(plants$height / 100, plants$biomass)  # r has no units
[1] 0.6998406
> cov(plants$height, plants$biomass) /
+   (sd(plants$height) * sd(plants$biomass))
[1] 0.6998406
> cov(scale(plants$height), scale(plants$biomass))
          [,1]
[1,] 0.6998406
```

$$r = \frac{\mathrm{cov}(x, y)}{s_x \, s_y} = \frac{44.98}{13.39 \times 4.80} = 0.70$$

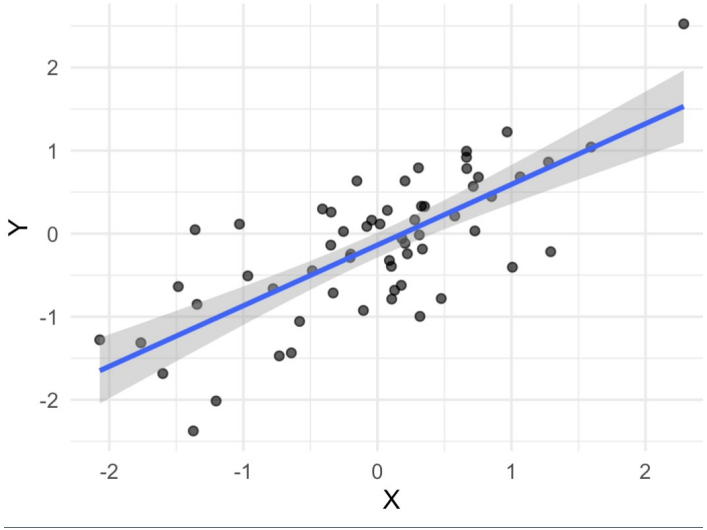
- Covariance changes with the units: cm → m makes it **100× smaller**
- Divide by both SDs: **r** has no units, −1 to +1
- **r = covariance of z-scores** (Monday's scale())

What r looks like



Pearson correlation

```
> df
           xC          yC
1  -0.34778693 -0.13858920
2   0.66671889  0.78298571
3   0.66478866  0.99223687
4  -1.60079686 -1.68439178
5  -0.20319713 -0.28725030
6  -0.15295493  0.63394353
7   0.07441540  0.28013303
8   1.59382310  1.04195056
9  -2.07130493 -1.27841449
10  0.20650953 -0.11749842
11 -0.10440737 -0.92468670
12 -1.48556503 -0.63764525
13 -0.19935287 -0.24739595
14 -1.02915229  0.11441361
```




```
> ct_pear <- cor.test(xC, yC, method = "pearson")
> ct_pear

        Pearson's product-moment correlation

data:  xC and yC
t = 8.2438, df = 58, p-value = 2.393e-11
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5908006 0.8330748
sample estimates:
      cor
0.7345317
```

Testing r

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} \qquad df = n-2$$

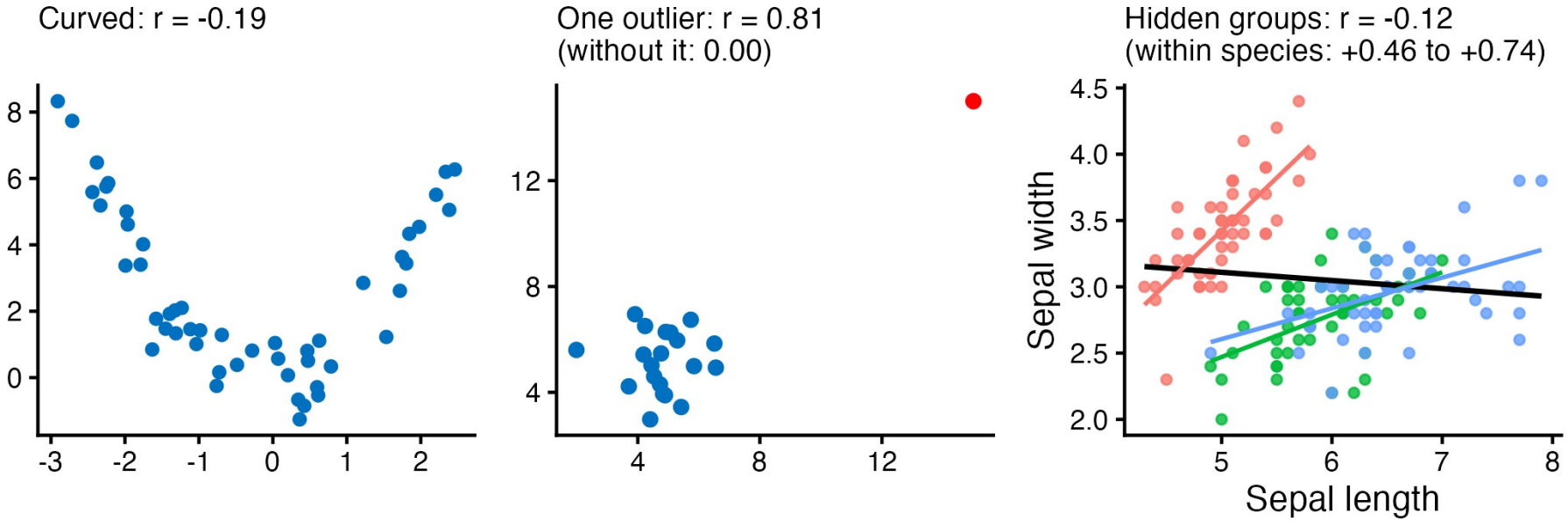
$$t = \frac{0.7345\sqrt{58}}{\sqrt{1-0.7345^2}} = 8.24$$

The t and df on the previous slide

```
> r <- 0.30
> for (n in c(10, 30, 100)) {
+   t <- r * sqrt(n - 2) / sqrt(1 - r^2)
+   p <- 2 * pt(-abs(t), df = n - 2)
+   cat("n =", n, " t =", round(t, 2), " p =", signif(p, 2),
"\n")
+ }
n = 10  t = 0.89  p = 0.4
n = 30  t = 1.66  p = 0.11
n = 100  t = 3.11  p = 0.0024
```

- **H_0: ρ = 0** (no linear association)
- Same r = 0.30: p = 0.4 with 10 plants, **0.002** with 100
- r^2 = shared variance: **r = 0.70 → 49%** of biomass variation goes with height

Correlation pitfalls: always plot



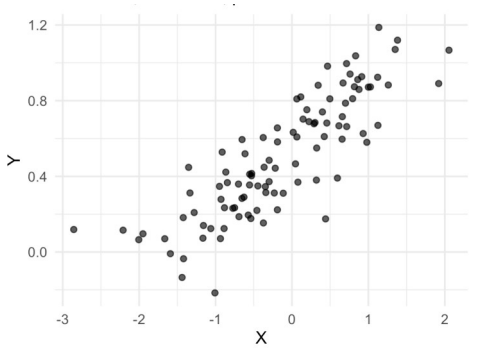
- r only measures **straight-line** association: a perfect curve can give r ≈ 0
- One point can create a correlation; groups can reverse its sign. And correlation is not causation.

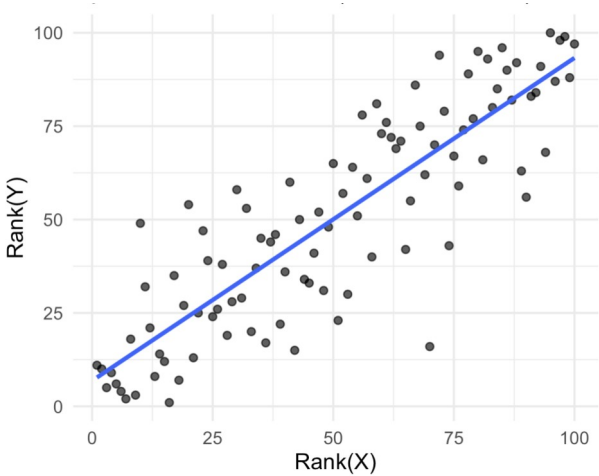
Spearman correlation (non-parametric)

```
> cor_spear <- cor.test(df_cor_s$x, df_cor_s$y, method = "spearman")
> cor_spear

        Spearman's rank correlation rho

data:  df_cor_s$x and df_cor_s$y
S = 22610, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.8643264
```

```
> df_cor_s
             x           y rank_x rank_y
1   -0.65299176  0.594404400     30     58
2    1.35050938  1.070701180     97     98
3   -0.52991148  0.402799808     37     44
4   -0.92826260  0.278389803     19     27
5   -0.36435588  0.448172637     43     50
6    0.14544443  0.702384925     60     73
7    0.30126605  0.684225086     64     71
8   -2.85561802  0.118934330      1     11
9    1.13652272  1.187260760     95    100
10  -0.65905478  0.715439229     77     74
```

Ranked >>>





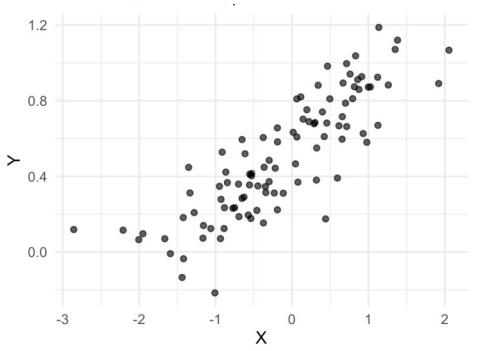
Spearman correlation (non-parametric)

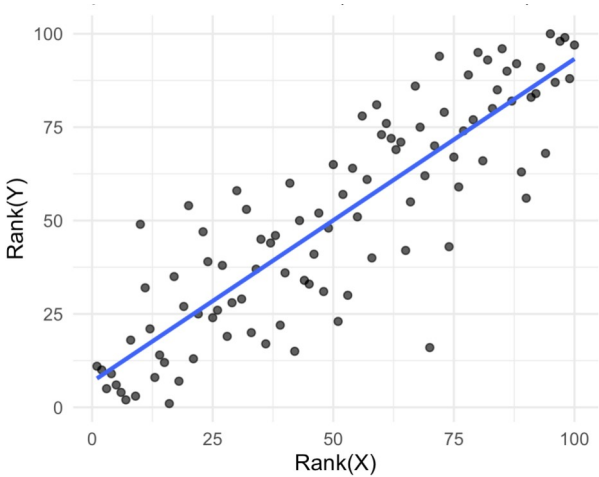
```
> cor_spear <- cor.test(df_cor_s$x, df_cor_s$y, method = "spearman")
> cor_spear

        Spearman's rank correlation rho

data:  df_cor_s$x and df_cor_s$y
S = 22610, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.8643264
```

```
> cor.test(df_cor_s$rank_x, df_cor_s$rank_y, method = "pearson")

        Pearson's product-moment correlation

data:  df_cor_s$rank_x and df_cor_s$rank_y
t = 17.013, df = 98, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8044853 0.9067980
sample estimates:
      cor
0.8643264
```

```
> df_cor_s
             x           y rank_x rank_y
1  -0.65299176  0.594404400     30     58
2   1.35050938  1.070701180     97     98
3  -0.52991148  0.402799808     37     44
4  -0.92826260  0.278389803     19     27
5  -0.36435588  0.448172637     43     50
6   0.14544443  0.702384925     60     73
7   0.30126605  0.684225086     64     71
8  -2.85561802  0.118934330      1     11
9   1.13652272  1.187260760     95    100
10 -0.65905478  0.715439229     77     74
```

Ranked >>>





The covariance matrix

$$\mathbf{S} = \begin{pmatrix} s_1^2 & s_{12} & s_{13} \\ s_{12} & s_2^2 & s_{23} \\ s_{13} & s_{23} & s_3^2 \end{pmatrix}$$

- **Diagonal**: each variable's variance
- **Off-diagonal**: covariance of each pair; symmetric ($s_{12} = s_{21}$)
- p variables → p × p matrix

```
> traits <- iris[, 1:4]
> S <- cov(traits)          # covariance matrix
> round(S, 2)
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length         0.69       -0.04         1.27        0.52
Sepal.Width         -0.04        0.19        -0.33       -0.12
Petal.Length         1.27       -0.33         3.12        1.30
Petal.Width          0.52       -0.12         1.30        0.58
> var(traits$Petal.Length)  # the diagonal = variances
[1] 3.116278
```

Covariance matrix → correlation matrix

$$r_{jk} = \frac{s_{jk}}{s_j \, s_k}$$

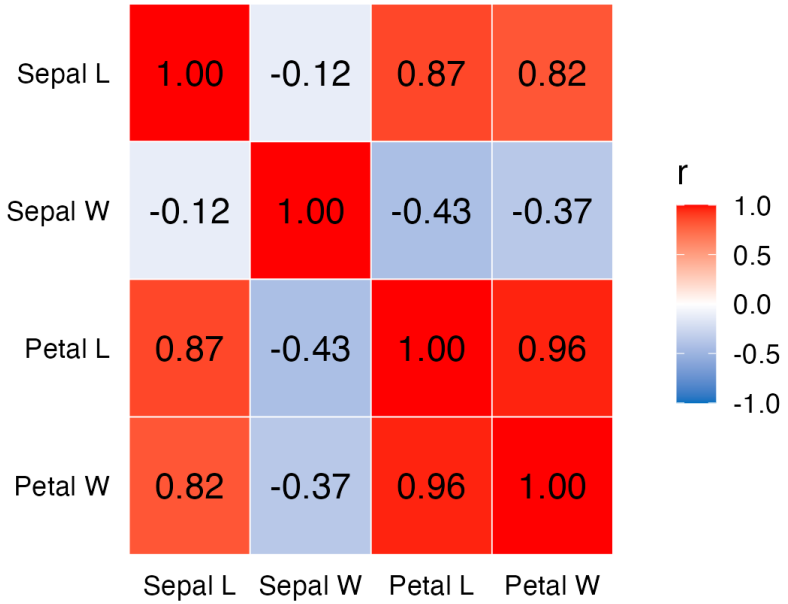
Divide each cell by both SDs. The diagonal becomes 1.

```
> diag(S)                    # variances: very different sizes
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.6856935    0.1899794    3.1162779    0.5810063
> R <- cov2cor(S)            # divide each cell by both SDs
> all.equal(R, cor(traits))
[1] TRUE
> all.equal(cov(scale(traits)), cor(traits))
[1] TRUE
```

- Petal length has **16× the variance** of sepal width: in S, big-variance traits dominate
- **cor = cov of standardized data**: every trait counts equally. PCA on S vs. R (Week 7)

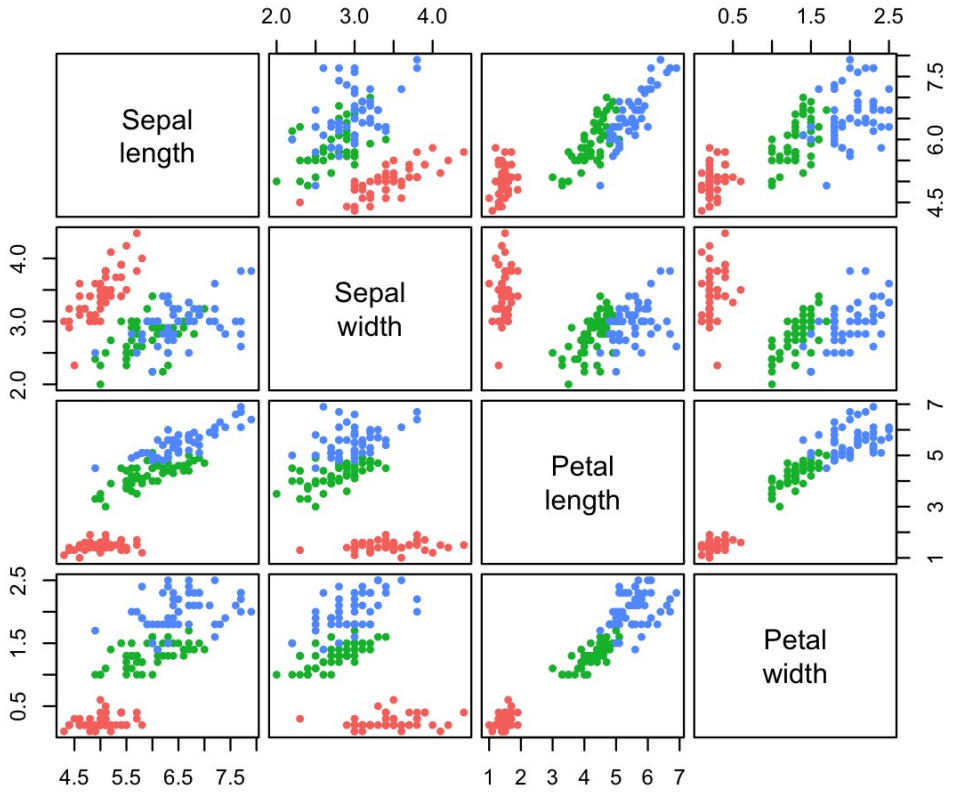
Many variables: the correlation matrix

```
> traits <- iris[, 1:4]     # 150 flowers, 4 traits
> round(cor(traits), 2)
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length         1.00       -0.12         0.87        0.82
Sepal.Width         -0.12        1.00        -0.43       -0.37
Petal.Length         0.87       -0.43         1.00        0.96
Petal.Width          0.82       -0.37         0.96        1.00
```



- **cor()** on a data frame: every pair at once
- Diagonal = 1; the matrix is symmetric
- **cov()** gives the covariance matrix: the starting point for **PCA (Week 7)**

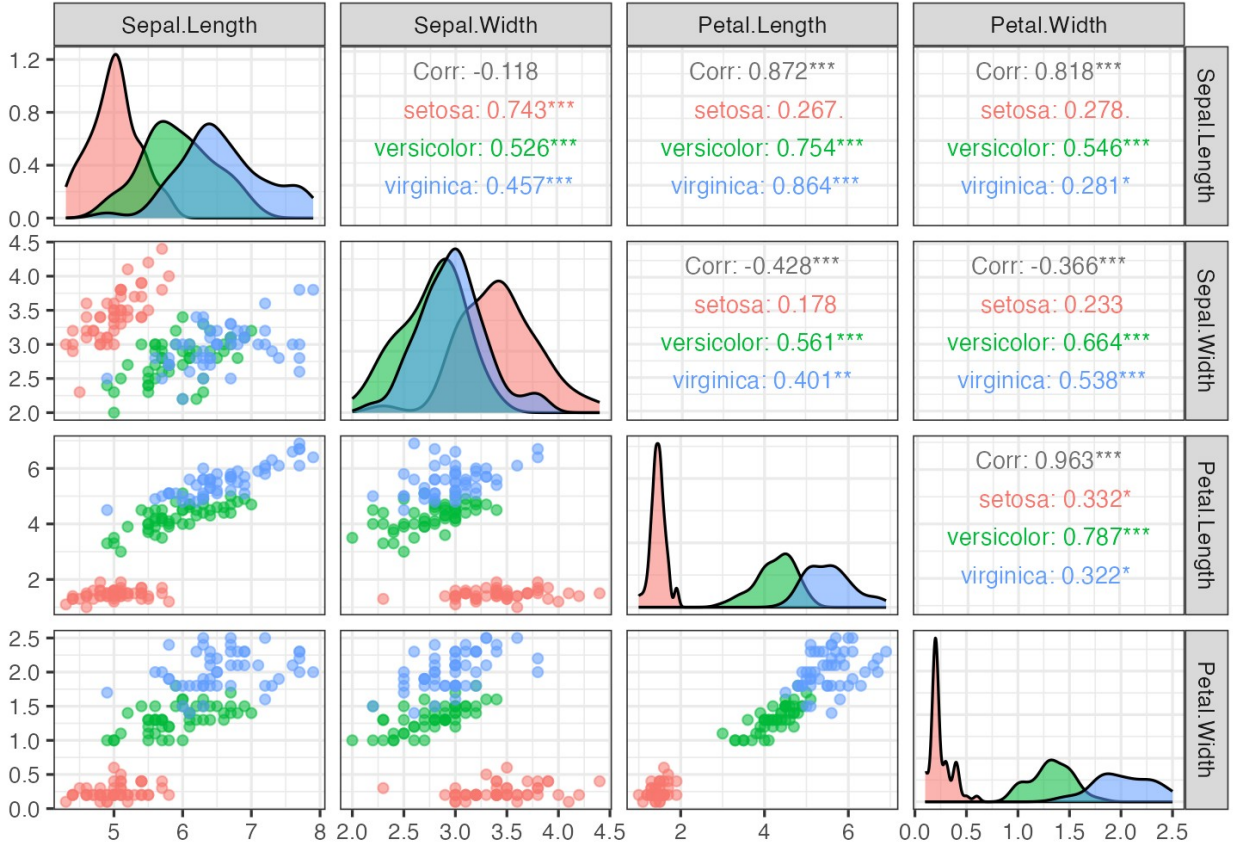
Scatterplot matrix: pairs()




```
> pairs(traits, col = iris$Species, pch = 19)
```

- Every pair in one figure: the plots behind the matrix
- Shows what r hides: **groups, curves, outliers**
- Base R, no packages

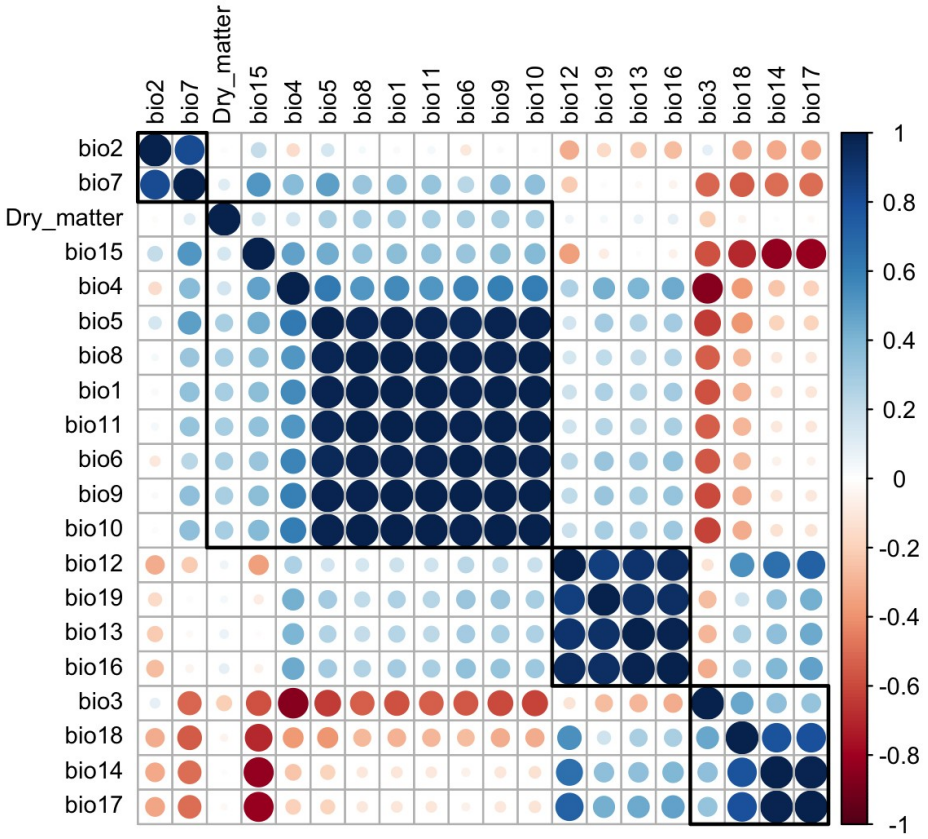
ggpairs(): plots, densities, and r together




```
> library(GGally)
> ggpairs(iris, columns = 1:4,
+         aes(colour = Species, alpha = 0.5))
```

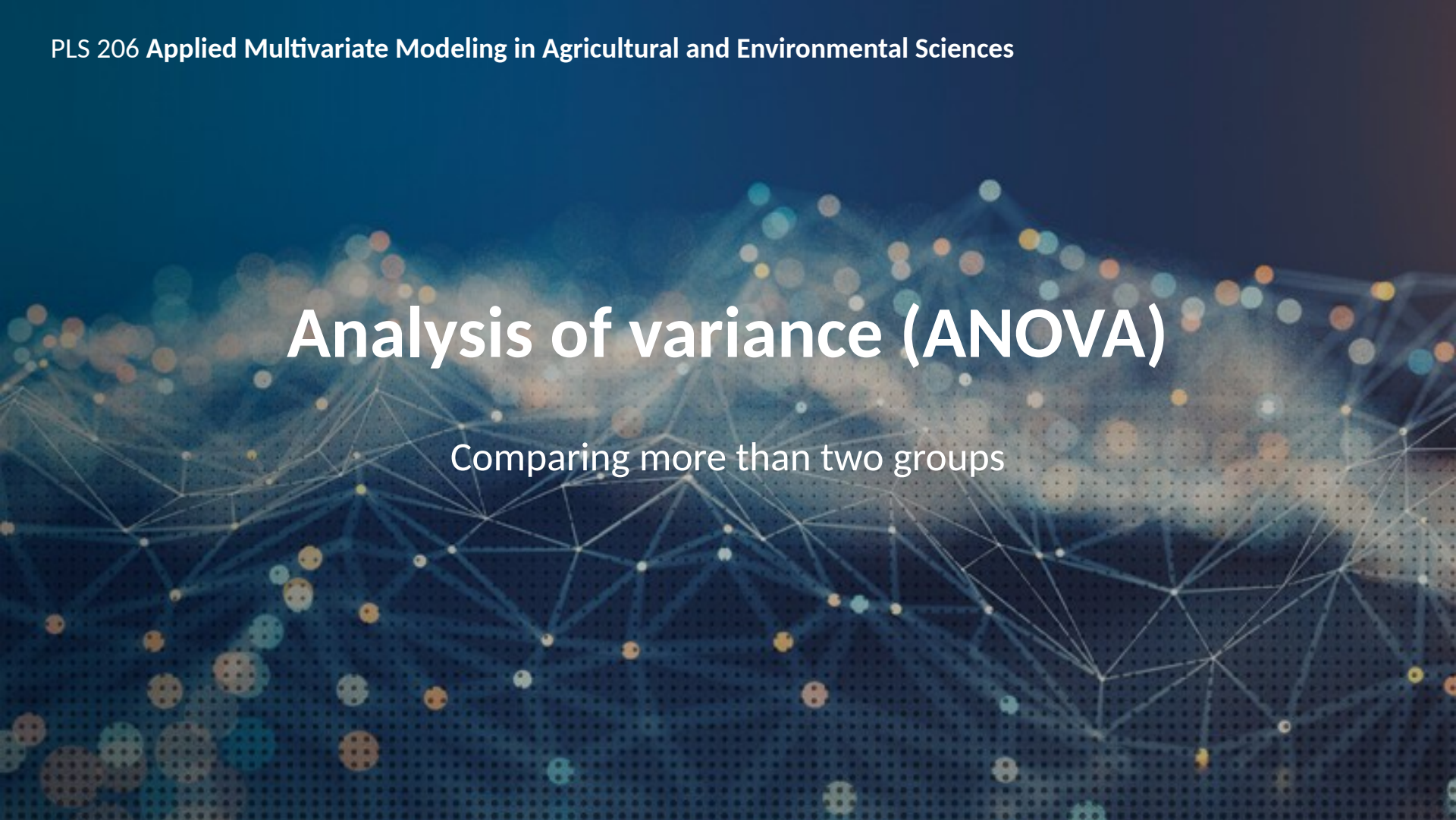
- Lower: scatterplots; diagonal: distributions; upper: **r overall and per group**
- You'll use it in **Problem Set 1**

corrplot(): many variables at once




```
> library(corrplot)
> cassava <- read.csv("data/cassava.csv")  # 1006 landraces
> dim(cassava)
[1] 1006   20
> corrplot(cor(cassava), order = "hclust", addrect = 4)
```

- 20 variables = **190 pairs**: too many for a scatterplot matrix
- **order = "hclust"** groups similar variables: **blocks** of redundant climate variables
- 32 pairs have |r| > 0.8. That redundancy is what **PCA (Week 7)** and **clustering (Week 8)** use



Analysis of variance (ANOVA)

Comparing more than two groups

Live poll

Number: live histogram (%)

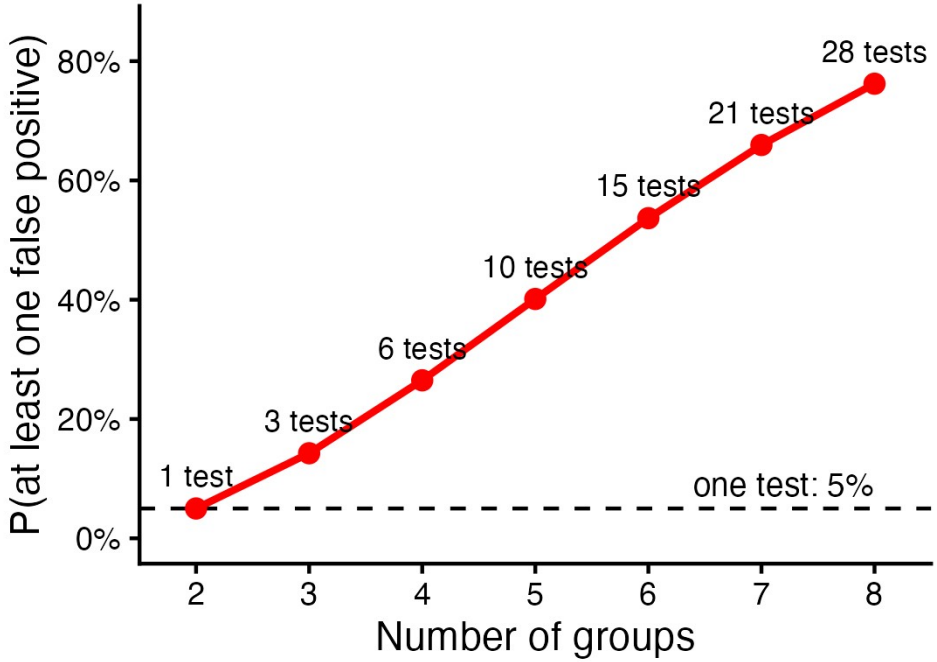
5 treatments that all do nothing. You t-test every pair: 10 tests. Chance that at least one gives p < 0.05?

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w1-familywise

host screen

Why not just run lots of t-tests?

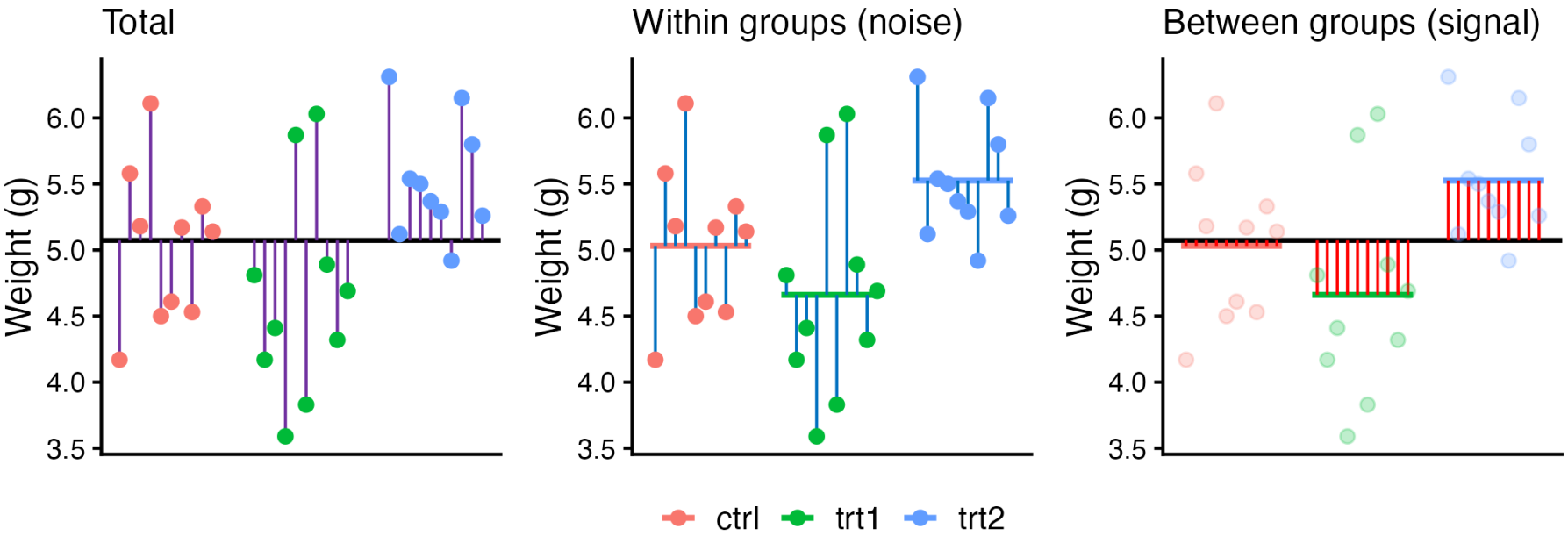


$$P(\text{at least one false positive}) = 1 - (1 - \alpha)^m$$

- 3 groups: 3 tests, **14%**
- 5 groups: 10 tests, **40%**
- ANOVA asks **one** question first: do any of the means differ?

$$H_0: \ \mu_{\text{ctrl}} = \mu_{\text{trt1}} = \mu_{\text{trt2}} \qquad H_A: \ \text{at least one differs}$$

Splitting the variation



$$\underbrace{SS_{\text{total}}}_{14.26} = \underbrace{SS_{\text{between}}}_{3.77} + \underbrace{SS_{\text{within}}}_{10.49}$$

Sums of squares by hand

$$SS_{\text{total}} \quad \sum_j \sum_i (y_{ij} - \bar{y})^2$$

$$SS_{\text{between}} \quad \sum_j n_j (\bar{y}_j - \bar{y})^2$$

$$SS_{\text{within}} \quad \sum_j \sum_i (y_{ij} - \bar{y}_j)^2$$

```
> pg    <- PlantGrowth
> grand <- mean(pg$weight)
> gmean <- ave(pg$weight, pg$group)  # group means
> sum((pg$weight - grand)^2)  # total
[1] 14.25843
> sum((gmean - grand)^2)      # between
[1] 3.76634
> sum((pg$weight - gmean)^2)  # within
[1] 10.49209
```

- y_{ij} = plant i in group j; $\bar{y}_j$ = group mean; $\bar{y}$ = grand mean
- Groups differ when **between** is large relative to **within**

F = signal / noise

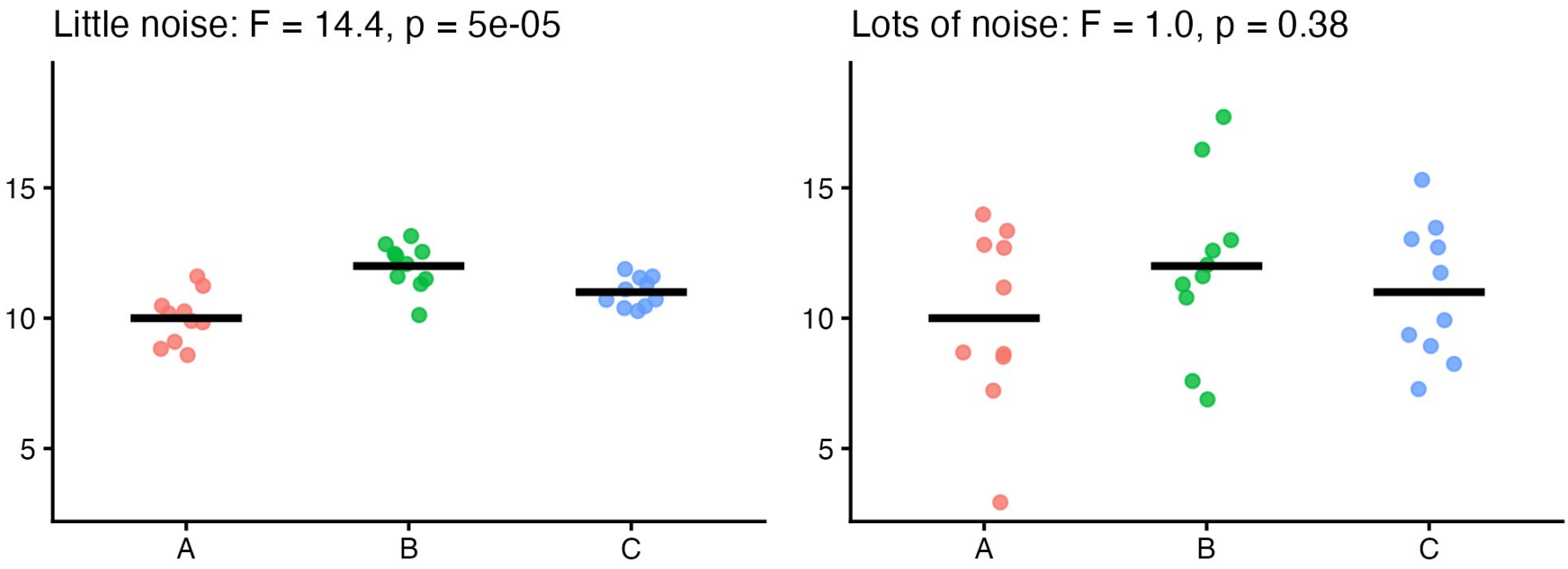
$$F = \frac{MS_{\text{between}}}{MS_{\text{within}}} = \frac{SS_{\text{between}}/(k-1)}{SS_{\text{within}}/(N-k)}$$

$$F = \frac{3.766/2}{10.492/27} = \frac{1.883}{0.389} = 4.85$$

```
> ms_group <- 3.766 / (3 - 1)   # k - 1 = 2 df
> ms_resid <- 10.492 / (30 - 3) # N - k = 27 df
> ms_group / ms_resid           # F
[1] 4.845692
> pf(4.846, df1 = 2, df2 = 27, lower.tail = FALSE)
[1] 0.01591099
```

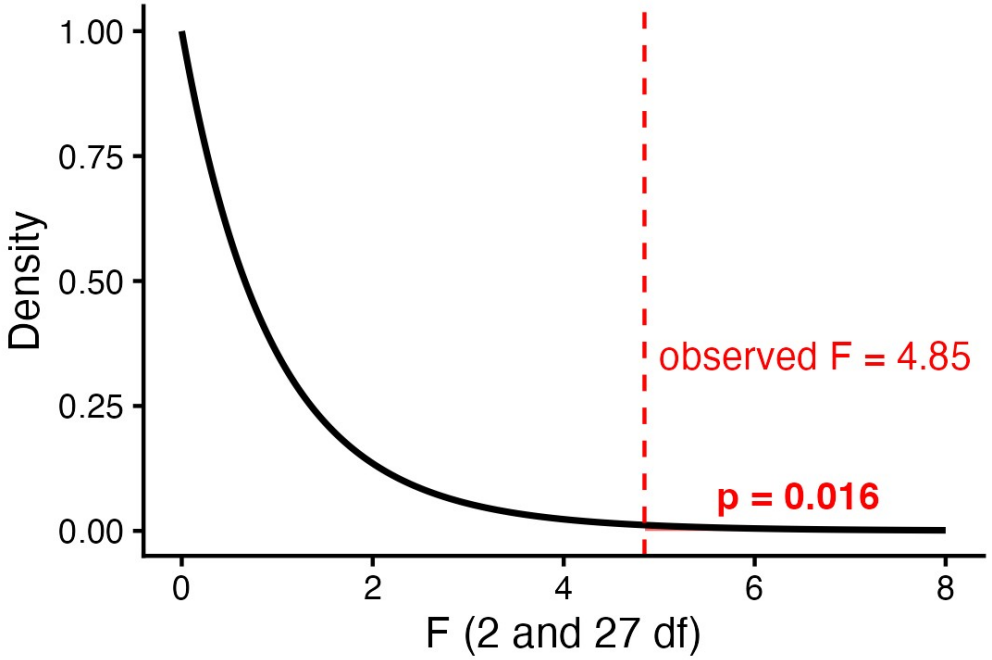
- **Mean square** = SS / df: a variance
- k = 3 groups, N = 30 plants
- H_0 true: F ≈ **1**
- **Big F: the means differ more than noise would make them**

Same means, different noise



Group means are identical (10, 12, 11) in both panels. **F compares the spread of the means to the spread within groups.**

From F to p



- The F distribution has **two** df: k − 1 = 2 and N − k = 27
- Only big F counts as evidence: **one-tailed**
- **p = 0.016:** at least one mean differs. ANOVA doesn't say which.

Reading the ANOVA table

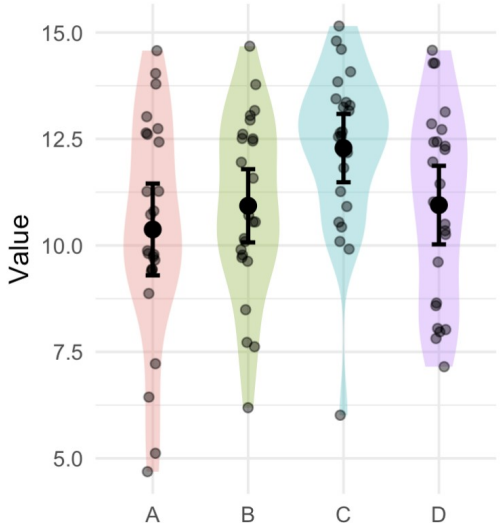
	Df	Sum Sq	Mean Sq	F	p
group (between)	$k-1=2$	3.77	$3.77/2=1.88$	$1.88/0.389=4.85$	0.016
Residuals (within)	$N-k=27$	10.49	$10.49/27=0.389$		
Total	29	14.26			

```
> fit <- aov(weight ~ group, data = PlantGrowth)
> summary(fit)
            Df Sum Sq Mean Sq F value Pr(>F)
group        2  3.766  1.8832   4.846 0.0159 *
Residuals   27 10.492  0.3886
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- Every column is a step we just did by hand
- **Pr(>F)** is the p-value

One way analysis of variance (ANOVA)

```
> df_aov
    group      value
1       A 13.790387
2       D  9.609565
3       C 11.265531
4       C 12.370461
5       A  9.442422
6       B 10.160362
7       C 14.605085
8       B 13.050642
9       B 10.934007
10      A  9.874572
```



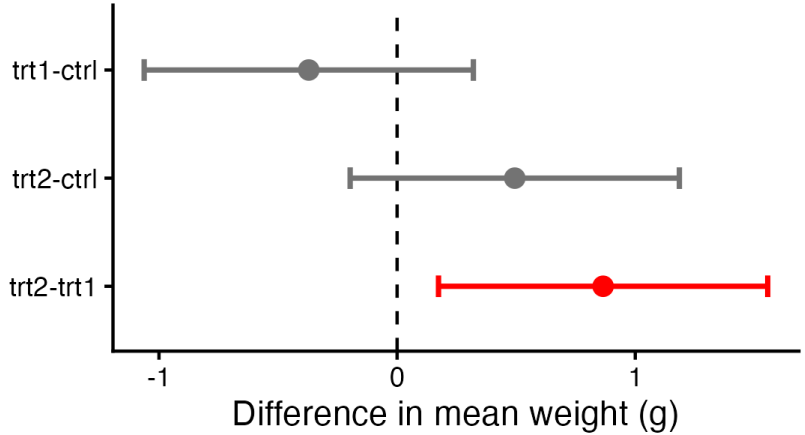

```
> anovafit <- aov(value ~ group, data = df_aov)
> summary(anovafit)
            Df Sum Sq Mean Sq F value Pr(>F)
group        3   49.5  16.487   3.308 0.0234 *
Residuals   96  478.5   4.985
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> fit
Call:
   aov(formula = value ~ group, data = df_aov)

Terms:
                    group Residuals
Sum of Squares    49.4604  478.5241
Deg. of Freedom         3        96

Residual standard error: 2.232628
Estimated effects may be unbalanced
```

```
> str(df_aov)
'data.frame':   100 obs. of  2 variables:
 $ group: Factor w/ 4 levels
"A","B","C","D": 1 4 3 3 1 2 3 2 2 1 ...
 $ value: num  13.79 9.61 11.27 12.37 9.44
```

```
> TukeyHSD(fit)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = value ~ group, data = df_aov)

$group
           diff        lwr       upr     p adj
B-A  0.55539226 -1.0956860 2.2064706 0.8154664
C-A  1.91041309  0.2593348 3.5614914 0.0165497
D-A  0.57157795 -1.0795003 2.2226563 0.8021317
C-B  1.35502083 -0.2960575 3.0060991 0.1462149
D-B  0.01618569 -1.6348926 1.6672640 0.9999939
D-C -1.33883514 -2.9899134 0.3122432 0.1540804
```

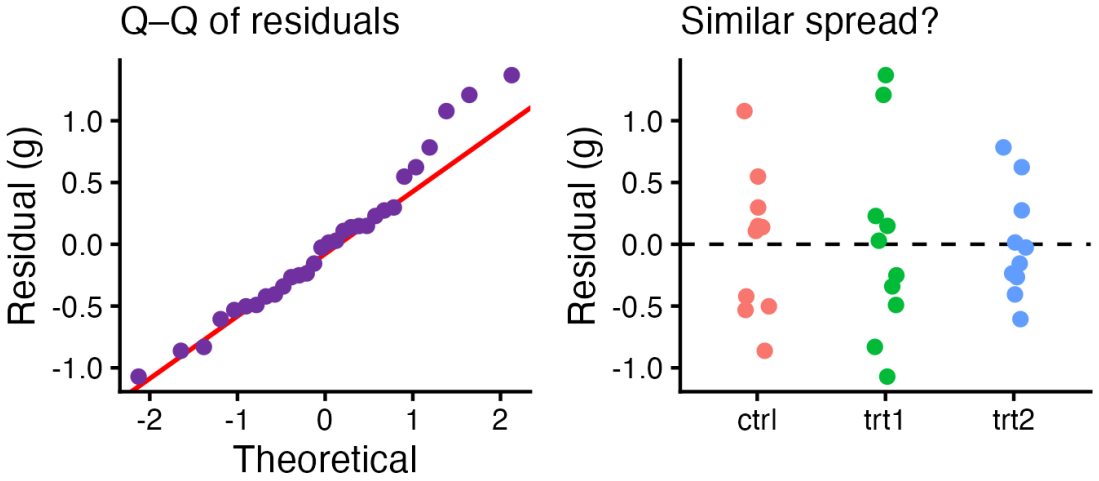
Which groups differ? Tukey HSD

```
> TukeyHSD(fit)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = weight ~ group, data = PlantGrowth)

$group
             diff        lwr       upr     p adj
trt1-ctrl -0.371 -1.0622161 0.3202161 0.3908711
trt2-ctrl  0.494 -0.1972161 1.1852161 0.1979960
trt2-trt1  0.865  0.1737839 1.5562161 0.0120064
```



- Only **trt2 vs. trt1** differs (p = 0.012)
- Tukey adjusts for all pairwise comparisons: the 95% is for the whole family
- An interval that crosses 0: no evidence of a difference

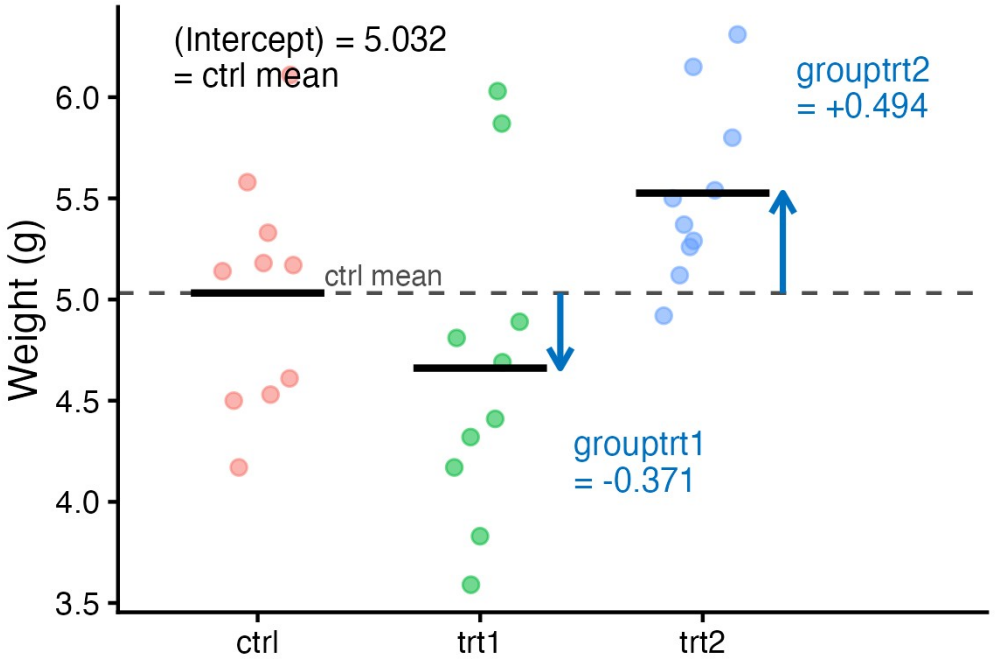
Check the residuals



- Assumptions are about the **residuals**: normal, similar spread, independent
- Here: Shapiro p = 0.44, Bartlett p = 0.24: fine
- If not: transform, **oneway.test()** (Welch), or Kruskal–Wallis

```
> res <- residuals(fit)            # plant − its group mean
> shapiro.test(res)$p.value
[1] 0.4378986
> bartlett.test(weight ~ group, data = PlantGrowth)$p.value
[1] 0.2370968
```

ANOVA is a linear model



$$\text{weight}_i = \beta_0 + \beta_1 \,\text{trt1}_i + \beta_2 \,\text{trt2}_i + \varepsilon_i$$

$$\text{trt1}_i = \begin{cases} 1 & \text{plant } i \text{ in trt1} \\ 0 & \text{otherwise} \end{cases}$$

```
> fit <- lm(weight ~ group, data = PlantGrowth)
> coef(fit)
(Intercept)   grouptrt1   grouptrt2
      5.032      -0.371       0.494
> tapply(PlantGrowth$weight, PlantGrowth$group, mean)
 ctrl  trt1  trt2
5.032 4.661 5.526
```

Each coefficient is an **arrow**: that group's difference from the ctrl mean. The intercept is the ctrl mean.

lm() gives the same ANOVA table

	Df	Sum Sq	Mean Sq	F	p
group (between)	$k-1=2$	3.77	$3.77/2=1.88$	$1.88/0.389=4.85$	0.016
Residuals (within)	$N-k=27$	10.49	$10.49/27=0.389$		
Total	29	14.26			

↑ by hand (sums-of-squares slides) ↓ anova(lm()): the same numbers

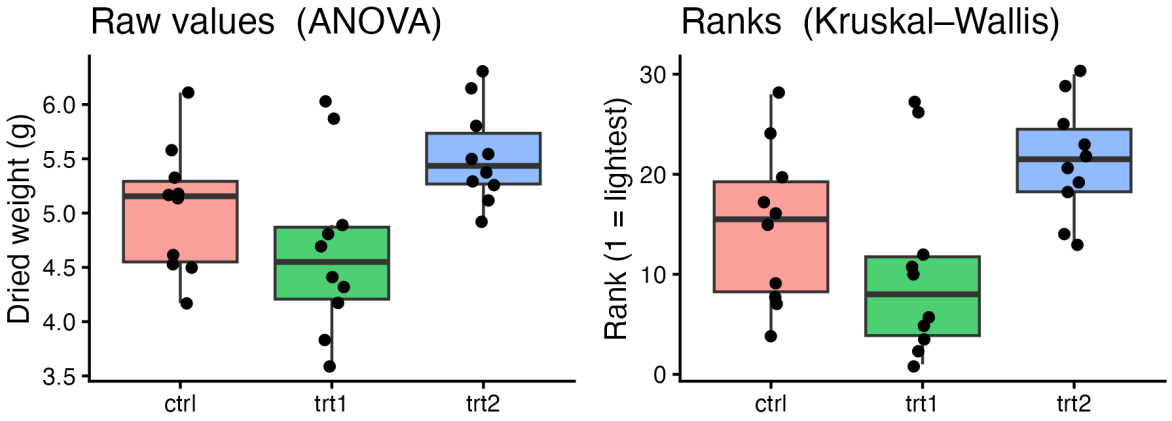
```
> anova(fit)              # the same table as summary(aov())
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq F value Pr(>F)
group      2   3.77   1.883    4.85  0.016 *
Residuals 27  10.49   0.389
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Two groups: $F = t^2$

```
> two <- droplevels(subset(PlantGrowth, group != "trt2"))
> t.test(weight ~ group, two, var.equal = TRUE)$statistic^2
    t
1.419
> anova(lm(weight ~ group, two))$"F value"[1]
[1] 1.419
```

- **aov()** and **lm()** fit the same model; **summary(aov())** and **anova(lm())** print the same table
- **Next week: regression uses the same machinery**

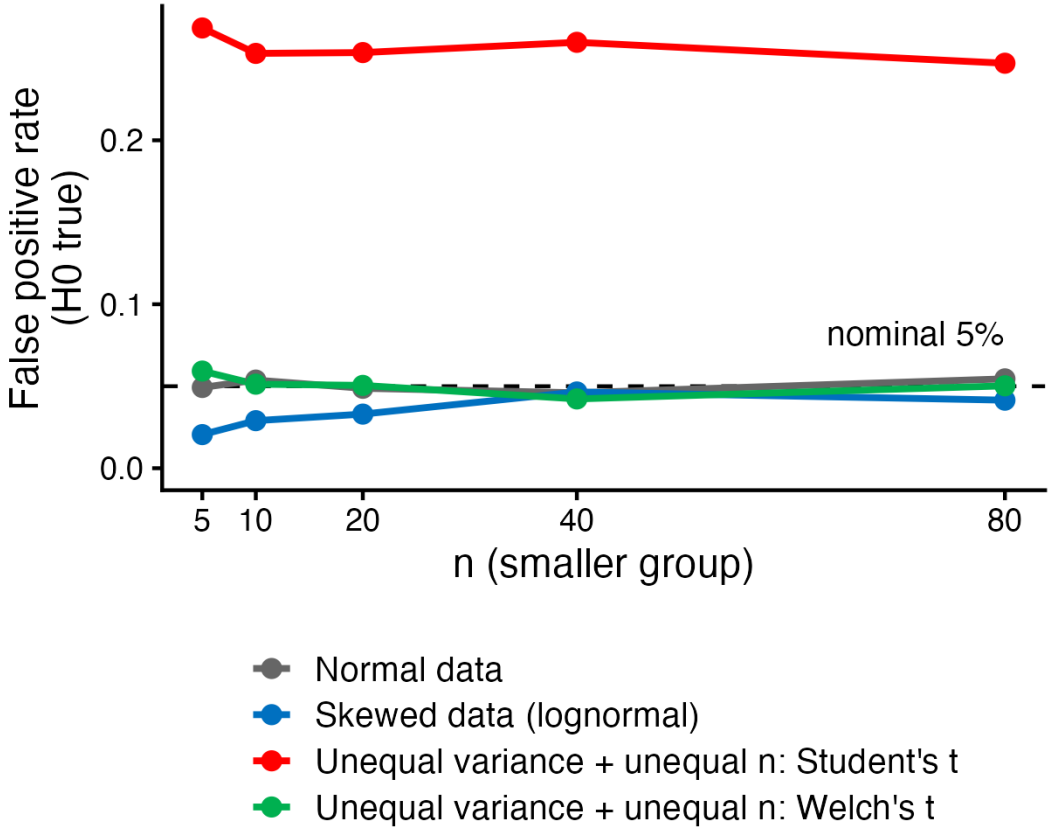
Parametric vs. non-parametric



Same question asked on **ranks**: fewer assumptions, a bit less power. Here: p = 0.016 vs. 0.018.

```
> pg <- PlantGrowth
> summary(aov(weight ~ group, data = pg))
            Df Sum Sq Mean Sq F value Pr(>F)
group        2  3.766  1.8832   4.846 0.0159 *
Residuals   27 10.492  0.3886
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> kruskal.test(weight ~ group, data = pg)$p.value
[1] 0.01842376
```

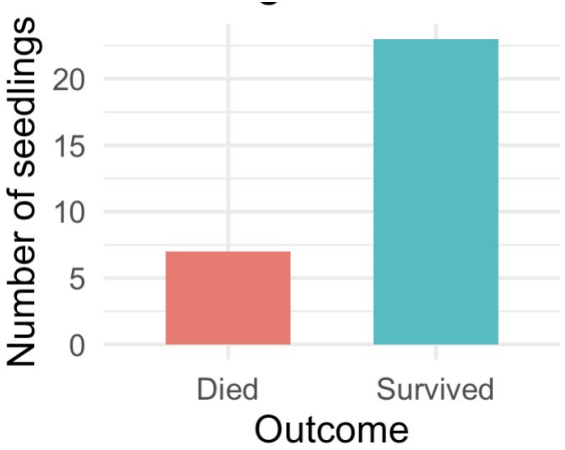
When do broken assumptions matter?



- **Skewed data:** t-test gets conservative (fewer false positives, less power)
- **Unequal variances + unequal n:** Student's t gives ~25% false positives
- **Welch's t** (R's default) fixes it

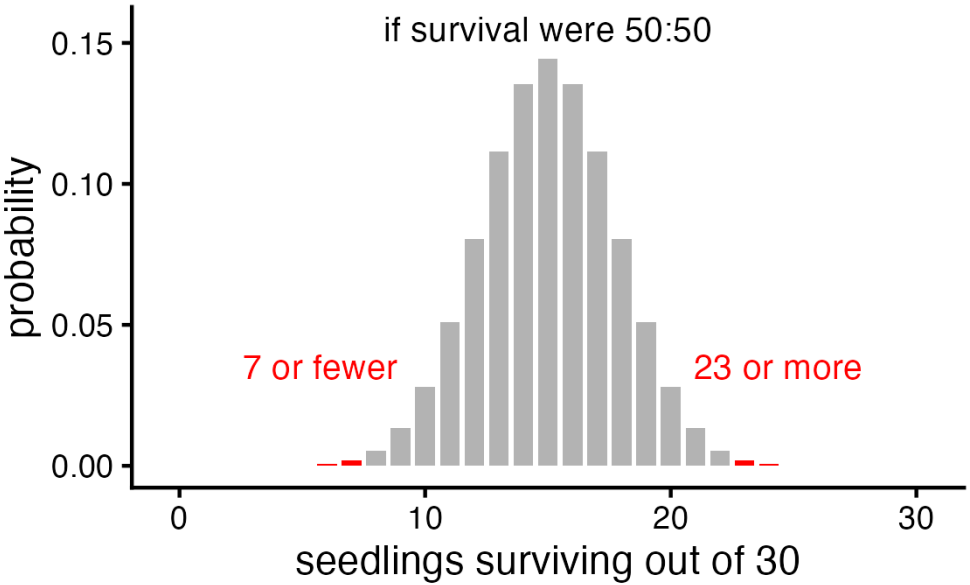
Binomial test

```r
# survival data
survived <- 23
total <- 30
```




```
> binom.test(survived, total, p = 0.5, alternative = "two.sided")

        Exact binomial test

data:  survived and total
number of successes = 23, number of trials = 30, p-value = 0.005223
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.5771635 0.9006621
sample estimates:
probability of success
             0.7666667
```

Binomial test under the hood



$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

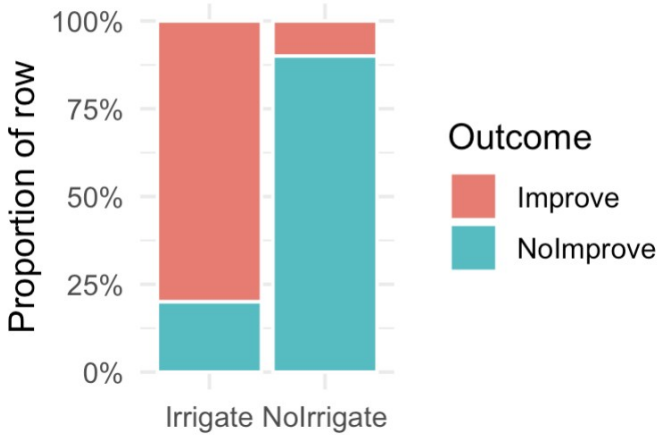
If survival were 50:50, how likely is each count?
Add up the red bars.

```
> k <- 0:30
> p <- dbinom(k, size = 30, prob = 0.5)
> sum(p[k >= 23])              # 23 or more
[1] 0.00261144
> sum(p[k >= 23 | k <= 7])    # two-sided
[1] 0.005222879
```

Exact: no normal curve, no approximation. Two-sided: counts as far from 15 in either direction (≥ 23 or ≤ 7).

Fishers Test

```
> tab_2x2
           Outcome
Treatment    Improve NoImprove
  Irrigate         8         2
  NoIrrigate       1         9
```

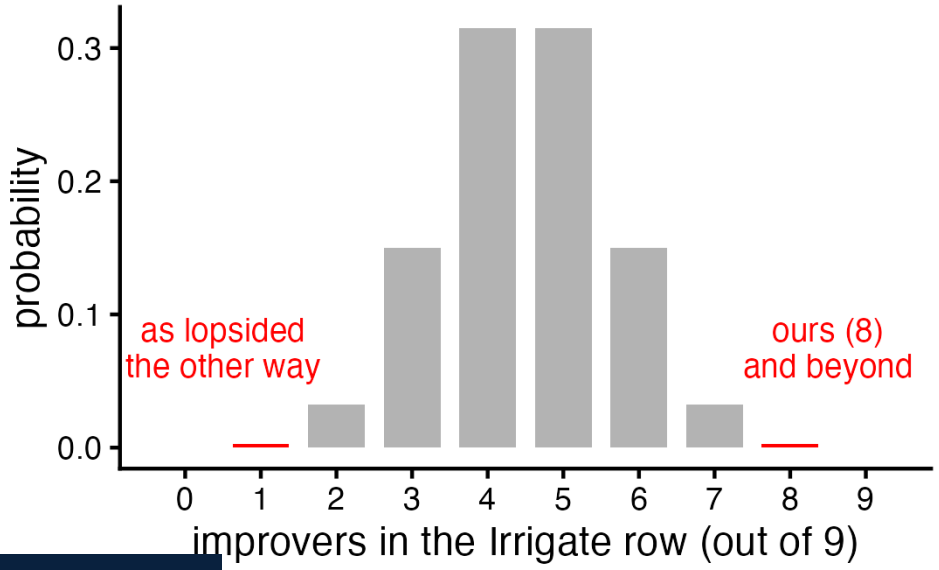



```
> fish_2x2 <- fisher.test(tab_2x2)
> fish_2x2

        Fisher's Exact Test for Count Data

data:  tab_2x2
p-value = 0.005477
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
    2.057999 1740.081669
sample estimates:
odds ratio
  27.32632
```

Fisher's exact test under the hood

- Fix the margins: 10 irrigated, 10 not; 9 improved, 11 not
- If irrigation does nothing, how many of the 9 improvers land in the **Irrigate** row?

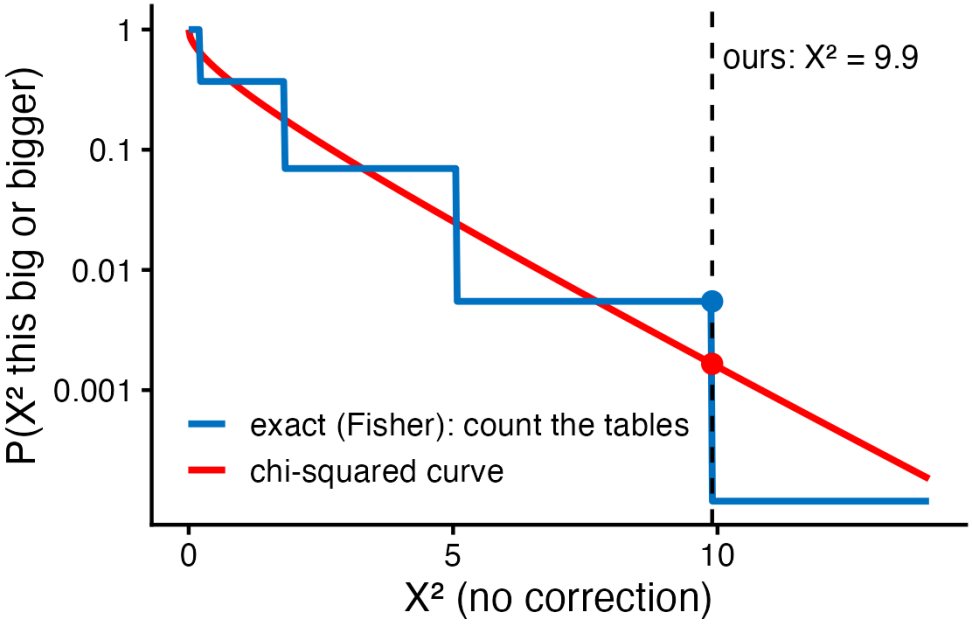
$$P(a) = \frac{\binom{9}{a}\binom{11}{10-a}}{\binom{20}{10}}$$




```
> a <- 0:9    # improvers in the Irrigate row
> p <- dhyper(a, m = 9, n = 11, k = 10)
> round(p, 4)
 [1] 0.0001 0.0027 0.0322 0.1500 0.3151 0.3151 0.1500 0.0322
 [9] 0.0027 0.0001
> sum(p[a >= 8 | a <= 1])   # p-value
[1] 0.005477495
```

p = sum of the **red bars**: every table at least as lopsided as ours

2 × 2: Fisher or chi-squared?



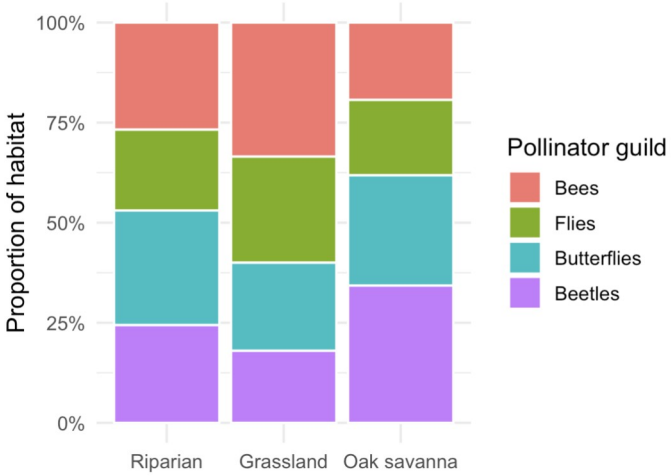

```
> chisq.test(tab_2x2)$expected   # < 5
           Outcome
Treatment    Improve NoImprove
  Irrigate       4.5       5.5
  NoIrrigate     4.5       5.5
> fisher.test(tab_2x2)$p.value   # exact
[1] 0.005477495
> chisq.test(tab_2x2, correct = FALSE)$p.value
[1] 0.001653695
> chisq.test(tab_2x2)$p.value    # + Yates
[1] 0.007000942
```

- **Chi-squared**: a smooth curve for lumpy counts. OK when every expected count ≥ 5
- **Fisher**: exact, counts the tables. Use it for small counts
- Large N: they agree. Big tables: chi-squared

Here expected counts are 4.5: R warns “Chi-squared approximation may be incorrect”

Chi-squared test

```
> tab_rc
             Guild
Habitat       Bees Flies Butterflies Beetles
  Riparian      57    43          61      52
  Grassland     67    53          44      36
  Oak savanna   40    39          57      71
```




```
> chi_rc  <- chisq.test(tab_rc)
> chi_rc

        Pearson's Chi-squared test

data:  tab_rc
X-squared = 23.561, df = 6, p-value = 0.0006289
```

Chi-squared under the hood

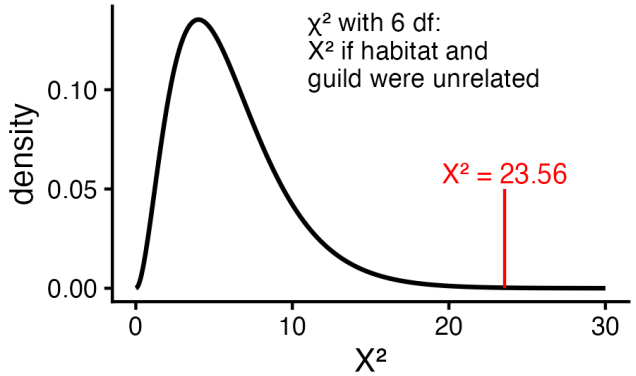
$$E_{ij} = \frac{\text{row}_i \text{ total} \times \text{col}_j \text{ total}}{N}$$

$$X^2 = \sum_{\text{cells}} \frac{(O - E)^2}{E}$$

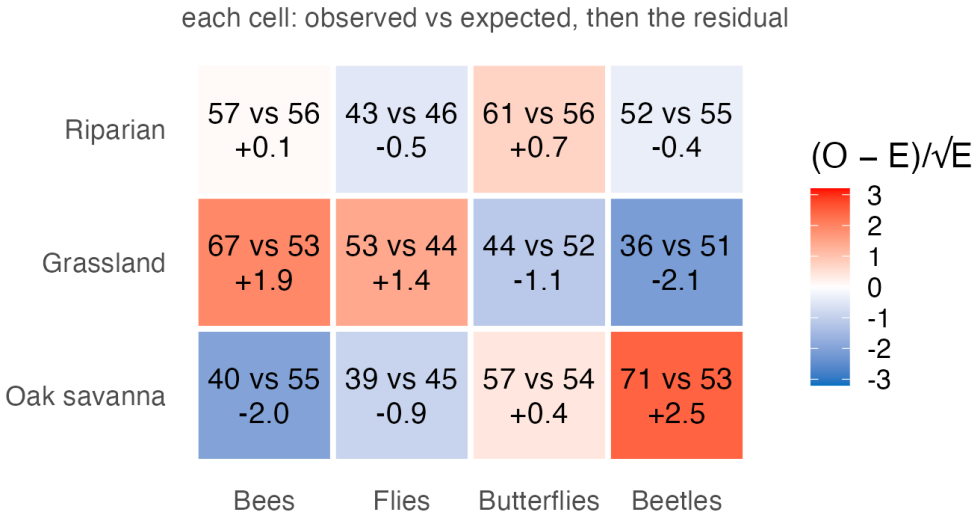
$$df = (r - 1)(c - 1) = 2 \times 3 = 6$$

E = what each cell would hold if habitat and guild were **unrelated**

```
> O <- tab_rc
> E <- outer(rowSums(O), colSums(O)) / sum(O)
> round(E, 1)
            Bees Flies Butterflies Beetles
Riparian    56.3  46.4        55.7    54.6
Grassland   52.9  43.5        52.3    51.3
Oak savanna 54.8  45.1        54.1    53.1
> sum((O - E)^2 / E)              # X-squared
[1] 23.56071
> (nrow(O) - 1) * (ncol(O) - 1)   # df
[1] 6
> pchisq(23.56, df = 6, lower.tail = FALSE)
[1] 0.0006290623
```



Observed vs. expected: where is the effect?



- **Red**: more than expected if unrelated
- **Blue**: fewer than expected
- |residual| > 2 stands out: beetles love oak savanna, avoid grassland

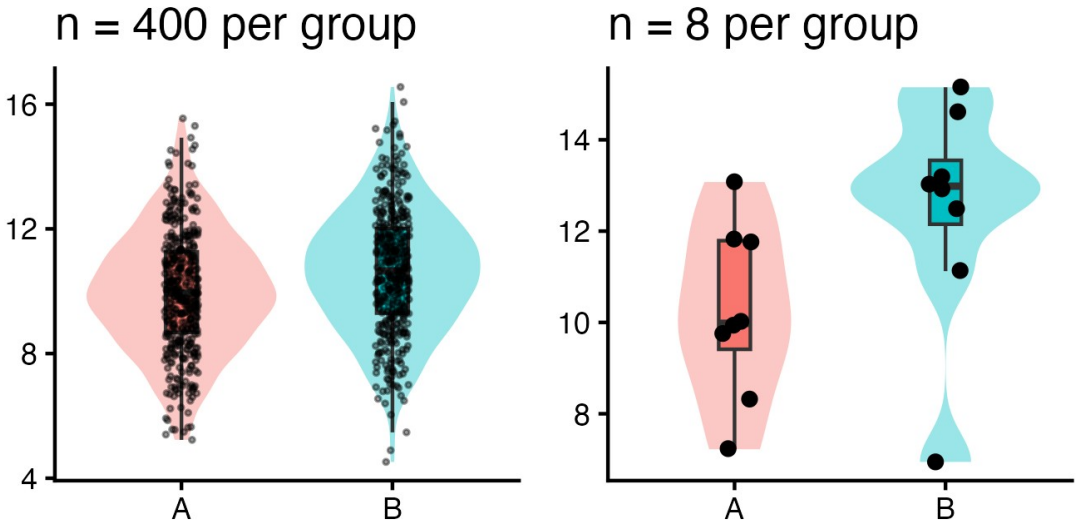
```
> chi_rc <- chisq.test(tab_rc)
> round(chi_rc$residuals, 1)   # (O - E) / sqrt(E)
             Guild
Habitat       Bees Flies Butterflies Beetles
  Riparian     0.1  -0.5         0.7    -0.4
  Grassland    1.9   1.4        -1.1    -2.1
  Oak savanna -2.0  -0.9         0.4     2.5
> sum(chi_rc$residuals^2)      # = X-squared
[1] 23.56071
```

One line in base R: **mosaicplot(tab_rc, shade = TRUE)**

Rough summary and guide (green discussed in lecture):

Goal	Parametric test	Non-parametric alternative	When to use parametric	When to use non-parametric
Compare **one sample mean** vs. a value	One-sample **t-test**	**Wilcoxon signed-rank**	Data ≈ normal	Data skewed, ordinal, outliers
Compare **two independent groups**	Two-sample **t-test**	**Mann–Whitney U** (a.k.a. Wilcoxon rank-sum)	Normal-ish distribution, equal variances	Skewed, ordinal data, unequal variances, outliers
Compare **two paired samples** (before/after)	Paired **t-test**	**Wilcoxon signed-rank**	Normal-ish differences	Non-normal differences, small N, outliers
Compare **>2 groups**	**ANOVA**	**Kruskal–Wallis**	Normal-ish, equal variances	Skewed/ordinal, unequal variances
Compare **>2 repeated measures**	Repeated-measures **ANOVA**	**Friedman test**	Normal-ish, sphericity assumption	Non-normal, ordinal, violations of sphericity
Test **correlation**	**Pearson** correlation	**Spearman** correlation	Linear relationship, normal data	Monotonic (not necessarily linear), skewed/outliers, ordinal
Test **association in counts**	**Chi-squared** test	**Fisher's exact test**	Large sample, expected counts >5	Small sample, expected counts <5
Test a **sample proportion** vs. expected probability	Large-sample z-test for proportions (normal approx)	**Binomial test**	Large n (so normal approximation to binomial is valid, expected successes/failures > 5)	Small n, skewed proportions, exact inference needed

Effect size vs. p-value



n = 400: p = 0.000001, d = 0.35 (small)

n = 8: p = 0.07, d = 0.97 (large)

p mixes effect size with sample size. Report the **effect size** and its CI.

Cohen's d

$$d = \frac{|\bar{x}_1 - \bar{x}_2|}{s_p} \qquad s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$$

```
> cohens_d <- function(x1, x2) {
+   n1 <- length(x1); n2 <- length(x2)
+   sp <- sqrt(((n1-1)*var(x1) + (n2-1)*var(x2)) / (n1+n2-2))
+   abs(mean(x1) - mean(x2)) / sp
+ }
> t.test(y ~ g, data = big_n)$p.value      # 400 per group
[1] 1.193761e-06
> with(big_n, cohens_d(y[g == "A"], y[g == "B"]))
[1] 0.346086
> t.test(y ~ g, data = small_n)$p.value    # 8 per group
[1] 0.07406688
> with(small_n, cohens_d(y[g == "A"], y[g == "B"]))
[1] 0.9708547
```

d:
0.2 small
0.5 medium
0.8 large

Eta-squared (ANOVA)

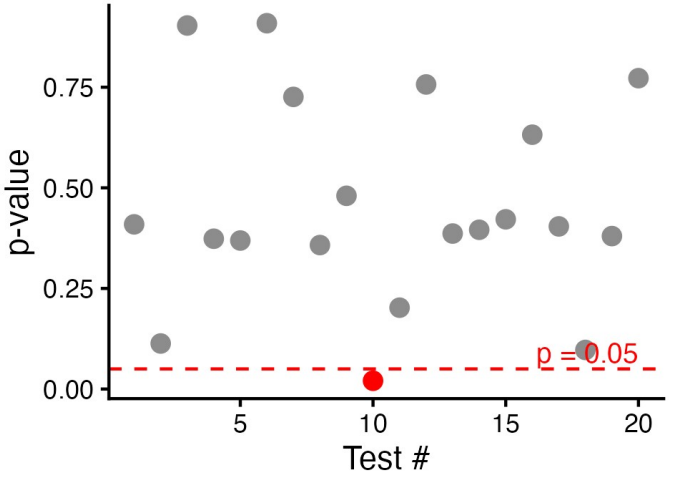
$$\eta^2 = \frac{SS_{\text{group}}}{SS_{\text{total}}} = \frac{3.77}{3.77 + 10.49} = 0.26$$

```
> a1 <- aov(weight ~ group, data = pg)
> ss <- summary(a1)[[1]][, "Sum Sq"]
> ss                    # SS_group, SS_residual
[1]  3.76634 10.49209
> ss[1] / sum(ss)       # eta-squared
[1] 0.2641483
```

Treatment explains **26%** of the variation in plant weight. η^2: 0.01 small, 0.06 medium, 0.14 large

Multiple testing

```
> set.seed(3)
> # 20 comparisons where NOTHING is going on
> p <- replicate(20, t.test(rnorm(20), rnorm(20))$p.value)
> round(sort(p), 3)
 [1] 0.021 0.097 0.113 0.202 0.358 0.369 0.374 0.380 0.386 0.396
[11] 0.404 0.409 0.422 0.480 0.632 0.726 0.757 0.772 0.903 0.909
> sum(p < 0.05)
[1] 1
> 1 - 0.95^20      # P(at least one false positive)
[1] 0.6415141
```



$$P(\text{at least one false positive}) = 1 - (1 - 0.05)^{20} = 0.64$$

Correcting for multiple tests

```
> set.seed(1)
> null <- replicate(18, t.test(rnorm(20), rnorm(20))$p.value)
> real <- c(t.test(rnorm(20, 1), rnorm(20))$p.value,
+           t.test(rnorm(20, 2), rnorm(20))$p.value)
> raw_p <- c(null, real)     # tests 19 and 20 are real
> which(raw_p < 0.05)
[1]  5 12 19 20
> sum(p.adjust(raw_p, method = "holm") < 0.05)
[1] 2
> sum(p.adjust(raw_p, method = "BH") < 0.05)
[1] 2
```

$$\alpha_{\text{Bonferroni}} = \frac{\alpha}{m} = \frac{0.05}{20} = 0.0025$$

- **Holm:** controls the chance of any false positive
- **BH (FDR):** controls the share of false discoveries; for many tests (genomics)
- Pick the method **before** looking at results

Next week: Linear regression

Problem Set 1 due Monday, October 5, 1:10 PM