

STAC32: Applications of Statistical Methods

Lecture notes

Section 1

Statistical Inference

Statistical Inference and Science

- Previously: descriptive statistics. “Here are data; what do they say?”.
- May need to take some action based on information in data.
- Or want to generalize beyond data (sample) to larger world (population).
- Science: first guess about how world works.
- Then collect data, by sampling.
- Is guess correct (based on data) for whole world, or not?

Sample data are imperfect

- Sample data never entirely represent what you're observing.
- There is always random error present.
- Thus you can never be entirely certain about your conclusions.
- The Toronto Blue Jays' average home attendance in part of 2015 season was 25,070 (up to May 27 2015, from baseball-reference.com).
- Does that mean the attendance at every game was exactly 25,070?
Certainly not. Actual attendance depends on many things, eg.:
 - how well the Jays are playing
 - the opposition
 - day of week
 - weather
 - random chance

Packages for this section

```
library(tidyverse)
library(smmr)
# library(PMCMRplus)
```

Reading the attendances

...as a .csv file:

```
jays = read_csv("jays15-home.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   row = col_double(),
##   game = col_double(),
##   venue = col_logical(),
##   runs = col_double(),
##   Oppruns = col_double(),
##   innings = col_double(),
##   position = col_double(),
##   `game time` = col_time(format = ""),
##   attendance = col_double()
## )
```

Taking a look

jays

row	game	date	box	team	venue	opp	result	runs	Oppru
82	7	Monday, Apr 13	boxscore	TOR	NA	TBR	L	1	
83	8	Tuesday, Apr 14	boxscore	TOR	NA	TBR	L	2	
84	9	Wednesday, Apr 15	boxscore	TOR	NA	TBR	W	12	
85	10	Thursday, Apr 16	boxscore	TOR	NA	TBR	L	2	
86	11	Friday, Apr 17	boxscore	TOR	NA	ATL	L	7	
87	12	Saturday, Apr 18	boxscore	TOR	NA	ATL	W-wo	6	
88	13	Sunday, Apr 19	boxscore	TOR	NA	ATL	L	2	
89	14	Tuesday, Apr 21	boxscore	TOR	NA	BAL	W	13	
90	15	Wednesday, Apr 22	boxscore	TOR	NA	BAL	W	4	
91	16	Thursday, Apr 23	boxscore	TOR	NA	BAL	W	7	
92	27	Monday, May 4	boxscore	TOR	NA	NYN	W	3	
93	28	Tuesday, May 5	boxscore	TOR	NA	NYN	L	3	
94	29	Wednesday, May 6	boxscore	TOR	NA	NYN	W	5	
95	30	Friday, May 8	boxscore	TOR	NA	BOS	W	7	
96	31	Saturday, May 9	boxscore	TOR	NA	BOS	W	7	
97	32	Sunday, May 10	boxscore	TOR	NA	BOS	L	3	
98	40	Monday, May 18	boxscore	TOR	NA	LAA	W	10	
99	41	Tuesday, May 19	boxscore	TOR	NA	LAA	L	2	
100	42	Wednesday, May 20	boxscore	TOR	NA	LAA	L	3	
101	43	Thursday, May 21	boxscore	TOR	NA	LAA	W	8	

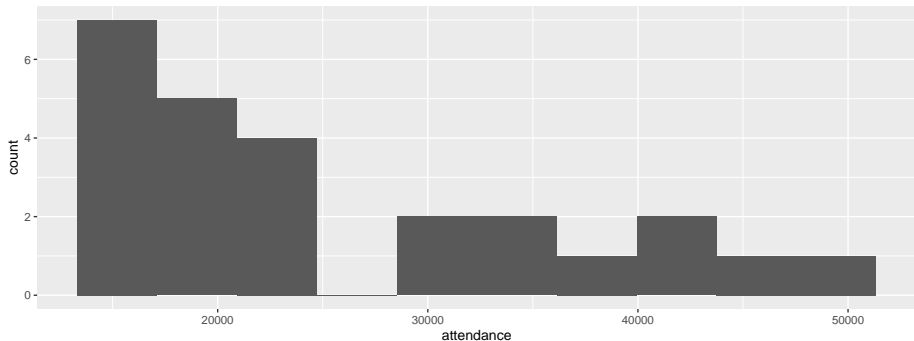
Another way

```
glimpse(jays)
```

```
## Rows: 25
## Columns: 21
## $ row      <dbl> 82, 83, 84, 85, 86, 87, 88,...
## $ game     <dbl> 7, 8, 9, 10, 11, 12, 13, 14...
## $ date     <chr> "Monday, Apr 13", "Tuesday,...
## $ box      <chr> "boxscore", "boxscore", "bo...
## $ team     <chr> "TOR", "TOR", "TOR", "TOR",...
## $ venue    <lgl> NA, NA, NA, NA, NA, NA, NA,...
## $ opp      <chr> "TBR", "TBR", "TBR", "TBR",...
## $ result   <chr> "L", "L", "W", "L", "L", "W...
## $ runs     <dbl> 1, 2, 12, 2, 7, 6, 2, 13, 4...
## $ Oppruns  <dbl> 2, 3, 7, 4, 8, 5, 5, 6, 2, ...
## $ innings  <dbl> NA, NA, NA, NA, NA, 10, NA,...
## $ wl       <chr> "4-3", "4-4", "5-4", "5-5",...
## $ position <dbl> 2, 3, 2, 4, 4, 3, 4, 2, 2, ...
## $ gb       <chr> "1", "2", "1", "1.5", "2.5"...
## $ winner   <chr> "Odorizzi", "Geltz", "Buehr...
## $ loser    <chr> "Dickey", "Castro", "Ramire...
## $ save     <chr> "Boxberger", "Jepsen", NA, ...
## $ `game time` <time> 02:30:00, 03:06:00, 03:02:...
## $ Daynight <chr> "N", "N", "N", "N", "N", "D...
## $ attendance <dbl> 48414, 17264, 15086, 14433,...
## $ streak   <chr> "-", "--", "+", "-", "--", ...
```


Attendance histogram

```
ggplot(jays, aes(x = attendance)) + geom_histogram(bins = 10)
```



Comments

- Attendances have substantial variability, ranging from just over 10,000 to around 50,000.
- Distribution somewhat skewed to right (but no outliers).
- These are a sample of “all possible games” (or maybe “all possible games played in April and May”). What can we say about mean attendance in all possible games based on this evidence?
- Think about:
 - Confidence interval
 - Hypothesis test.

Getting CI for mean attendance

- `t.test` function does CI and test. Look at CI first:

```
t.test(jays$attendance)
```

```
##  
## One Sample t-test  
##  
## data: jays$attendance  
## t = 11.389, df = 24, p-value = 3.661e-11  
## alternative hypothesis: true mean is not equal to 0  
## 95 percent confidence interval:  
## 20526.82 29613.50  
## sample estimates:  
## mean of x  
## 25070.16
```

- From 20,500 to 29,600.

Or, 90% CI

- by including a value for `conf.level`:

```
t.test(jays$attendance, conf.level = 0.90)
```

```
##  
## One Sample t-test  
##  
## data: jays$attendance  
## t = 11.389, df = 24, p-value = 3.661e-11  
## alternative hypothesis: true mean is not equal to 0  
## 90 percent confidence interval:  
## 21303.93 28836.39  
## sample estimates:  
## mean of x  
## 25070.16
```

- From 21,300 to 28,800. (Shorter, as it should be.)

Comments

- Need to say “column attendance within data frame jays” using \$.
- 95% CI from about 20,000 to about 30,000.
- Not estimating mean attendance well at all!
- Generally want confidence interval to be shorter, which happens if:
 - SD smaller
 - sample size bigger
 - confidence level smaller
- Last one is a cheat, really, since reducing confidence level increases chance that interval won't contain pop. mean at all!

Another way to access data frame columns

```
with(jays, t.test(attendance))
```

```
##  
## One Sample t-test  
##  
## data: attendance  
## t = 11.389, df = 24, p-value = 3.661e-11  
## alternative hypothesis: true mean is not equal to 0  
## 95 percent confidence interval:  
## 20526.82 29613.50  
## sample estimates:  
## mean of x  
## 25070.16
```

Hypothesis test

- CI answers question “what is the mean?”
- Might have a value μ in mind for the mean, and question “Is the mean equal to μ , or not?”
- For example, 2014 average attendance was 29,327.
- “Is the mean this?” answered by **hypothesis test**.
- Value being assessed goes in **null hypothesis**: here, $H_0 : \mu = 29327$.
- **Alternative hypothesis** says how null might be wrong, eg.
 $H_a : \mu \neq 29327$.
- Assess evidence against null. If that evidence strong enough, *reject null hypothesis*; if not, *fail to reject null hypothesis* (sometimes *retain null*).
- Note asymmetry between null and alternative, and utter absence of word “accept”.

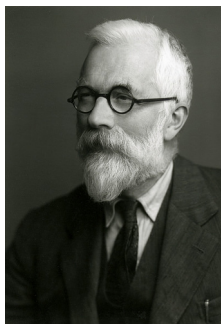
α and errors

- Hypothesis test ends with decision:
 - reject null hypothesis
 - do not reject null hypothesis.
- but decision may be wrong:

Truth	Decision	
	Do not reject	Reject null
Null true	Correct	Type I error
Null false	Type II error	Correct

- Either type of error is bad, but for now focus on controlling Type I error: write $\alpha = P(\text{type I error})$, and devise test so that α small, typically 0.05.
- That is, **if null hypothesis true**, have only small chance to reject it (which would be a mistake).
- Worry about type II errors later (when we consider power of test).

Why 0.05? This man.



Responsible for:

- analysis of variance
- Fisher information
- Linear discriminant analysis
- Fisher's z -transformation
- Fisher-Yates shuffle
- Behrens-Fisher problem

Sir Ronald A. Fisher, 1890–1962.

Why 0.05? (2)

- From The Arrangement of Field Experiments (1926):

the line at about the level at which we can say: "Either there is something in the treatment, or a coincidence has occurred such as does not occur more than once in twenty trials." This level, which we may call the 5 per cent. point, would be indicated, though very roughly, by the greatest chance deviation observed in twenty successive trials. To

- and

If one in twenty does not seem high enough odds, we may, if we prefer it, draw the line at one in fifty (the 2 per cent. point), or one in a hundred (the 1 per cent. point). Personally, the writer prefers to set a low standard of significance at the 5 per cent. point, and ignore entirely all results which fail to reach this level. A scientific fact should be regarded as experimentally established only if a properly designed experiment rarely fails to give this level of significance. The very high

Three steps:

- from data to test statistic
 - how far are data from null hypothesis
- from test statistic to P-value
 - how likely are you to see “data like this” **if the null hypothesis is true**
- from P-value to decision
 - reject null hypothesis if P-value small enough, fail to reject it otherwise

Using t.test:

```
t.test(jays$attendance, mu=29327)
```

```
##  
## One Sample t-test  
##  
## data: jays$attendance  
## t = -1.9338, df = 24, p-value = 0.06502  
## alternative hypothesis: true mean is not equal to 29327  
## 95 percent confidence interval:  
## 20526.82 29613.50  
## sample estimates:  
## mean of x  
## 25070.16
```

- See test statistic -1.93 , P-value 0.065 .
- Do not reject null at $\alpha = 0.05$: no evidence that mean attendance has changed.

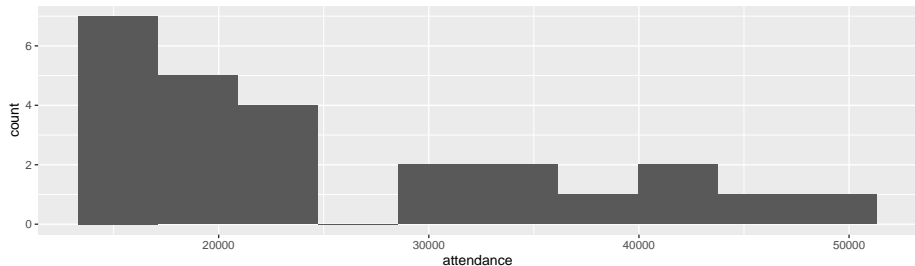
Assumptions

- Theory for t -test: assumes normally-distributed data.
- What actually matters is sampling distribution of sample mean: if this is approximately normal, t -test is OK, even if data distribution is not normal.
- Central limit theorem: if sample size large, sampling distribution approx. normal even if data distribution somewhat non-normal.
- So look at shape of data distribution, and make a call about whether it is normal enough, given the sample size.

Blue Jays attendances again:

- You might say that this is not normal enough for a sample size of $n = 25$, in which case you don't trust the t -test result:

```
ggplot(jays, aes(x = attendance)) + geom_histogram(bins = 10)
```



Another example: learning to read

- You devised new method for teaching children to read.
- Guess it will be more effective than current methods.
- To support this guess, collect data.
- Want to generalize to “all children in Canada”.
- So take random sample of all children in Canada.
- Or, argue that sample you actually have is “typical” of all children in Canada.
- Randomization (1): whether or not a child in sample or not has nothing to do with anything else about that child.
- Randomization (2): randomly choose whether each child gets new reading method (t) or standard one (c).

Reading in data

- File at <http://www.utsc.utoronto.ca/~butler/c32/drp.txt>.
- Proper reading-in function is `read_delim` (check file to see)
- Read in thus:

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/drp.txt"  
kids=read_delim(my_url, " ")
```

```
## Parsed with column specification:  
## cols(  
##   group = col_character(),  
##   score = col_double()  
## )
```

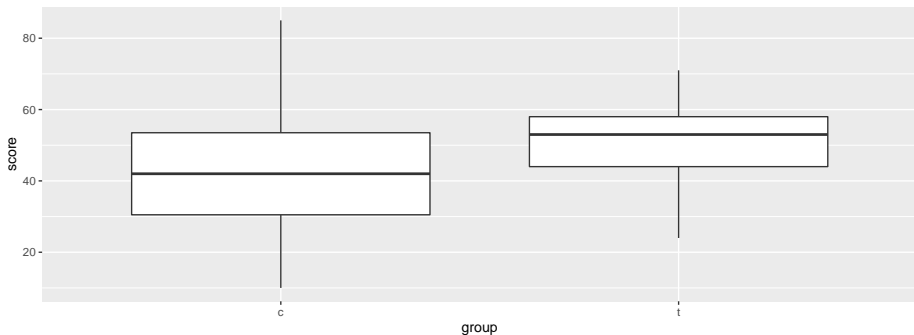

The data (some)

kids

group	score
t	24
t	61
t	59
t	46
t	43
t	44
t	52
t	43
t	58
t	67
t	62
t	57
t	71

Boxplots

```
ggplot(kids, aes(x = group, y = score)) + geom_boxplot()
```



Two kinds of two-sample t-test

- Do the two groups have same spread (SD, variance)?
- If yes (shaky assumption here), can use pooled t-test.
- If not, use Welch-Satterthwaite t-test (safe).
- Pooled test derived in STAB57 (easier to derive).
- Welch-Satterthwaite is test used in STAB22 and is generally safe.
- Assess (approx) equality of spreads using boxplot.

The (Welch-Satterthwaite) t-test

- c (control) before t (treatment) alphabetically, so proper alternative is “less”.
- R does Welch-Satterthwaite test by default
- new reading program really helps?
- (in a moment) how to get R to do pooled test?

Welch-Satterthwaite

```
t.test(score ~ group, data = kids, alternative = "less")

##
##  Welch Two Sample t-test
##
## data:  score by group
## t = -2.3109, df = 37.855, p-value = 0.01319
## alternative hypothesis: true difference in means is less th
## 95 percent confidence interval:
##      -Inf -2.691293
## sample estimates:
## mean in group c mean in group t
##      41.52174      51.47619
```

The pooled t-test

```
t.test(score ~ group, data = kids,
       alternative = "less", var.equal = T)
```

```
##
## Two Sample t-test
##
## data:  score by group
## t = -2.2666, df = 42, p-value = 0.01431
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##      -Inf -2.567497
## sample estimates:
## mean in group c mean in group t
##      41.52174      51.47619
```

Two-sided test; CI

- To do 2-sided test, leave out alternative:

```
t.test(score ~ group, data = kids)
```

```
##
##  Welch Two Sample t-test
##
## data:  score by group
## t = -2.3109, df = 37.855, p-value = 0.02638
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -18.67588  -1.23302
## sample estimates:
## mean in group c mean in group t
##           41.52174           51.47619
```

Comments:

- P-values for pooled and Welch-Satterthwaite tests very similar (even though the pooled test seemed inferior): 0.013 vs. 0.014.
- Two-sided test also gives CI: new reading program increases average scores by somewhere between about 1 and 19 points.
- Confidence intervals inherently two-sided, so do 2-sided test to get them.

Jargon for testing

- Alternative hypothesis: what we are trying to prove (new reading program is effective).
- Null hypothesis: “there is no difference” (new reading program no better than current program). Must contain “equals”.
- One-sided alternative: trying to prove better (as with reading program).
- Two-sided alternative: trying to prove different.
- Test statistic: something expressing difference between data and null (eg. difference in sample means, t statistic).
- P-value: probability of observing test statistic value as extreme or more extreme, if null is true.
- Decision: either reject null hypothesis or do not reject null hypothesis.
Never “accept”.

Logic of testing

- Work out what would happen if null hypothesis were true.
- Compare to what actually did happen.
- If these are too far apart, conclude that null hypothesis is not true after all. (Be guided by P-value.)
- As applied to our reading programs:
 - If reading programs equally good, expect to see a difference in means close to 0.
 - Mean reading score was 10 higher for new program.
 - Difference of 10 was unusually big (P-value small from t-test). So conclude that new reading program is effective.
- Nothing here about what happens if null hypothesis is false. This is power and type II error probability.

Errors in testing

What can happen:

Truth	Decision	
	Do not reject	Reject null
Null true	Correct	Type I error
Null false	Type II error	Correct

Tension between truth and decision about truth (imperfect).

- Prob. of type I error denoted α . Usually fix α , eg. $\alpha = 0.05$.
- Prob. of type II error denoted β . Determined by the planned experiment. Low β good.
- Prob. of not making type II error called **power** ($= 1 - \beta$). *High power* good.

Power

- Suppose $H_0 : \theta = 10$, $H_a : \theta \neq 10$ for some parameter θ .
- Suppose H_0 wrong. What does that say about θ ?
- Not much. Could have $\theta = 11$ or $\theta = 8$ or $\theta = 496$. In each case, H_0 wrong.
- How likely a type II error is depends on what θ is:
 - If $\theta = 496$, should be able to reject $H_0 : \theta = 10$ even for small sample, so β should be small (power large).
 - If $\theta = 11$, might have hard time rejecting H_0 even with large sample, so β would be larger (power smaller).
- Power depends on true parameter value, and on sample size.
- So we play “what if”: “if θ were 11 (or 8 or 496), what would power be?”.

Figuring out power

- Time to figure out power is before you collect any data, as part of planning process.
- Need to have idea of what kind of departure from null hypothesis of interest to you, eg. average improvement of 5 points on reading test scores. (Subject-matter decision, not statistical one.)
- Then, either:
 - “I have this big a sample and this big a departure I want to detect. What is my power for detecting it?”
 - “I want to detect this big a departure with this much power. How big a sample size do I need?”

How to understand/estimate power?

- Suppose we test $H_0 : \mu = 10$ against $H_a : \mu \neq 10$, where μ is population mean.
- Suppose in actual fact, $\mu = 8$, so H_0 is wrong. We want to reject it. How likely is that to happen?
- Need population SD (take $\sigma = 4$) and sample size (take $n = 15$). In practice, get σ from pilot/previous study, and take the n we plan to use.
- Idea: draw a random sample from the true distribution, test whether its mean is 10 or not.
- Repeat previous step “many” times.
- “Simulation”.

Making it go

- Random sample of 15 normal observations with mean 8 and SD 4:

```
x = rnorm(15, 8, 4)
x
```

```
## [1] 14.487469  5.014611  6.924277  5.201860
## [5]  8.852952 10.835874  3.686684 11.165242
## [9]  8.016188 12.383518  1.378099  3.172503
## [13] 13.074996 11.353573  5.015575
```

- Test whether x from population with mean 10 or not (over):

...continued

```
t.test(x, mu = 10)
```

```
##
```

```
## One Sample t-test
```

```
##
```

```
## data: x
```

```
## t = -1.8767, df = 14, p-value = 0.08157
```

```
## alternative hypothesis: true mean is not equal to 10
```

```
## 95 percent confidence interval:
```

```
## 5.794735 10.280387
```

```
## sample estimates:
```

```
## mean of x
```

```
## 8.037561
```

- Fail to reject the mean being 10 (a Type II error).

or get just P-value

```
t.test(x, mu = 10)$p.value
```

```
## [1] 0.0815652
```

Run this lots of times

- Two steps:
 - Generate a bunch of random samples
 - extract the P-value for the t-test from each
- without a loop!
- Use `rerun` to generate the random samples
- Use `map` to run the test on each random sample
- Use `map_dbl` to pull out the P-value for each test
- Count up how many of the P-values are 0.05 or less.

In code

```
rerun(10000, rnorm(15, 8, 4)) %>%
  map( ~ t.test(., mu = 10)) %>%
  map_dbl("p.value") ->
  pvals
tibble(pvals) %>% count(pvals <= 0.05)
```

pvals <= 0.05	n
FALSE	5547
TRUE	4453

We correctly rejected 422 times out of 1000, so the estimated power is 0.422.

Calculating power

- Simulation approach very flexible: will work for any test. But answer different each time because of randomness.
- In some cases, for example 1-sample and 2-sample t-tests, power can be calculated.
- `power.t.test`. delta difference between null and true mean:

```
power.t.test(n = 15, delta = 10-8, sd = 4, type = "one.sample")
```

```
##  
##      One-sample t test power calculation  
##  
##              n = 15  
##            delta = 2  
##             sd = 4  
##      sig.level = 0.05  
##            power = 0.4378466  
## alternative = two.sided
```

Comparison of results

Method	Power
Simulation	0.422
power.t.test	0.4378

- Simulation power is similar to calculated power; to get more accurate value, repeat more times (eg. 10,000 instead of 1,000), which takes longer.
- CI for power based on simulation approx. 0.42 ± 0.03 .
- With this small a sample size, the power is not great. With a bigger sample, the sample mean should be closer to 8 most of the time, so would reject $H_0 : \mu = 10$ more often.

Calculating required sample size

- Often, when planning a study, we do not have a particular sample size in mind. Rather, we want to know how big a sample to take. This can be done by asking how big a sample is needed to achieve a certain power.
- The simulation approach does not work naturally with this, since you have to supply a sample size.
- For the power-calculation method, you supply a value for the power, but leave the sample size missing.
- Re-use the same problem: $H_0 : \mu = 10$ against 2-sided alternative, true $\mu = 8$, $\sigma = 4$, but now aim for power 0.80.

Using power.t.test

- No n=, replaced by a power=:

```
power.t.test(power=0.80, delta=10-8, sd=4, type="one.sample")
```

```
##  
##      One-sample t test power calculation  
##  
##              n = 33.3672  
##            delta = 2  
##             sd = 4  
##    sig.level = 0.05  
##         power = 0.8  
## alternative = two.sided
```

- Sample size must be a whole number, so round up to 34 (to get at least as much power as you want).

Power curves

- Rather than calculating power for one sample size, or sample size for one power, might want a picture of relationship between sample size and power.
- Or, likewise, picture of relationship between difference between true and null-hypothesis means and power.
- Called power curve.
- Build and plot it yourself.

Building it

- If you feed `power.t.test` a collection (“vector”) of values, it will do calculation for each one.
- Do power for variety of sample sizes, from 10 to 100 in steps of 10:

```
ns=seq(10,100,10)
ns
```

```
## [1] 10 20 30 40 50 60 70 80 90 100
```

- Calculate powers:

```
ans=power.t.test(n=ns, delta=10-8, sd=4, type="one.sample")
ans$power
```

```
## [1] 0.2928286 0.5644829 0.7539627 0.8693979
## [5] 0.9338976 0.9677886 0.9847848 0.9929987
## [9] 0.9968496 0.9986097
```

Building a plot

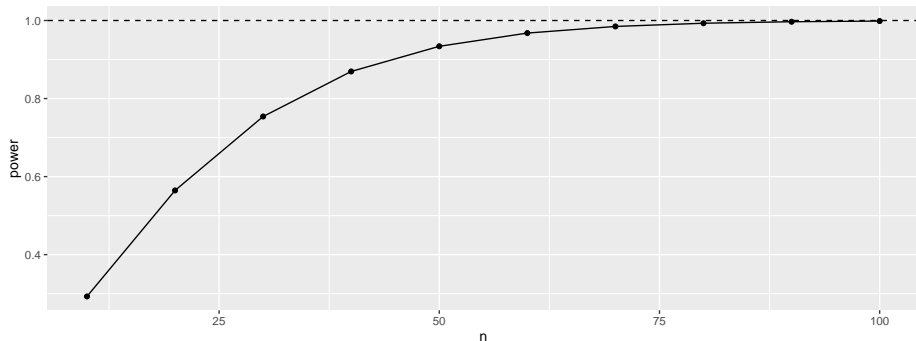
- Make a data frame out of the values to plot:

```
d=tibble(n=ns, power=ans$power)
d
```

n	power
10	0.2928286
20	0.5644829
30	0.7539627
40	0.8693979
50	0.9338976
60	0.9677886
70	0.9847848
80	0.9929987
90	0.9968496
100	0.9986097

The power curve

g



Another way to do it:

```
tibble(n=ns) %>%
  mutate(power_output=map(n, ~power.t.test(n=., delta=10-8, s=1,
  mutate(power=map_dbl(power_output, "power"))) %>%
```

Power curves for means

- Can also investigate power as it depends on what the true mean is (the farther from null mean 10, the higher the power will be).
- Investigate for two different sample sizes, 15 and 30.
- First make all combos of mean and sample size:

```
means=seq(6,10,0.5)
```

```
means
```

```
## [1] 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0
```

```
ns=c(15,30)
```

```
ns
```

```
## [1] 15 30
```

```
combos=crossing(mean=means, n=ns)
```

The combos

combos

mean	n
6.0	15
6.0	30
6.5	15
6.5	30
7.0	15
7.0	30
7.5	15
7.5	30
8.0	15
8.0	30
8.5	15
8.5	30
9.0	15

Calculate and plot

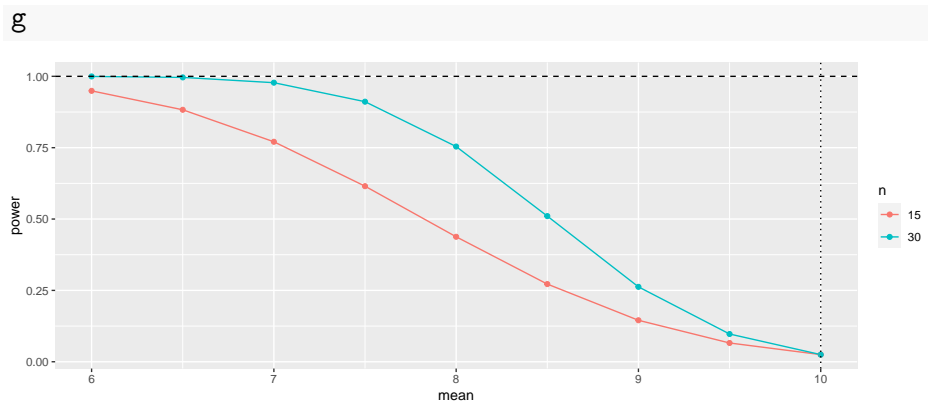
- Calculate the powers, carefully:

```
ans=with(combos, power.t.test(n=n, delta=mean-10, sd=4,
                             type="one.sample"))
ans
```

```
##
##      One-sample t test power calculation
##
##              n = 15, 30, 15, 30, 15, 30, 15, 30, 15, 30, 1
##              delta = 4.0, 4.0, 3.5, 3.5, 3.0, 3.0, 2.5, 2.5, 2
##              sd = 4
##              sig.level = 0.05
##              power = 0.94908647, 0.99956360, 0.88277128, 0.996
##              alternative = two.sided
```

```
names(ans)
```

The power curves



Comments

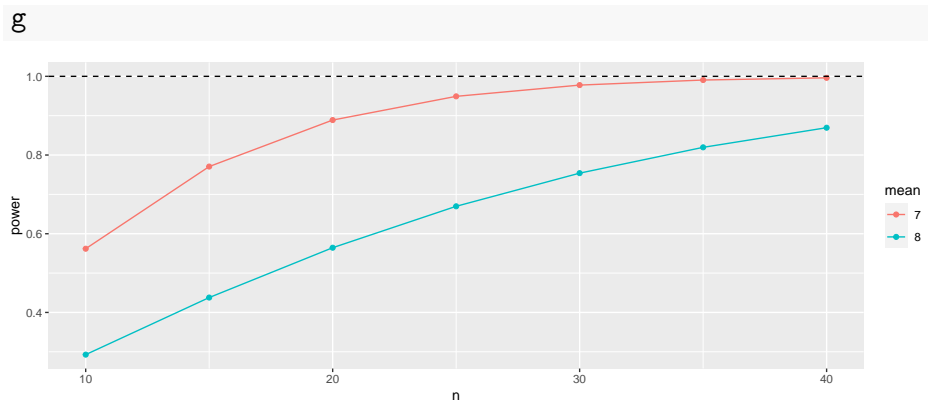
- When $\text{mean}=10$, that is, the true mean equals the null mean, H_0 is actually true, and the probability of rejecting it then is $\alpha = 0.05$.
- As the null gets more wrong (mean decreases), it becomes easier to correctly reject it.
- The blue power curve is above the red one for any mean < 10 , meaning that no matter how wrong H_0 is, you always have a greater chance of correctly rejecting it with a larger sample size.
- Previously, we had $H_0 : \mu = 10$ and a true $\mu = 8$, so a mean of 8 produces power 0.42 and 0.80 as shown on the graph.
- With $n = 30$, a true mean that is less than about 7 is almost certain to be correctly rejected. (With $n = 15$, the true mean needs to be less than 6.)

Power by sample size for means 7 and 8

Similar procedure to before:

```
means=c(7, 8)
ns=seq(10, 40, 5)
combos=crossing(mean=means, n=ns)
ans=with(combos, power.t.test(n=n, delta=10-mean, sd=4,
                             type="one.sample"))
d=tibble(mean=factor(combos$mean), n=combos$n,
          power=ans$power)
g=ggplot(d, aes(x=n, y=power, colour=mean)) +
  geom_point() + geom_line() +
  geom_hline(yintercept=1, linetype="dashed")
```

The power curves



Two-sample power

- For kids learning to read, had sample sizes of 22 (approx) in each group
- and these group SDs:

kids

group	score
t	24
t	61
t	59
t	46
t	43
t	44
t	52
t	43
t	58
t	67
+	62

Setting up

- suppose a 5-point improvement in reading score was considered important (on this scale)
- in a 2-sample test, null (difference of) mean is zero, so δ is true difference in means
- what is power for these sample sizes, and what sample size would be needed to get power up to 0.80?
- SD in both groups has to be same in power.t.test, so take as 14.

Calculating power for sample size 22 (per group)

```
power.t.test(n=22, delta=5, sd=14, type="two.sample",  
             alternative="one.sided")
```

```
##  
##      Two-sample t test power calculation  
##  
##              n = 22  
##            delta = 5  
##             sd = 14  
##      sig.level = 0.05  
##            power = 0.3158199  
## alternative = one.sided  
##  
## NOTE: n is number in *each* group
```

sample size for power 0.8

```
power.t.test(power=0.80, delta=5, sd=14, type="two.sample",  
             alternative="one.sided")
```

```
##  
##      Two-sample t test power calculation  
##  
##              n = 97.62598  
##            delta = 5  
##             sd = 14  
##      sig.level = 0.05  
##             power = 0.8  
## alternative = one.sided  
##  
## NOTE: n is number in *each* group
```

Comments

- The power for the sample sizes we have is very small (to detect a 5-point increase).
- To get power 0.80, we need 98 kids in *each* group!

Duality between confidence intervals and hypothesis tests

- Tests and CIs really do the same thing, if you look at them the right way. They are both telling you something about a parameter, and they use same things about data.
- To illustrate, some data (two groups):

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/duality.txt"  
twogroups=read_delim(my_url, " ")
```

```
## Parsed with column specification:  
## cols(  
##   y = col_double(),  
##   group = col_double()  
## )
```


The data

twogroups

y	group
10	1
11	1
11	1
13	1
13	1
14	1
14	1
15	1
16	1
13	2
13	2
14	2
17	2
18	2
19	2

95% CI (default)

```
t.test(y ~ group, data = twogroups)
```

```
##
```

```
## Welch Two Sample t-test
```

```
##
```

```
## data: y by group
```

```
## t = -2.0937, df = 8.7104, p-value = 0.0668
```

```
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## -5.5625675 0.2292342
```

```
## sample estimates:
```

```
## mean in group 1 mean in group 2
```

```
## 13.00000 15.66667
```

90% CI

```
t.test(y ~ group, data = twogroups, conf.level = 0.90)

##
##  Welch Two Sample t-test
##
## data:  y by group
## t = -2.0937, df = 8.7104, p-value = 0.0668
## alternative hypothesis: true difference in means is not equal to 0
## 90 percent confidence interval:
##  -5.010308 -0.323025
## sample estimates:
## mean in group 1 mean in group 2
##      13.00000      15.66667
```

Hypothesis test

Null is that difference in means is zero:

```
t.test(y ~ group, mu=0, data = twogroups)
```

```
##
```

```
## Welch Two Sample t-test
```

```
##
```

```
## data: y by group
```

```
## t = -2.0937, df = 8.7104, p-value = 0.0668
```

```
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## -5.5625675 0.2292342
```

```
## sample estimates:
```

```
## mean in group 1 mean in group 2
```

```
## 13.00000 15.66667
```

Comparing results

Recall null here is $H_0 : \mu_1 - \mu_2 = 0$. P-value 0.0668.

- 95% CI from -5.6 to 0.2 , contains 0 .
- 90% CI from -5.0 to -0.3 , does not contain 0 .
- At $\alpha = 0.05$, would not reject H_0 since P-value > 0.05 .
- At $\alpha = 0.10$, would reject H_0 since P-value < 0.10 .

Not just coincidence. Let $C = 100(1 - \alpha)$, so $C\%$ gives corresponding CI to level- α test. Then following always true. (\iff means “if and only if”.)

Reject H_0 at level α	\iff	$C\%$ CI does not contain H_0 value
Do not reject H_0 at level α	\iff	$C\%$ CI contains H_0 value

Idea: “Plausible” parameter value inside CI, not rejected; “Implausible” parameter value outside CI, rejected.

The value of this

- If you have a test procedure but no corresponding CI:
- you make a CI by including all the parameter values that would not be rejected by your test.
- Use:
 - $\alpha = 0.01$ for a 99% CI,
 - $\alpha = 0.05$ for a 95% CI,
 - $\alpha = 0.10$ for a 90% CI, and so on.

Testing for non-normal data

- The IRS (“Internal Revenue Service”) is the US authority that deals with taxes (like Revenue Canada).
- One of their forms is supposed to take no more than 160 minutes to complete. A citizen’s organization claims that it takes people longer than that on average.
- Sample of 30 people; time to complete form recorded.
- Read in data, and do t -test of $H_0 : \mu = 160$ vs. $H_a : \mu > 160$.
- For reading in, there is only one column, so can pretend it is delimited by anything.

Read in data

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/irs.txt"  
irs = read_csv(my_url)
```

```
## Parsed with column specification:  
## cols(  
##   Time = col_double()  
## )
```

```
irs %>% glimpse()
```

```
## Rows: 30  
## Columns: 1  
## $ Time <dbl> 91, 64, 243, 167, 123, 65, 71, 204...
```


Test whether mean is 160 or greater

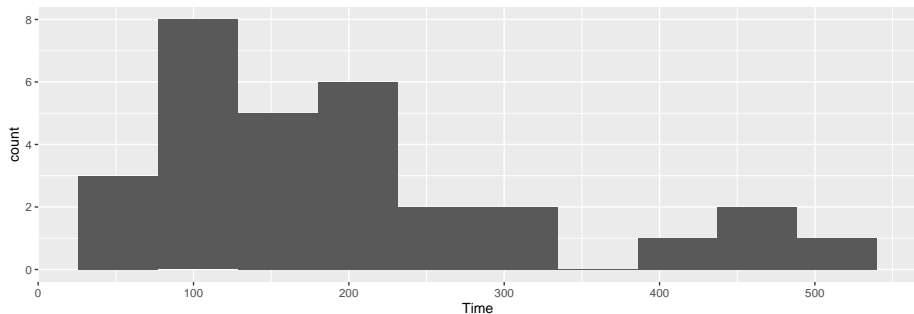
```
t.test(irs$Time, mu = 160, alternative = "greater")
```

```
##  
## One Sample t-test  
##  
## data:  irs$Time  
## t = 1.8244, df = 29, p-value = 0.03921  
## alternative hypothesis: true mean is greater than 160  
## 95 percent confidence interval:  
##  162.8305      Inf  
## sample estimates:  
## mean of x  
##  201.2333
```

Reject null; mean (for all people to complete form) greater than 160.

But, look at a graph

```
ggplot(irs, aes(x = Time)) + geom_histogram(bins = 10)
```



Comments

- Skewed to right.
- Should look at *median*, not mean.

The sign test

- But how to test whether the median is greater than 160?
- Idea: if the median really is 160 (H_0 true), the sampled values from the population are equally likely to be above or below 160.
- If the population median is greater than 160, there will be a lot of sample values greater than 160, not so many less. Idea: test statistic is number of sample values greater than hypothesized median.
- How to decide whether “unusually many” sample values are greater than 160? Need a sampling distribution.
- If H_0 true, pop. median is 160, then each sample value independently equally likely to be above or below 160.
- So number of observed values above 160 has binomial distribution with $n = 30$ (number of data values) and $p = 0.5$ (160 is hypothesized to be *median*).

Obtaining P-value for sign test 1/2

- Count values above/below 160:

```
irs %>% count(Time > 160)
```

Time > 160	n
FALSE	13
TRUE	17

- 17 above, 13 below. How unusual is that? Need a *binomial table*.

Obtaining P-value for sign test 2/2

- R function `dbinom` gives the probability of eg. exactly 17 successes in a binomial with $n = 30$ and $p = 0.5$:

```
dbinom(17, 30, 0.5)
```

```
## [1] 0.1115351
```

- but we want probability of 17 *or more*, so get all of those, find probability of each, and add them up:

```
tibble(x=17:30) %>%  
  mutate(prob=dbinom(x, 30, 0.5)) %>%  
  summarize(total=sum(prob))
```

total
0.2923324

Using my package smmr

- I wrote a package `smmr` to do the sign test (and some other things). Installation is a bit fiddly:
 - Install devtools with `install.packages("devtools")`
 - then install `smmr`:

```
library(devtools)
install_github("nxskok/smmr")
```

- Then load it:

```
library(smmr)
```

smmr for sign test

- smmr's function `sign_test` needs three inputs: a data frame, a column and a null median:

```
sign_test(irs, Time, 160)
```

```
## $above_below
## below above
##      13      17
##
## $p_values
##   alternative   p_value
## 1         lower 0.8192027
## 2         upper 0.2923324
## 3    two-sided 0.5846647
```


Comments (1/3)

- Testing whether population median *greater than* 160, so want *upper-tail* P-value 0.2923. Same as before.
- Also get table of values above and below; this too as we got.

Comments (2/3)

- P-values are:

Test	P-value
t	0.0392
Sign	0.2923

- These are very different: we reject a mean of 160 (in favour of the mean being bigger), but clearly *fail* to reject a median of 160 in favour of a bigger one.
- Why is that? Obtain mean and median:

```
irs %>% summarize(mean = mean(Time), median = median(Time))
```

mean	median
201.2333	172.5

Comments (3/3)

- The mean is pulled a long way up by the right skew, and is a fair bit bigger than 160.
- The median is quite close to 160.
- We ought to be trusting the sign test and not the t-test here (median and not mean), and therefore there is no evidence that the “typical” time to complete the form is longer than 160 minutes.
- Having said that, there are clearly some people who take a lot longer than 160 minutes to complete the form, and the IRS could focus on simplifying its form for these people.
- In this example, looking at any kind of average is not really helpful; a better question might be “do an unacceptably large fraction of people take longer than (say) 300 minutes to complete the form?”: that is, thinking about worst-case rather than average-case.

Confidence interval for the median

- The sign test does not naturally come with a confidence interval for the median.
- So we use the “duality” between test and confidence interval to say: the (95%) confidence interval for the median contains exactly those values of the null median that would not be rejected by the two-sided sign test (at $\alpha = 0.05$).

For our data

- The procedure is to try some values for the null median and see which ones are inside and which outside our CI.
- `smmr` has `pval_sign` that gets just the 2-sided P-value:

```
pval_sign(160, irs, Time)
```

```
## [1] 0.5846647
```

- Try a couple of null medians:

```
pval_sign(200, irs, Time)
```

```
## [1] 0.3615946
```

```
pval_sign(300, irs, Time)
```

```
## [1] 0.001430906
```

- So 200 inside the 95% CI and 300 outside.

Doing a whole bunch

- Choose our null medians first:

```
(d=tibble(null_median=seq(100,300,20)))
```

null_median
100
120
140
160
180
200
220
240
260
280
300

... and then

“for each null median, run the function `pval_sign` for that null median and get the P-value”:

```
d %>% mutate(p_value = map_dbl(null_median,
                                ~ pval_sign(., irs, Time)))
```

null_median	p_value
100	0.0003249
120	0.0987371
140	0.2004884
160	0.5846647
180	0.8555356
200	0.3615946
220	0.0427739
240	0.0161248
260	0.0052229
280	0.0014200

Make it easier for ourselves

```
d %>%
```

```
  mutate(p_value = map_dbl(null_median,
                           ~ pval_sign(., irs, Time))) %>%
  mutate(in_out = ifelse(p_value > 0.05, "inside", "outside"))
```

null_median	p_value	in_out
100	0.0003249	outside
120	0.0987371	inside
140	0.2004884	inside
160	0.5846647	inside
180	0.8555356	inside
200	0.3615946	inside
220	0.0427739	outside
240	0.0161248	outside
260	0.0052229	outside
280	0.0014309	outside

confidence interval for median?

- 95% CI to this accuracy from 120 to 200.
- Can get it more accurately by looking more closely in intervals from 100 to 120, and from 200 to 220.

A more efficient way: bisection

- Know that top end of CI between 200 and 220:

```
lo=200  
hi=220
```

- Try the value halfway between: is it inside or outside?

```
(try = (lo + hi) / 2)
```

```
## [1] 210
```

```
pval_sign(try,irs,Time)
```

```
## [1] 0.09873715
```

- Inside, so upper end is between 210 and 220. Repeat (over):

... bisection continued

```
lo = try  
(try = (lo + hi) / 2)
```

```
## [1] 215
```

```
pval_sign(try, irs, Time)
```

```
## [1] 0.06142835
```

- 215 is inside too, so upper end between 215 and 220.
- Continue until have as accurate a result as you want.

Bisection automatically

- A loop, but not a for since we don't know how many times we're going around. Keep going while a condition is true:

```
lo = 200
hi = 220
while (hi - lo > 1) {
  try = (hi + lo) / 2
  ptry = pval_sign(try, irs, Time)
  print(c(try, ptry))
  if (ptry <= 0.05)
    hi = try
  else
    lo = try
}
```

The output from this loop

```
## [1] 210.00000000    0.09873715
## [1] 215.00000000    0.06142835
## [1] 217.50000000    0.04277395
## [1] 216.25000000    0.04277395
## [1] 215.62500000    0.04277395
```

- 215 inside, 215.625 outside. Upper end of interval to this accuracy is 215.

Using smmr

- smmr has function `ci_median` that does this (by default 95% CI):

```
ci_median(irs,Time)
```

```
## [1] 119.0065 214.9955
```

- Uses a more accurate bisection than we did.
- Or get, say, 90% CI for median:

```
ci_median(irs,Time,conf.level=0.90)
```

```
## [1] 123.0031 208.9960
```

- 90% CI is shorter, as it should be.

Matched pairs

Some data:

subject	druga	drugb
1	2.0	3.5
2	3.6	5.7
3	2.6	2.9
4	2.6	2.4
5	7.3	9.9
6	3.4	3.3
7	14.9	16.7
8	6.6	6.0
9	2.3	3.8
10	2.0	4.0
11	6.8	9.1
12	8.5	20.9

Matched pairs data

- Data are comparison of 2 drugs for effectiveness at reducing pain.
- 12 subjects (cases) were arthritis sufferers
- Response is #hours of pain relief from each drug.
- In reading example, each child tried only one reading method.
- But here, each subject tried out both drugs, giving us two measurements.
- Possible because, if you wait long enough, one drug has no influence over effect of other.
- Advantage: focused comparison of drugs. Compare one drug with another on same person, removes a lot of variability due to differences between people.
- Matched pairs, requires different analysis.
- Design: randomly choose 6 of 12 subjects to get drug A first, other 6 get drug B first.

Paired t test: reading the data

Values aligned in columns:

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/analgesic.txt"  
pain=read_table(my_url)
```

```
## Parsed with column specification:  
## cols(  
##   subject = col_double(),  
##   druga = col_double(),  
##   drugb = col_double()  
## )
```

The data

pain

subject	druga	drugb
1	2.0	3.5
2	3.6	5.7
3	2.6	2.9
4	2.6	2.4
5	7.3	9.9
6	3.4	3.3
7	14.9	16.7
8	6.6	6.0
9	2.3	3.8
10	2.0	4.0
11	6.8	9.1
12	8.5	20.9

Paired t -test

```
with(pain, t.test(druga, drugb, paired = T))

##
## Paired t-test
##
## data:  druga and drugb
## t = -2.1677, df = 11, p-value = 0.05299
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.29941513  0.03274847
## sample estimates:
## mean of the differences
##                -2.133333
```

P-value is 0.053. Likewise, you can calculate the differences yourself and do a 1-sample t -test on them, over:

t-testing the differences

- First calculate a column of differences (in data frame):

```
(pain %>% mutate(diff=druga-drugb) -> pain)
```

subject	druga	drugb	diff
1	2.0	3.5	-1.5
2	3.6	5.7	-2.1
3	2.6	2.9	-0.3
4	2.6	2.4	0.2
5	7.3	9.9	-2.6
6	3.4	3.3	0.1
7	14.9	16.7	-1.8
8	6.6	6.0	0.6
9	2.3	3.8	-1.5
10	2.0	4.0	-2.0
11	6.8	9.1	-2.3
12	0.5	22.0	10.4

t-test on the differences

- then throw them into `t.test`, testing that the mean is zero, with same result as before:

```
with(pain,t.test(diff,mu=0))
```

```
##  
## One Sample t-test  
##  
## data: diff  
## t = -2.1677, df = 11, p-value = 0.05299  
## alternative hypothesis: true mean is not equal to 0  
## 95 percent confidence interval:  
## -4.29941513 0.03274847  
## sample estimates:  
## mean of x  
## -2.133333
```

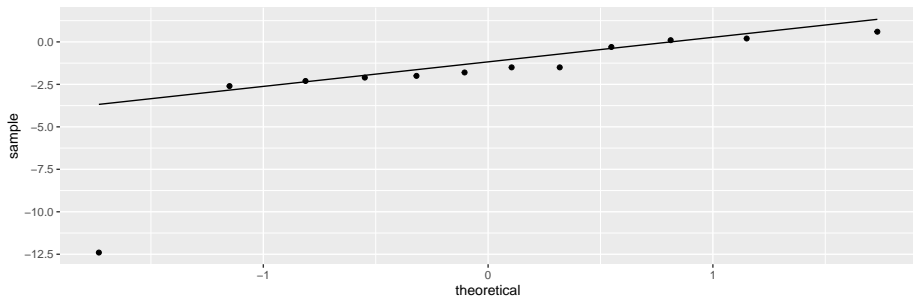
Assessing normality

- 1-sample and 2-sample t-tests assume (each) group normally distributed.
- Matched pairs analyses assume (theoretically) that differences normally distributed.
- Though we know that t-tests generally behave well even without normality.
- How to assess normality? A normal quantile plot.
 - Idea: scatter of points should follow the straight line, without curving.
 - Outliers show up at bottom left or top right of plot as points off the line.

The normal quantile plot

- of differences from matched pairs data

```
ggplot(pain, aes(sample=diff)) + stat_qq() + stat_qq_line()
```



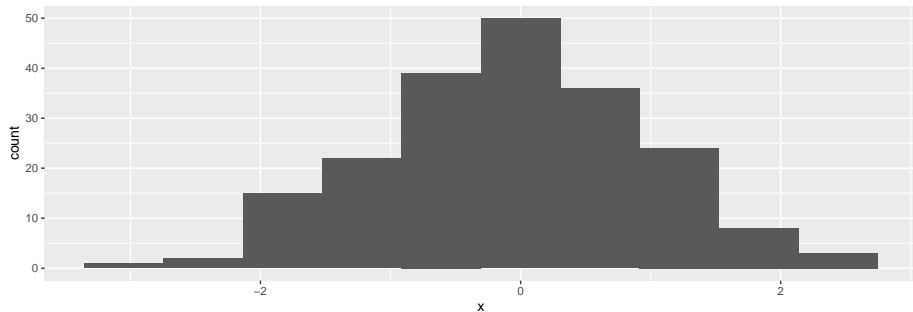
- Points should follow the straight line. Bottom left one way off, so normality questionable here: outlier.

More normal quantile plots

- How straight does a normal quantile plot have to be?
- There is randomness in real data, so even a normal quantile plot from normal data won't look perfectly straight.
- With a small sample, can look not very straight even from normal data.
- Looking for systematic departure from a straight line; random wiggles ought not to concern us.
- Look at some examples where we know the answer, so that we can see what to expect.

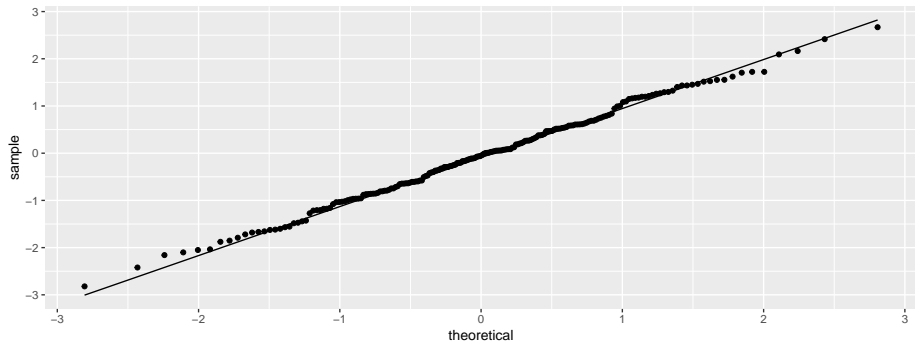
Normal data, large sample

```
d=tibble(x=rnorm(200))  
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

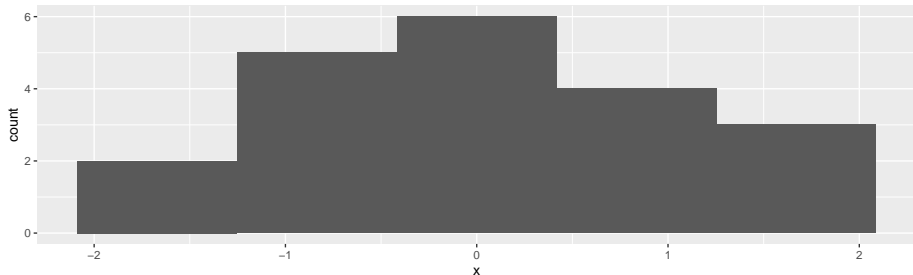
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Normal data, small sample

- Not so convincingly normal, but not obviously skewed:

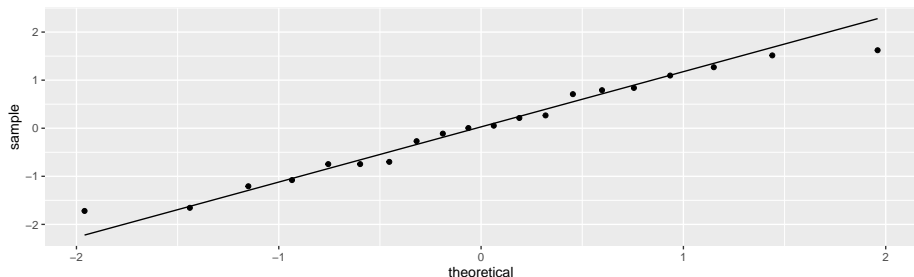
```
d=tibble(x=rnorm(20))  
ggplot(d,aes(x=x))+geom_histogram(bins=5)
```



The normal quantile plot

Good, apart from the highest and lowest points being slightly off. I'd call this good:

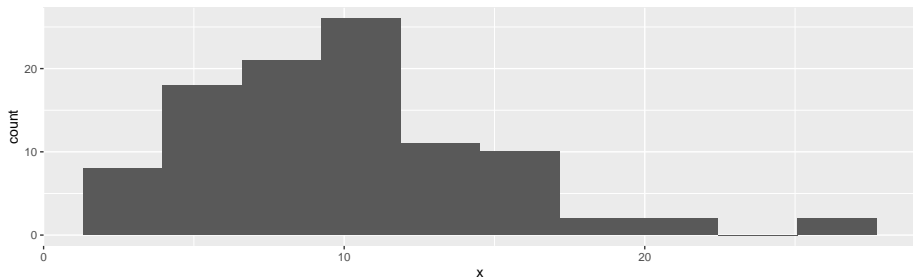
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Chi-squared data, $df = 10$

Somewhat skewed to right:

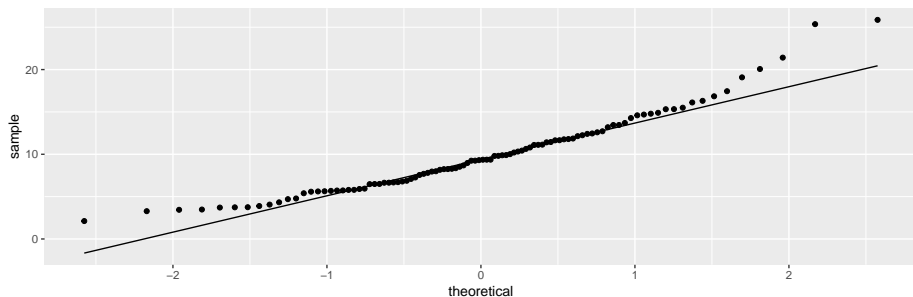
```
d=tibble(x=rchisq(100,10))  
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

Somewhat opening-up curve:

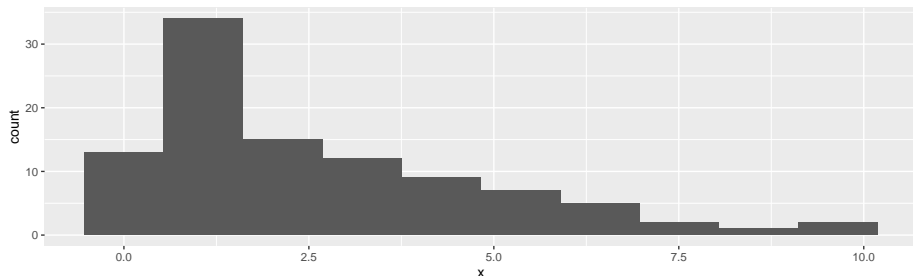
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Chi-squared data, $df = 3$

Definitely skewed to right:

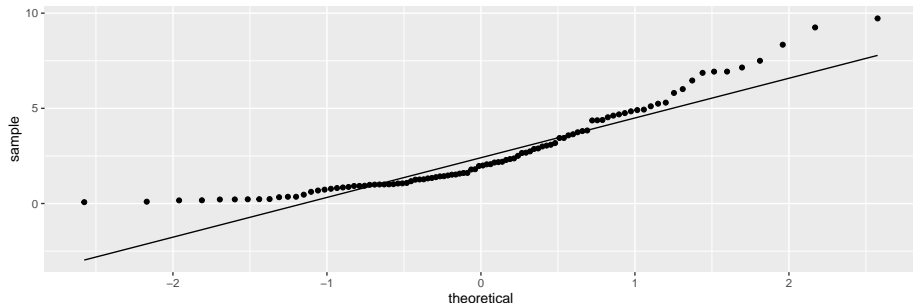
```
d=tibble(x=rchisq(100,3))  
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

Clear upward-opening curve:

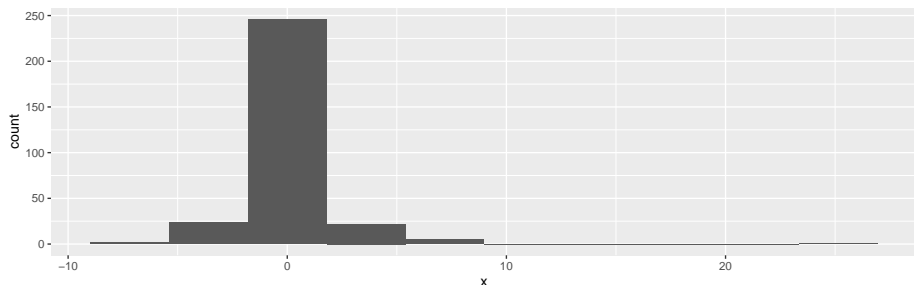
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



t-distributed data, $df = 3$

Long tails (or a very sharp peak):

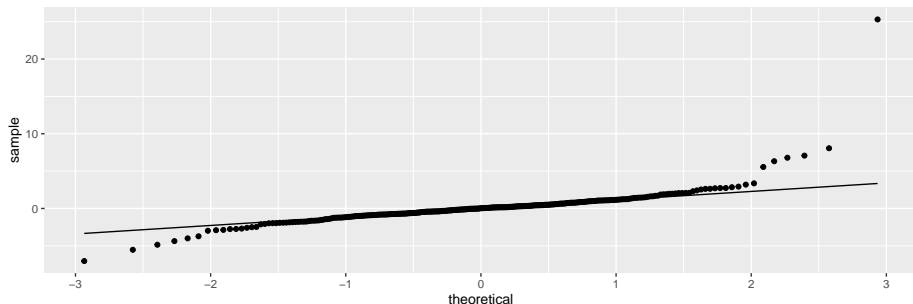
```
d=tibble(x=rt(300,3))  
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

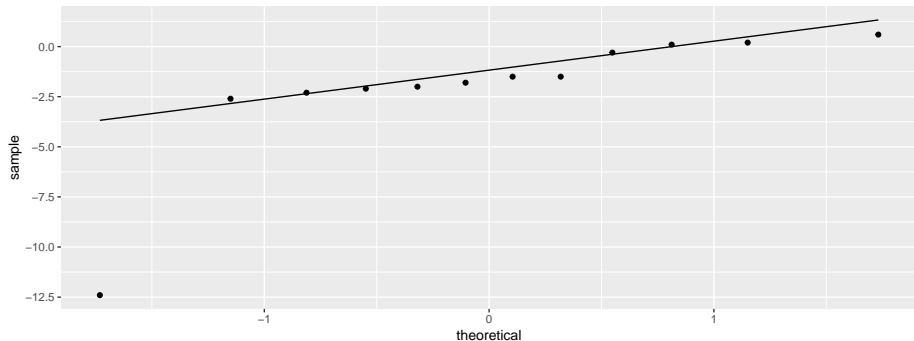
Low values too low and high values too high for normal.

```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Our pain-relief data

```
ggplot(pain, aes(sample=diff)) + stat_qq() + stat_qq_line()
```



Comments

- Definitely not normal. What to do?
- Sign test on differences, null median 0.

Sign test

- Most easily: calculate differences in data frame, then use `sgn.test`.
- Null median difference is 0:

```
data %>% mutate(mydiff=drugA-drugB) %>%
sgn.test(mydiff,0)
```

```
## $above_below
## below above
##      9      3
##
## $p_values
##   alternative    p_value
## 1         lower 0.07299805
## 2          upper 0.98071289
## 3    two-sided 0.14599609
```

Comments

- P-value 0.1460. No evidence that the drugs are different.
- Since we are working in a pipeline, input data frame to `sign_test` is “whatever came out of previous step”.

(Some of) the kids' reading data, again

```
kids %>% sample_n(12)
```

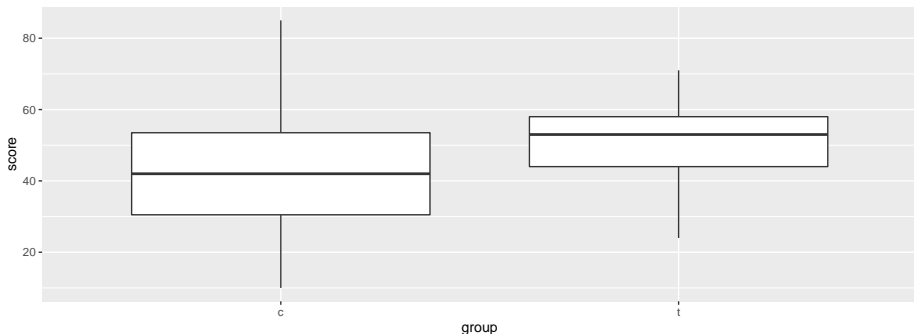
group	score
t	49
c	48
c	46
c	17
t	43
t	44
c	26
t	59
c	55
t	62
t	43
c	10

Where we are at

- 21 kids in “treatment”, new reading method; 23 in “control”, standard reading method.
- Assessing assumptions:
 - We did two-sample t-test (Satterthwaite-Welch) before.
 - Assumes approx. normal data within each group.
 - Does not assume equal spread.
 - (Pooled t-test *does* assume equal spread).
 - Assess each group separately.

Boxplots for reading data

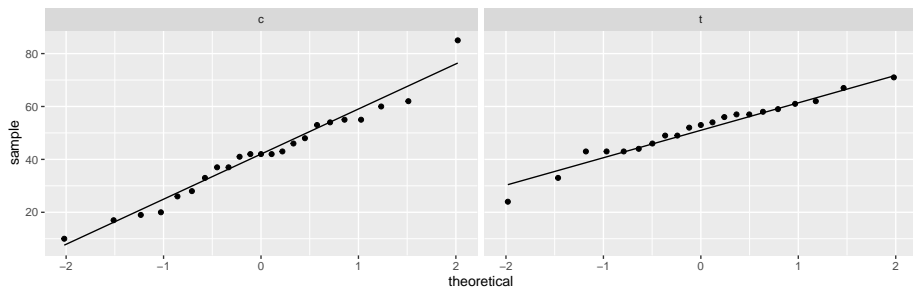
```
ggplot(kids, aes(x=group, y=score)) + geom_boxplot()
```



Facetted normal quantile plots

Done this way:

```
ggplot(kids, aes(sample=score)) + stat_qq() + stat_qq_line() +  
facet_wrap(~group)
```



Comments

- These plots show no problems with normality. Both groups are more or less symmetric/normal and there are no outliers.
- Equal spreads questionable, but we don't need that.
- Assess equal spreads by looking at *slopes* of normal quantile plots.
- We ought be happy with the (Welch) two-sample t-test (over)

Welch two-sample test

```
t.test(score~group,data=kids,alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  score by group
## t = -2.3109, df = 37.855, p-value = 0.01319
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##      -Inf -2.691293
## sample estimates:
## mean in group c mean in group t
##      41.52174      51.47619
```

from which we concluded that the new reading method really does help.

What to do if normality fails

- (On the previous page, the only indication of non-normality is the highest score in the control group, which is a little too high for normality.)
- If normality fails (for one or both of the groups), what do we do then?
- Again, can compare medians: use the thought process of the sign test, which does not depend on normality and is not damaged by outliers.
- A suitable test called Mood's median test.
- Before we get to that, a diversion.

The chi-squared test for independence

Suppose we want to know whether people are in favour of having daylight savings time all year round. We ask 20 males and 20 females whether they each agree with having DST all year round ("yes") or not ("no"). Some of the data:

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/dst.txt"
dst=read_delim(my_url," ")
dst %>% sample_n(5) # randomly sample 5 rows
```

gender	agree
female	no
female	no
male	no
male	yes
female	yes

... continued

Count up individuals in each category combination, and arrange in contingency table:

```
tab=with(dst, table(gender, agree))
tab
```

```
##           agree
## gender    no  yes
##  female  11   9
##   male   3   17
```

- Most of the males say “yes”, but the females are about evenly split.
- Looks like males more likely to say “yes”, ie. an association between gender and agreement.
- Test an H_0 of “no association” (“independence”) vs. alternative that there is really some association.
- Done with `chisq.test`.

...And finally

```
chisq.test(tab,correct=F)
```

```
##
```

```
##  Pearson's Chi-squared test
```

```
##
```

```
## data:  tab
```

```
## X-squared = 7.033, df = 1, p-value = 0.008002
```

- Reject null hypothesis of no association
- therefore there is a difference in rates of agreement between (all) males and females (or that gender and agreement are associated).
- Without `correct=F` uses “Yates correction”; this way, should give same answers as calculated by hand (if you know how).

Mood's median test

- Before our diversion, we wanted to compare medians of two groups.
- Recall sign test: count number of values above and below something (there, hypothesized median).
- Idea of Mood's median test:
 - Work out the median of all the data, regardless of group ("grand median").
 - Count how many data values in each group are above/below this grand median.
 - Make contingency table of group vs. above/below.
 - Test for association.
- If group medians equal, each group should have about half its observations above/below grand median. If not, one group will be mostly above grand median and other below.

Mood's median test for reading data

- Find overall median score:

```
(kids %>% summarize(med=median(score)) %>% pull(med) -> m)
```

```
## [1] 47
```

- Make table of above/below vs. group:

```
tab=with(kids, table(group, score>m))
tab
```

```
##
```

```
## group FALSE TRUE
```

```
##      c      15      8
```

```
##      t       7     14
```

- Treatment group scores mostly above median, control group scores mostly below, as expected.

The test

- Do chi-squared test:

```
chisq.test(tab,correct=F)
```

```
##
```

```
## Pearson's Chi-squared test
```

```
##
```

```
## data:  tab
```

```
## X-squared = 4.4638, df = 1, p-value = 0.03462
```

- This test actually two-sided (tests for any association).
- Here want to test that new reading method *better* (one-sided).
- Most of treatment children above overall median, so do 1-sided test by halving P-value to get 0.017.
- This way too, children do better at learning to read using the new method.

Or by smmr

- `median_test` does the whole thing:

```
median_test(kids,score,group)
```

```
## $table
##      above
## group above below
##      c      8      15
##      t     14       7
##
## $test
##      what      value
## 1 statistic 4.46376812
## 2          df 1.00000000
## 3    P-value 0.03462105
```

- P-value again two-sided.

Comments

- P-value 0.013 for (1-sided) t -test, 0.017 for (1-sided) Mood median test.
- Like the sign test, Mood's median test doesn't use the data very efficiently (only, is each value above or below grand median).
- Thus, if we can justify doing t -test, we should do it. This is the case here.
- The t -test will usually give smaller P-value because it uses the data more efficiently.
- The time to use Mood's median test is if we are definitely unhappy with the normality assumption (and thus the t -test P-value is not to be trusted).

Jumping rats

- Link between exercise and healthy bones (many studies).
- Exercise stresses bones and causes them to get stronger.
- Study (Purdue): effect of jumping on bone density of growing rats.
- 30 rats, randomly assigned to 1 of 3 treatments:
 - No jumping (control)
 - Low-jump treatment (30 cm)
 - High-jump treatment (60 cm)
- 8 weeks, 10 jumps/day, 5 days/week.
- Bone density of rats (mg/cm^3) measured at end.
- See whether larger amount of exercise (jumping) went with higher bone density.
- Random assignment: rats in each group similar in all important ways.
- So entitled to draw conclusions about cause and effect.

Reading the data

Values separated by spaces:

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/jumping.txt"  
rats=read_delim(my_url, " ")
```

```
## Parsed with column specification:
```

```
## cols(
```

```
##   group = col_character(),
```

```
##   density = col_double()
```

```
## )
```

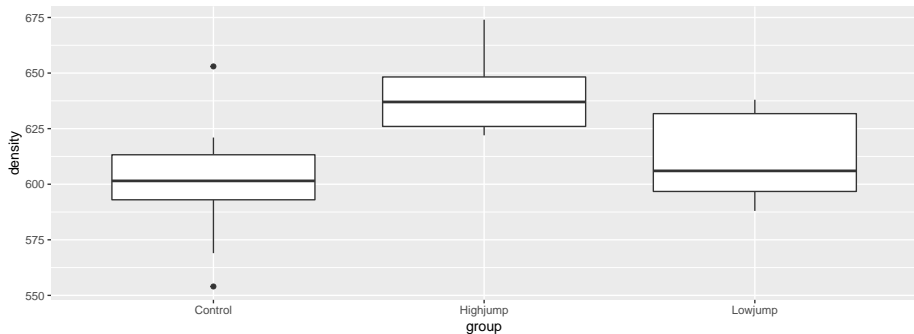
The data (some random rows)

```
rats %>% sample_n(12)
```

group	density
Highjump	650
Lowjump	599
Lowjump	588
Lowjump	632
Highjump	643
Lowjump	638
Highjump	643
Control	621
Highjump	626
Control	554
Lowjump	635
Lowjump	607

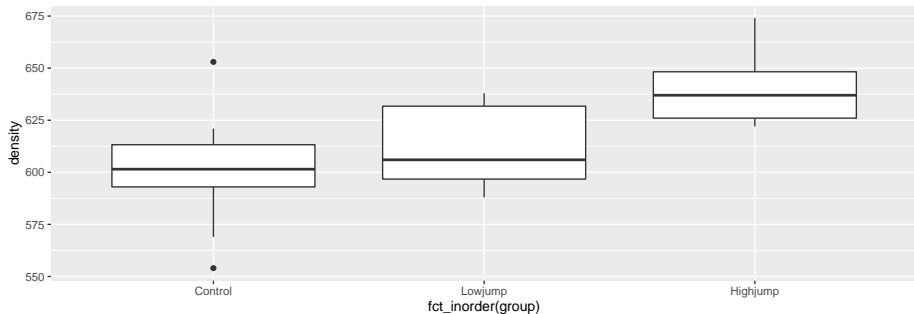
Boxplots

```
ggplot(rats, aes(y=density, x=group)) + geom_boxplot()
```



Or, arranging groups in data (logical) order

```
ggplot(rats, aes(y=density, x=fct_inorder(group))) +  
geom_boxplot()
```



Analysis of Variance

- Comparing > 2 groups of independent observations (each rat only does one amount of jumping).
- Standard procedure: analysis of variance (ANOVA).
- Null hypothesis: all groups have same mean.
- Alternative: “not all means the same”, at least one is different from others.

Testing: ANOVA in R

```
rats.aov=aov(density~group,data=rats)
summary(rats.aov)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## group         2    7434     3717   7.978 0.0019 **
## Residuals    27   12579       466
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Usual ANOVA table, small P-value: significant result.
- Conclude that the mean bone densities are not all equal.
- Reject null, but not very useful finding.

Which groups are different from which?

- ANOVA really only answers half our questions: it says “there are differences”, but doesn’t tell us which groups different.
- One possibility (not the best): compare all possible pairs of groups, via two-sample t.
- First pick out each group:

```
rats %>% filter(group=="Control") -> controls  
rats %>% filter(group=="Lowjump") -> lows  
rats %>% filter(group=="Highjump") -> highs
```

Control vs. low

```
t.test(controls$density, lows$density)
```

```
##
```

```
## Welch Two Sample t-test
```

```
##
```

```
## data: controls$density and lows$density
```

```
## t = -1.0761, df = 16.191, p-value = 0.2977
```

```
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## -33.83725 11.03725
```

```
## sample estimates:
```

```
## mean of x mean of y
```

```
## 601.1 612.5
```

No sig. difference here.

Control vs. high

```
t.test(controls$density, highs$density)
```

```
##
```

```
## Welch Two Sample t-test
```

```
##
```

```
## data: controls$density and highs$density
```

```
## t = -3.7155, df = 14.831, p-value = 0.002109
```

```
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## -59.19139 -16.00861
```

```
## sample estimates:
```

```
## mean of x mean of y
```

```
##      601.1      638.7
```

These are different.

Low vs. high

```
t.test( lows$density, highs$density )
```

```
##  
##  Welch Two Sample t-test  
##  
## data:  lows$density and highs$density  
## t = -3.2523, df = 17.597, p-value = 0.004525  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
##  -43.15242  -9.24758  
## sample estimates:  
## mean of x mean of y  
##      612.5      638.7
```

These are different too.

But...

- We just did 3 tests instead of 1.
- So we have given ourselves 3 chances to reject H_0 : all means equal, instead of 1.
- Thus α for this combined test is not 0.05.

John W. Tukey



- American statistician, 1915–2000
- Big fan of exploratory data analysis
- Invented boxplot
- Invented "honestly significant differences"
- Invented jackknife estimation
- Coined computing term "bit"
- Co-inventor of Fast Fourier Transform

Honestly Significant Differences

- Compare several groups with one test, telling you which groups differ from which.
- Idea: if all population means equal, find distribution of highest sample mean minus lowest sample mean.
- Any means unusually different compared to that declared significantly different.

Tukey on rat data

```
rats.aov=aov(density~group,data=rats)
TukeyHSD(rats.aov)
```

```
##    Tukey multiple comparisons of means
##      95% family-wise confidence level
##
## Fit: aov(formula = density ~ group, data = rats)
##
## $group
```

	diff	lwr	upr	p adj
Highjump-Control	37.6	13.66604	61.533957	0.0016388
Lowjump-Control	11.4	-12.53396	35.333957	0.4744032
Lowjump-Highjump	-26.2	-50.13396	-2.266043	0.0297843

- Again conclude that bone density for highjump group significantly higher than for other two groups.

Why Tukey's procedure better than all t-tests

Look at P-values for the two tests:

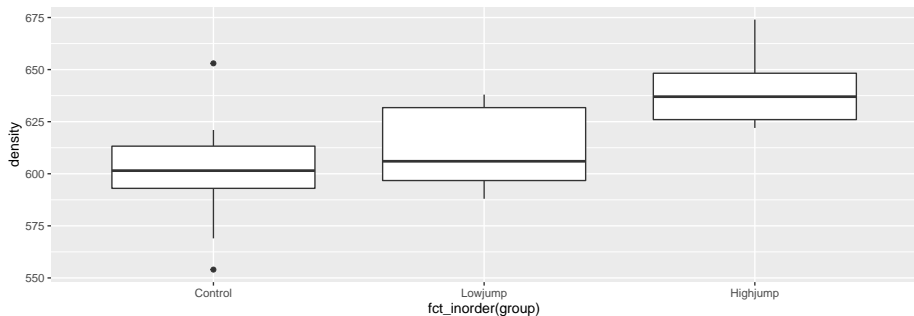
Comparison	Tukey	t-tests

Highjump-Control	0.0016	0.0021
Lowjump-Control	0.4744	0.2977
Lowjump-Highjump	0.0298	0.0045

- Tukey P-values (mostly) higher.
- Proper adjustment for doing three t-tests at once, not just one in isolation.
- lowjump-highjump comparison would no longer be significant at $\alpha = 0.01$.

Checking assumptions

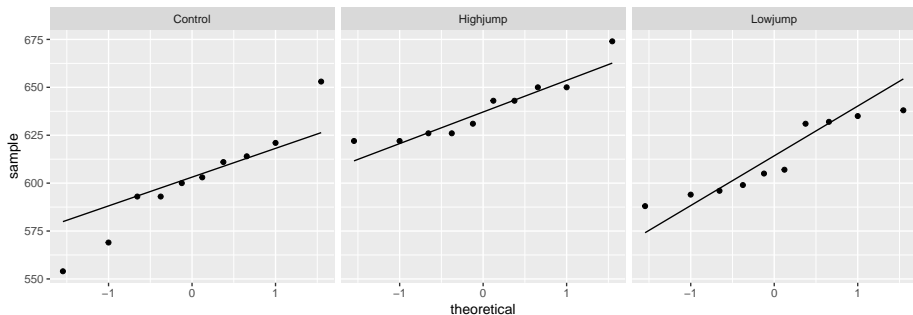
```
ggplot(rats, aes(y=density, x=fct_inorder(group))) +  
geom_boxplot()
```



Assumptions: - Normally distributed data within each group - with equal group SDs.

Normal quantile plots by group

```
ggplot(rats, aes(sample = density)) + stat_qq() +  
  stat_qq_line() + facet_wrap( ~ group)
```



The assumptions

- Normally-distributed data within each group
- Equal group SDs. These are shaky here because:
 - control group has outliers
 - highjump group appears to have less spread than others. Possible remedies (in general):
- Transformation of response (usually works best when SD increases with mean)
- If normality OK but equal spreads not, can use Welch ANOVA. (Regular ANOVA like pooled t-test; Welch ANOVA like Welch-Satterthwaite t-test.)
- Can also use Mood's Median Test (see over). This works for any number of groups.

Mood's median test 1/4

- Find median of all bone densities, regardless of group:

```
(rats %>% summarize(med = median(density)) %>% pull(med) -> m)
```

```
## [1] 621.5
```

- Count up how many observations in each group above or below overall median:

```
tab = with(rats, table(group, density > m))
tab
```

```
##
```

```
## group      FALSE TRUE
```

```
## Control      9    1
```

```
## Highjump     0   10
```

```
## Lowjump      6    4
```

Mood's median test 2/4

```
tab
```

```
##
```

```
## group      FALSE TRUE
```

```
##   Control      9    1
```

```
##   Highjump     0   10
```

```
##   Lowjump      6    4
```

- All Highjump obs above overall median.
- Most Control obs below overall median.
- Suggests medians differ by group.

Mood's median test 3/4

- Test whether association between group and being above/below overall median significant using chi-squared test for association:

```
chisq.test(tab,correct=F)
```

```
##
```

```
## Pearson's Chi-squared test
```

```
##
```

```
## data:  tab
```

```
## X-squared = 16.8, df = 2, p-value = 0.0002249
```

- Very small P-value says that being above/below overall median depends on group.
- That is, groups do not all have same median.

Mood's median test 4/4

Or with `median_test` from `smmr`, same as before.

```
median_test(rats,density,group)
```

```
## $table
##           above
## group      above below
## Control      1      9
## Highjump     10      0
## Lowjump       4      6
##
## $test
##           what           value
## 1 statistic 1.680000e+01
## 2          df 2.000000e+00
## 3    P-value 2.248673e-04
```

Comments

- No doubt that medians differ between groups (not all same).
- This test is equivalent of F -test, not of Tukey.
- To determine which groups differ from which, can compare all possible pairs of groups via (2-sample) Mood's median tests, then adjust P-values by multiplying by number of 2-sample Mood tests done (Bonferroni):

```
pairwise_median_test(rats,density,group)
```

g1	g2	p_value	adj_p_value
Control	Highjump	0.0001478	0.0004434
Control	Lowjump	0.3710934	1.0000000
Highjump	Lowjump	0.3710934	1.0000000

- Now, lowjump-highjump difference no longer significant.

Welch ANOVA

- For these data, Mood's median test probably best because we doubt both normality and equal spreads.
- When normality OK but spreads differ, Welch ANOVA way to go.
- Welch ANOVA done by `oneway.test` as shown (for illustration):

```
oneway.test(density~group,data=rats)
```

```
##  
## One-way analysis of means (not assuming  
## equal variances)  
##  
## data: density and group  
## F = 8.8164, num df = 2.000, denom df =  
## 17.405, p-value = 0.002268
```

- P-value very similar, as expected.
- Appropriate Tukey-equivalent here called Games-Howell.

Games-Howell

- Lives in package PMCMRplus (also userfriendlyscience). Install first.

```
library(PMCMRplus)
```

```
gamesHowellTest(density~factor(group),data=rats)
```

```
##
```

```
## Pairwise comparisons using Games-Howell test
```

```
## data: density by factor(group)
```

```
##           Control Highjump
```

```
## Highjump 0.0056  -
```

```
## Lowjump  0.5417  0.0120
```

```
##
```

```
## P value adjustment method: none
```