STAC32: Statistical Inference

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 equ
## 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 equ
## 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 equ
## 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

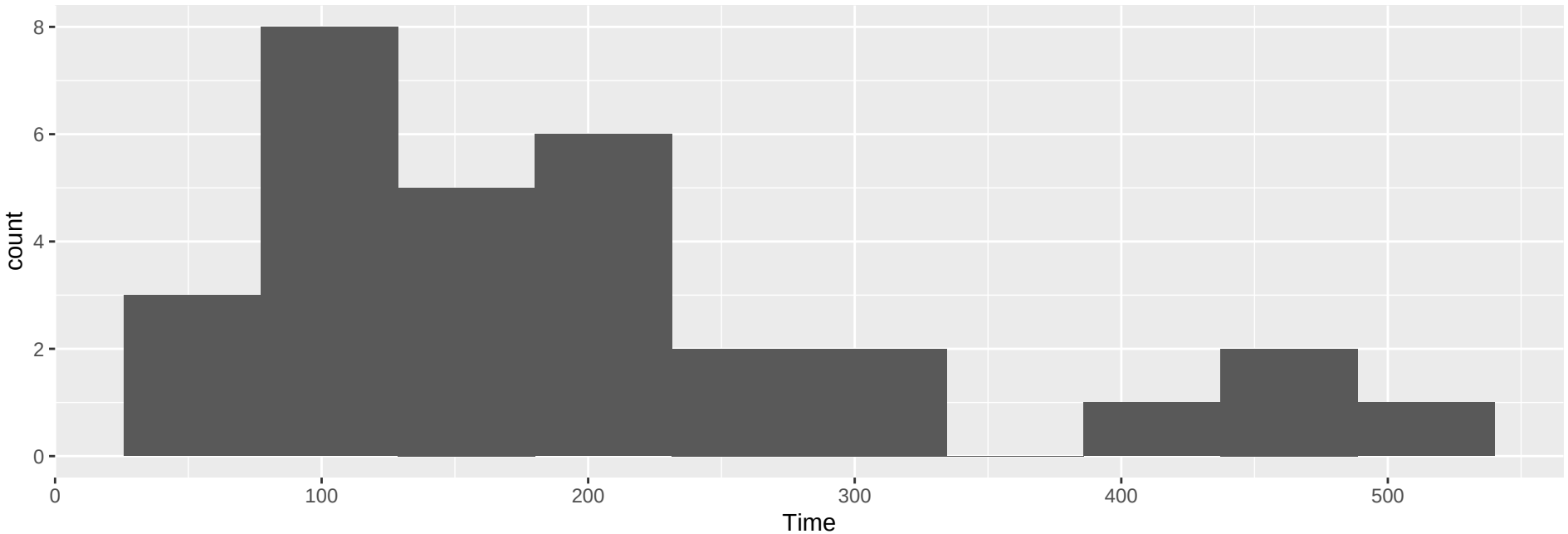
```
t.test(irs$Time, mu = 160, alternative = "greater")
```

```
##
##  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.t
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 equ
## 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	8.5	20.0	12.4

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
```

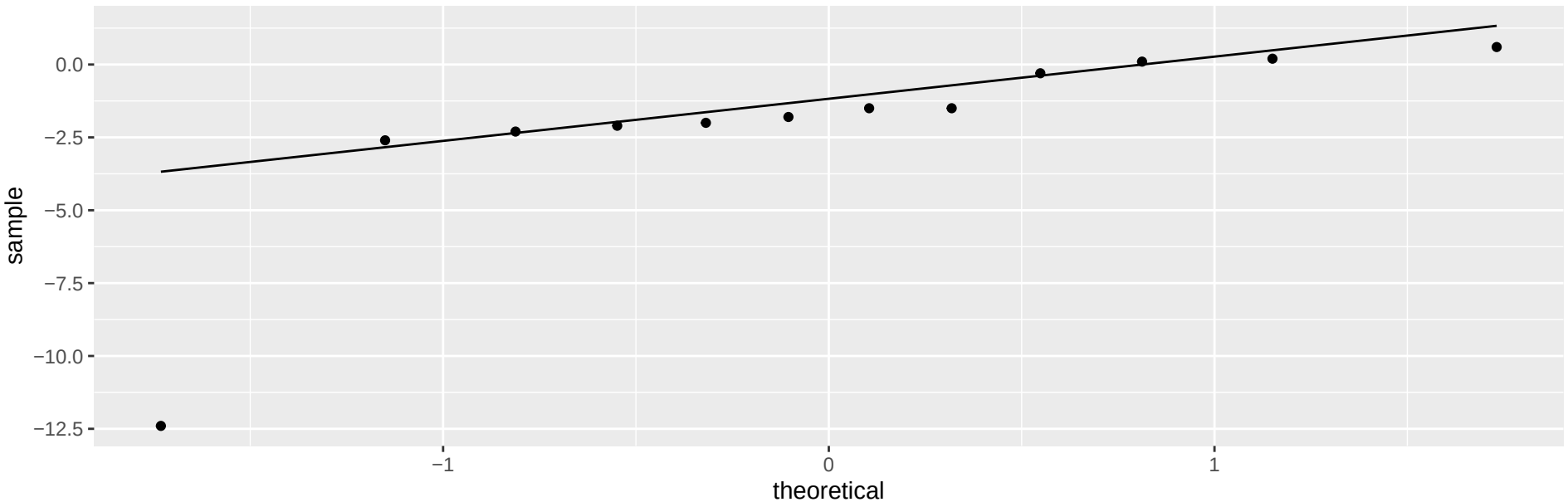
Assessing normality

- 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()
```



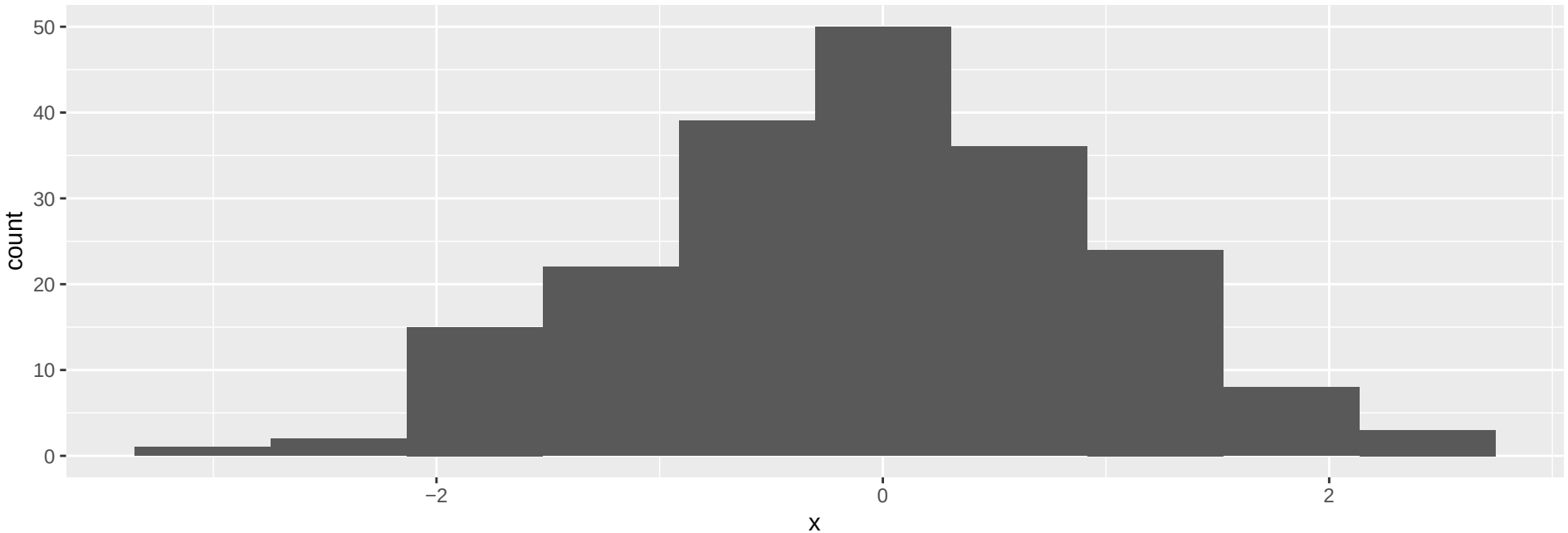
- Points should follow the straight line. Bottom left one way off, so normality questionable here: outlier.

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.

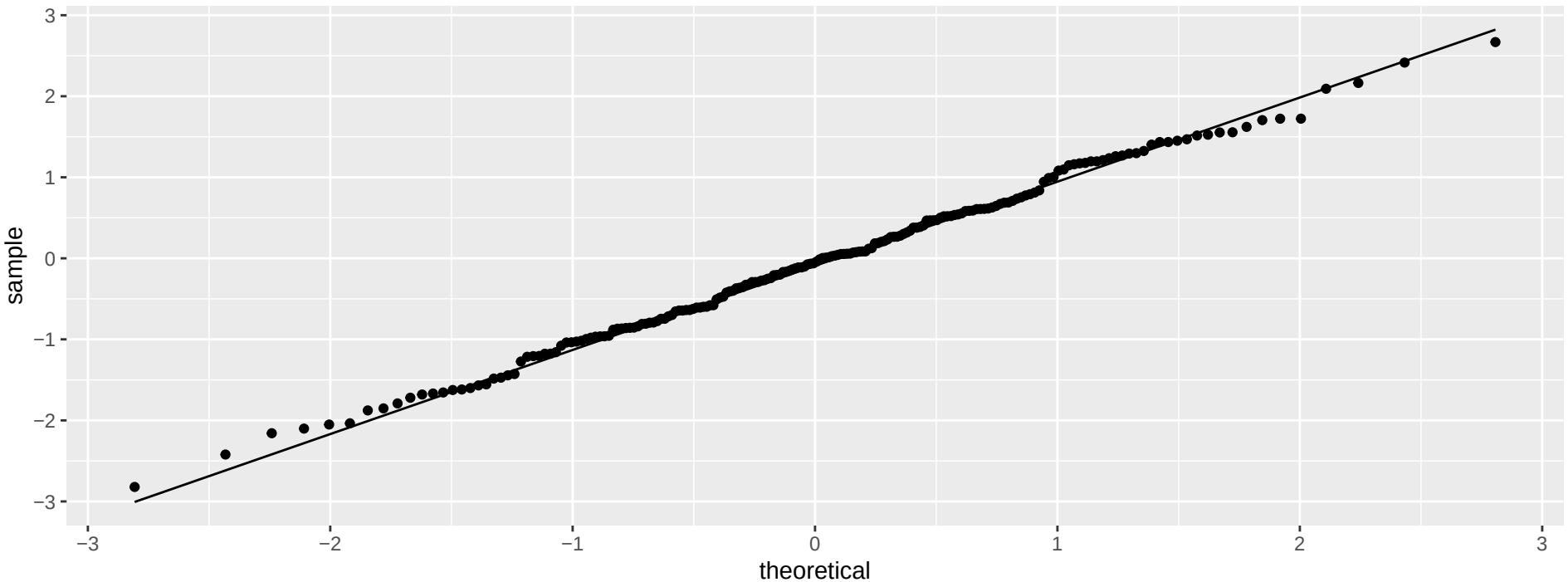
Normal data, large sample

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



The normal quantile plot

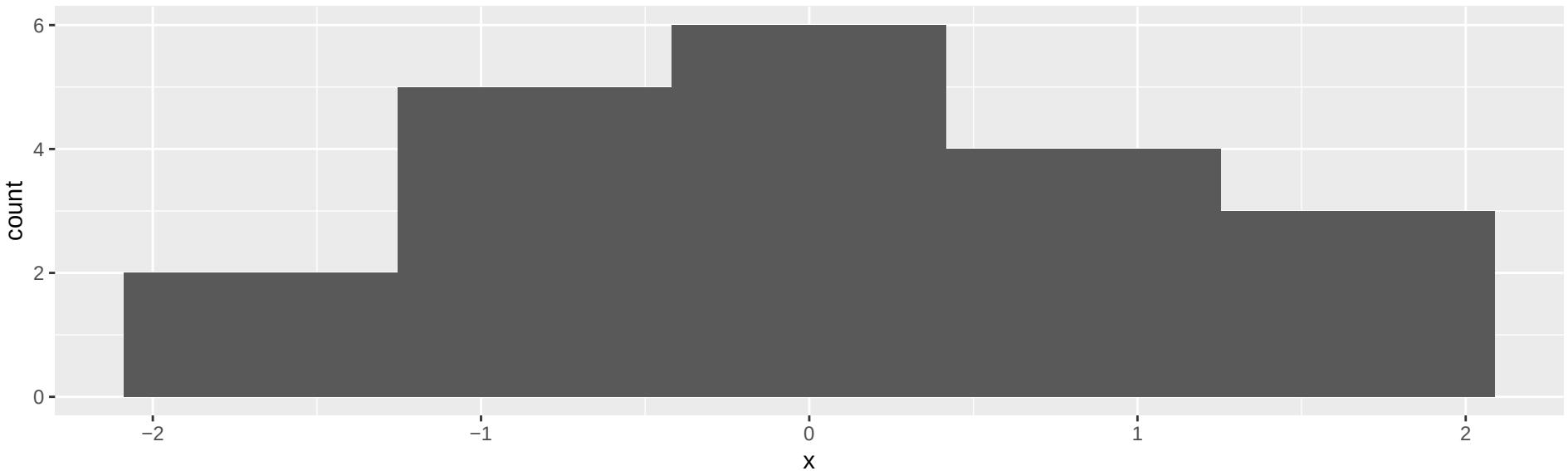
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Normal data, small sample

- Not so convincingly normal, but not obviously skewed:

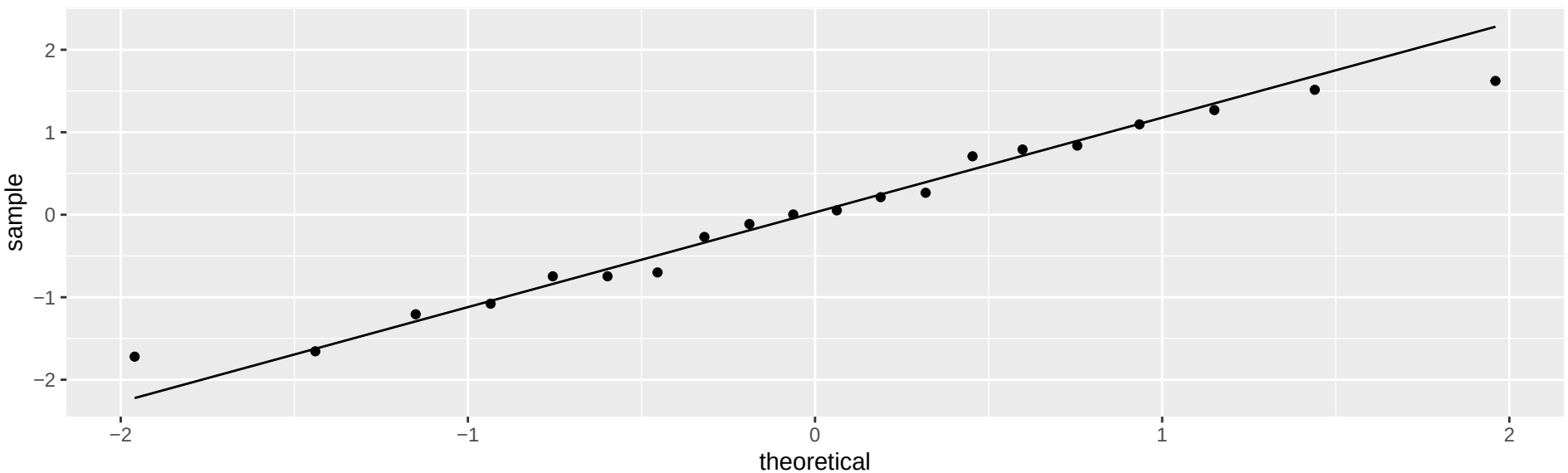
```
d=tibble(x=rnorm(20))
ggplot(d,aes(x=x))+geom_histogram(bins=5)
```



The normal quantile plot

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

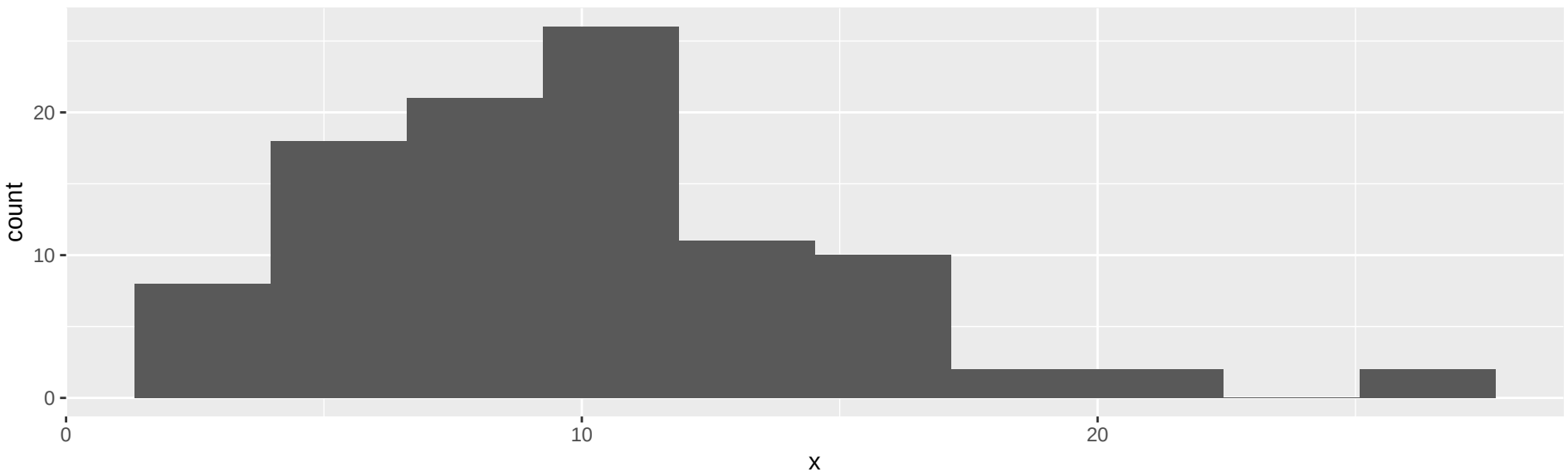
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Chi-squared data, $df = 10$

Somewhat skewed to right:

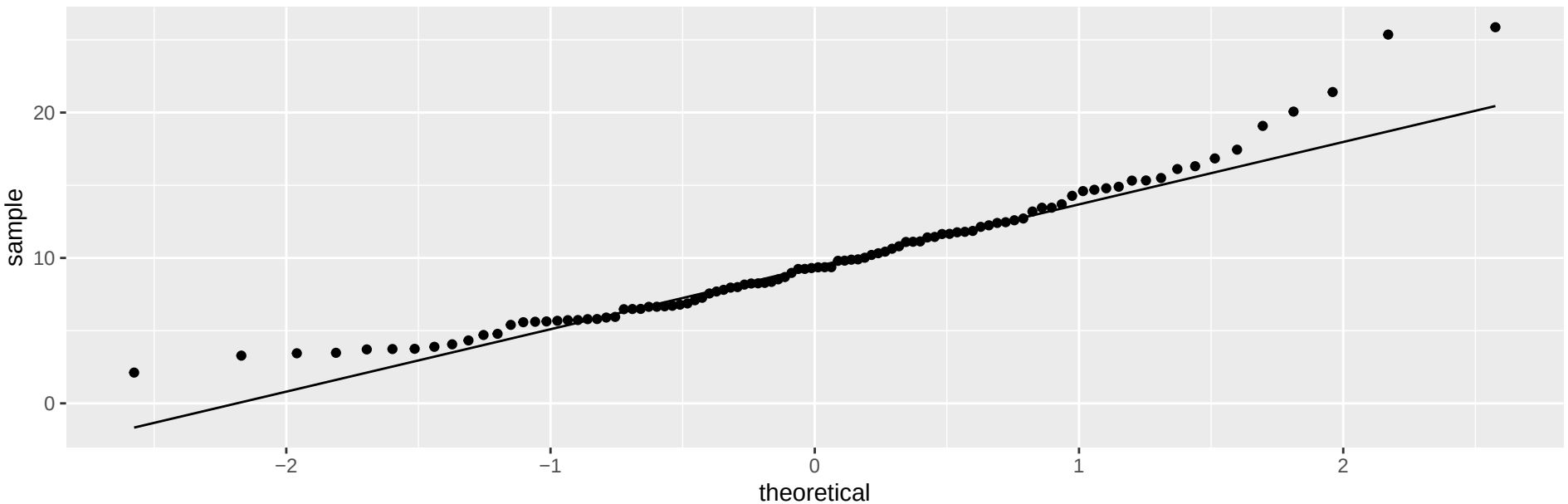
```
d=tibble(x=rchisq(100,10))
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

Somewhat opening-up curve:

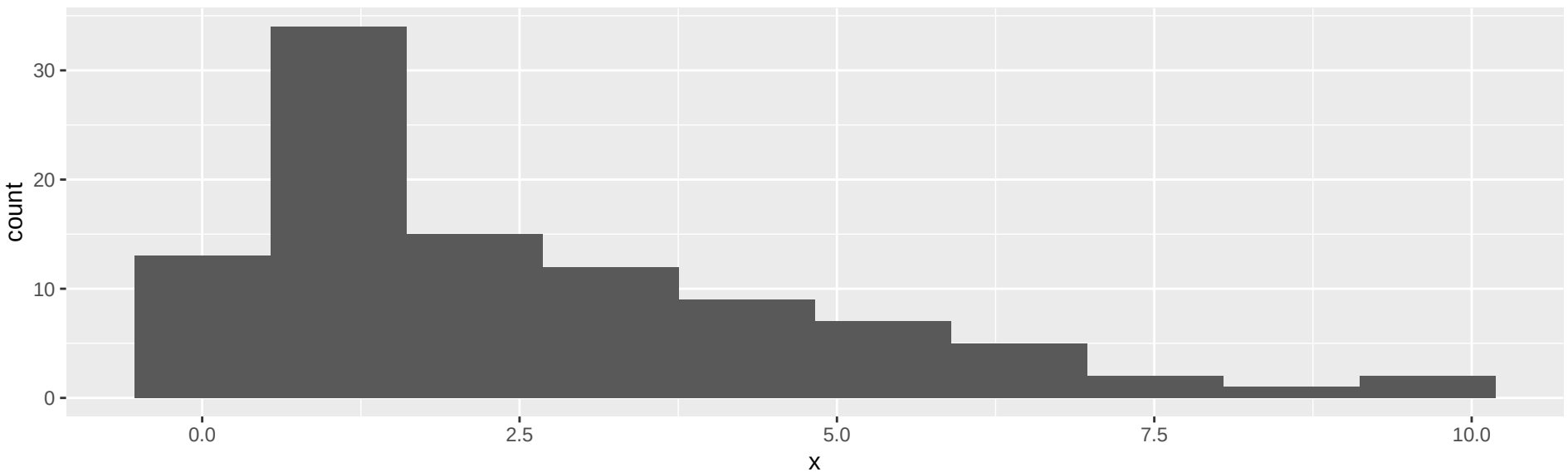
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



Chi-squared data, df = 3

Definitely skewed to right:

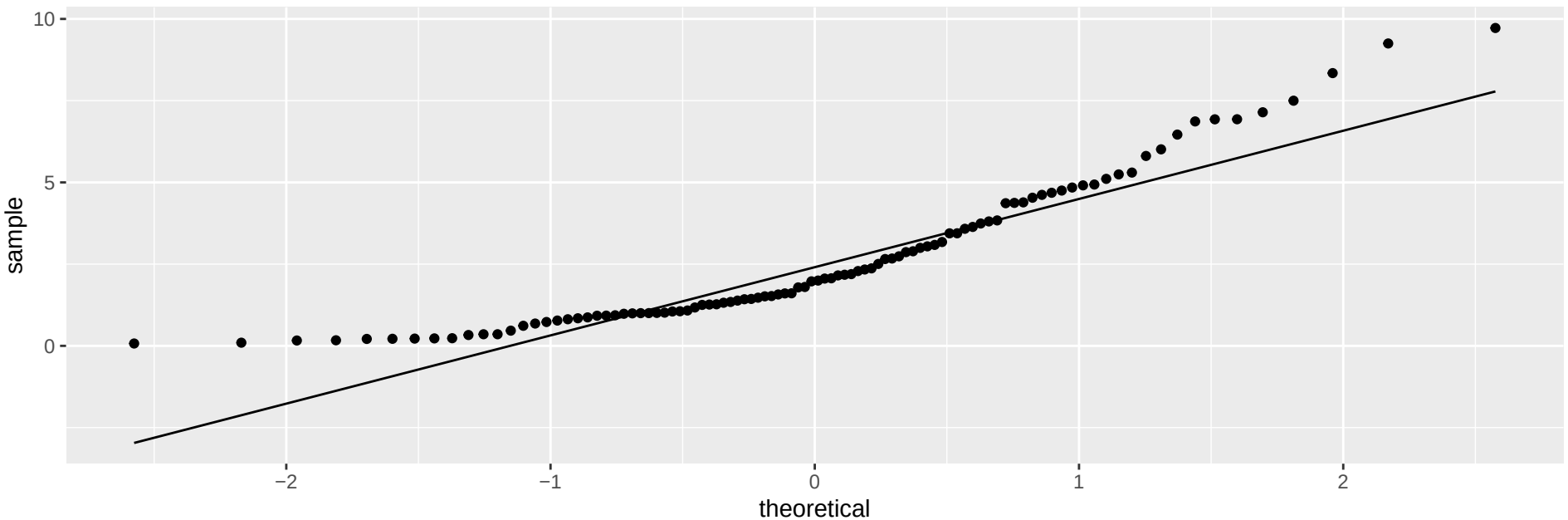
```
d=tibble(x=rchisq(100,3))
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

Clear upward-opening curve:

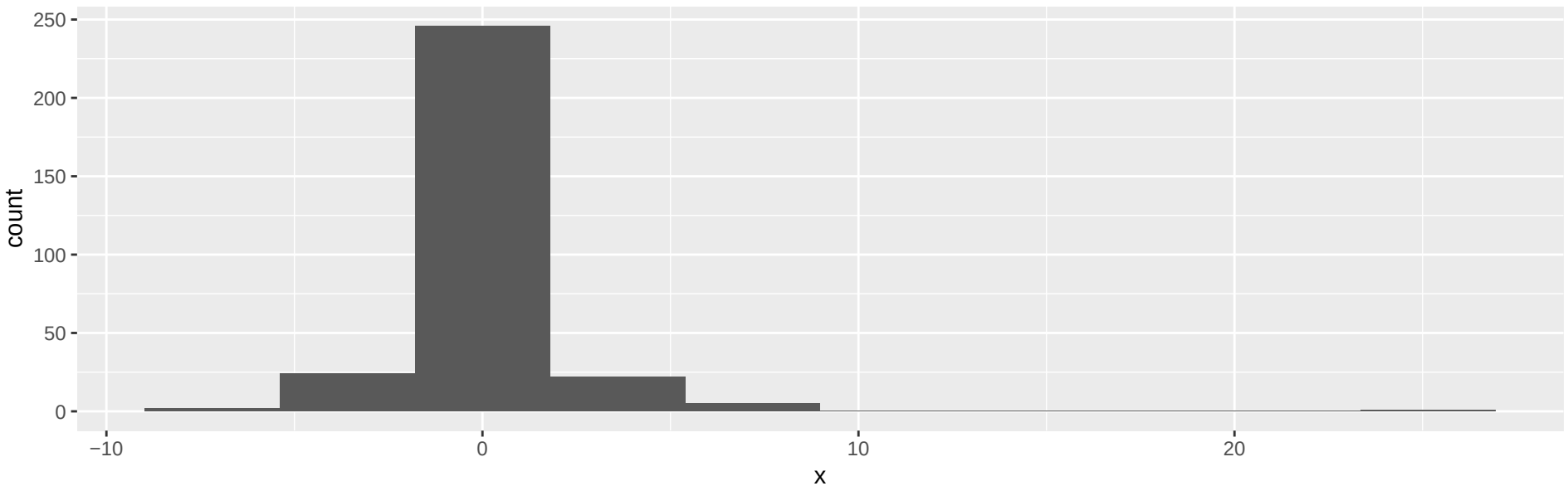
```
ggplot(d,aes(sample=x))+stat_qq()+stat_qq_line()
```



t-distributed data, df = 3

Long tails (or a very sharp peak):

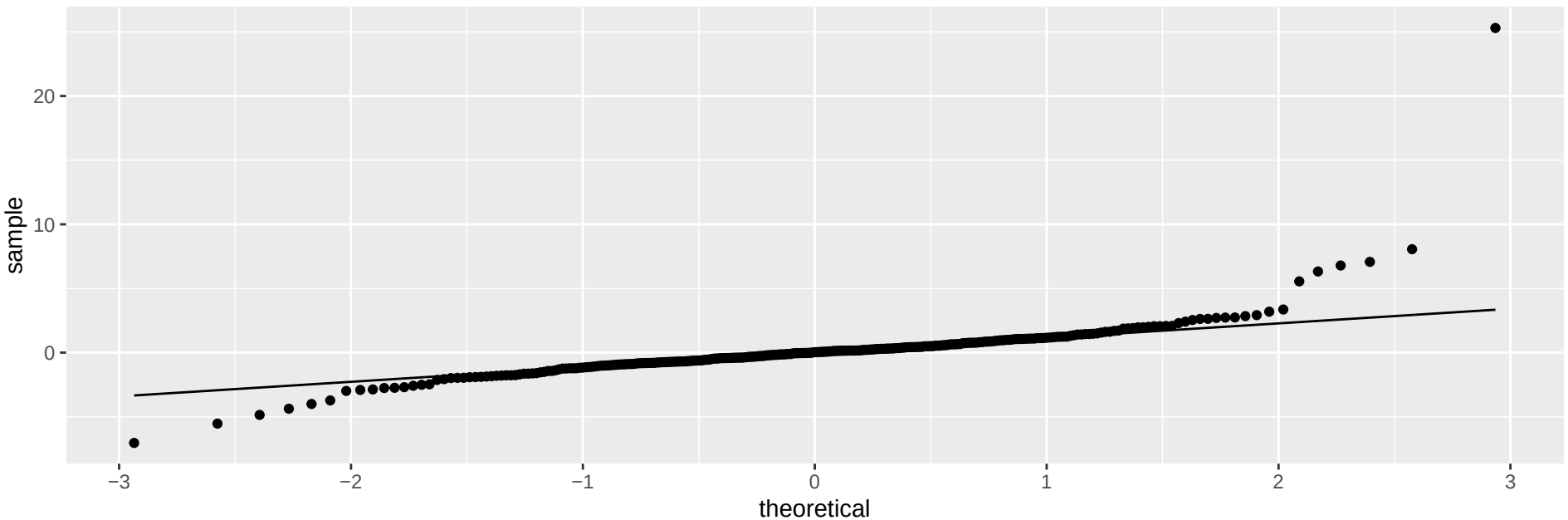
```
d=tibble(x=rt(300,3))
ggplot(d,aes(x=x))+geom_histogram(bins=10)
```



The normal quantile plot

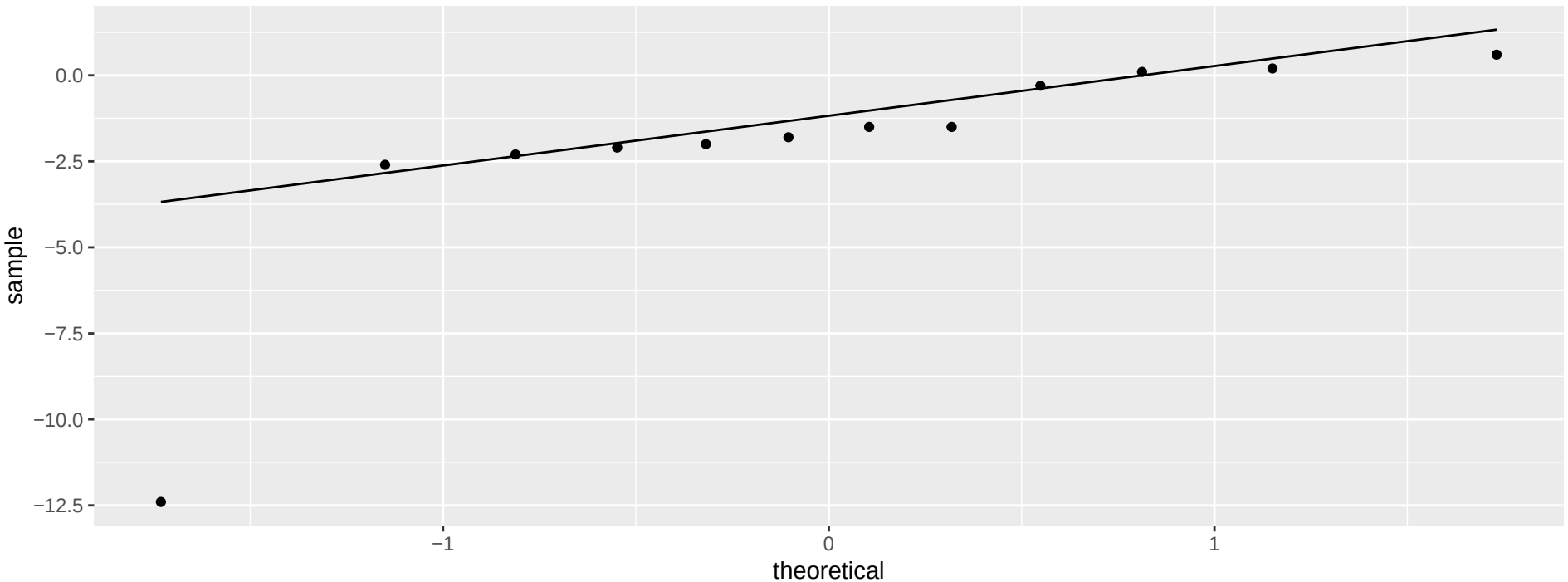
Low values too low and high values too high for normal.

```
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 `smmr`.
- Null median difference is 0:

```
pain %>% mutate(mydiff=druga-drugb) %>%
sign_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)
```

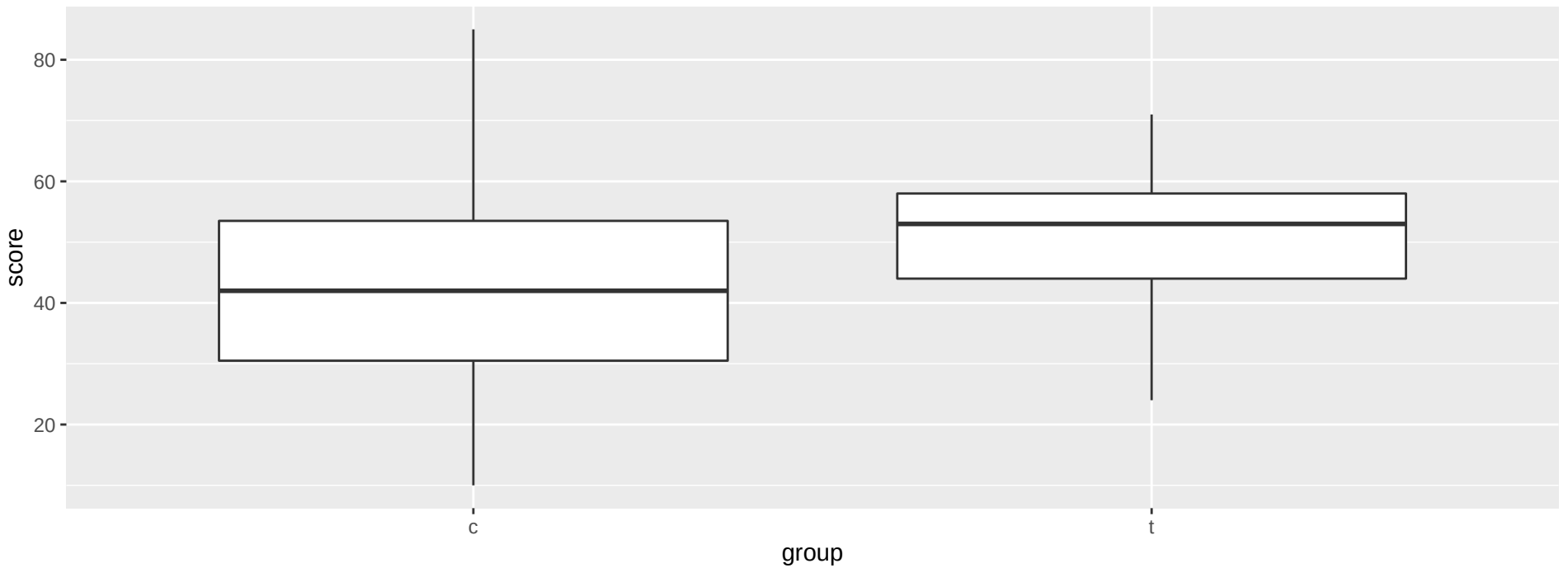
group	score
t	49
c	48
c	46
c	17
t	43
t	44
c	26
t	59
c	55
t	62
t	43
c	10

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

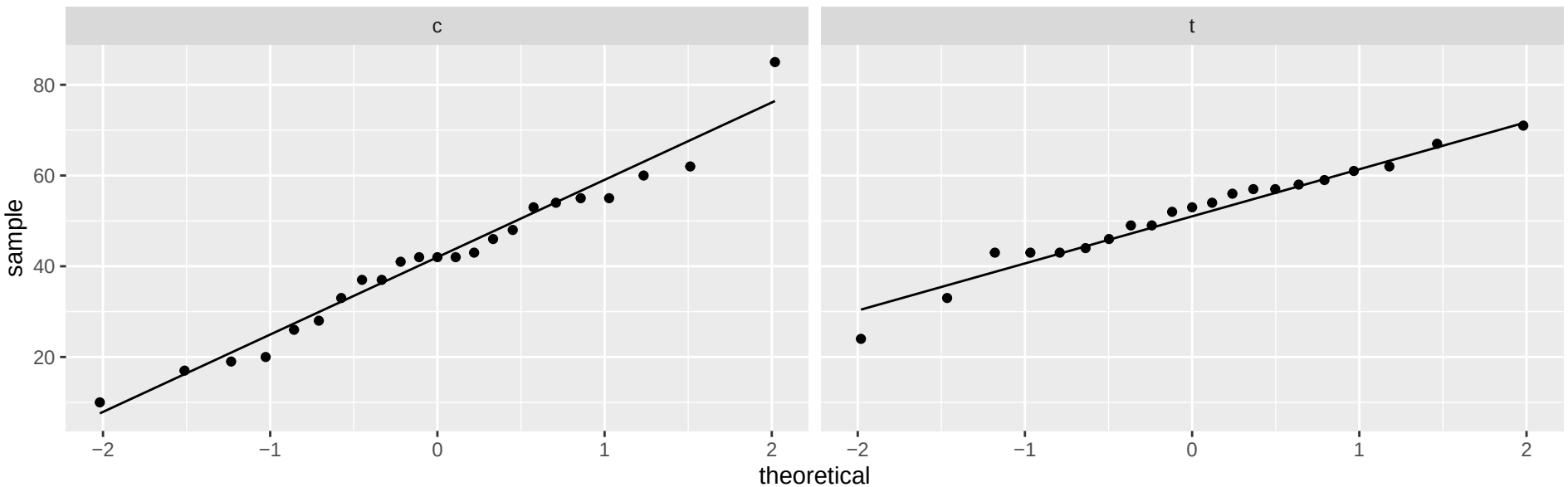
```
ggplot(kids,aes(x=group,y=score))+geom_boxplot()
```



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 th
## 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()
## )
```

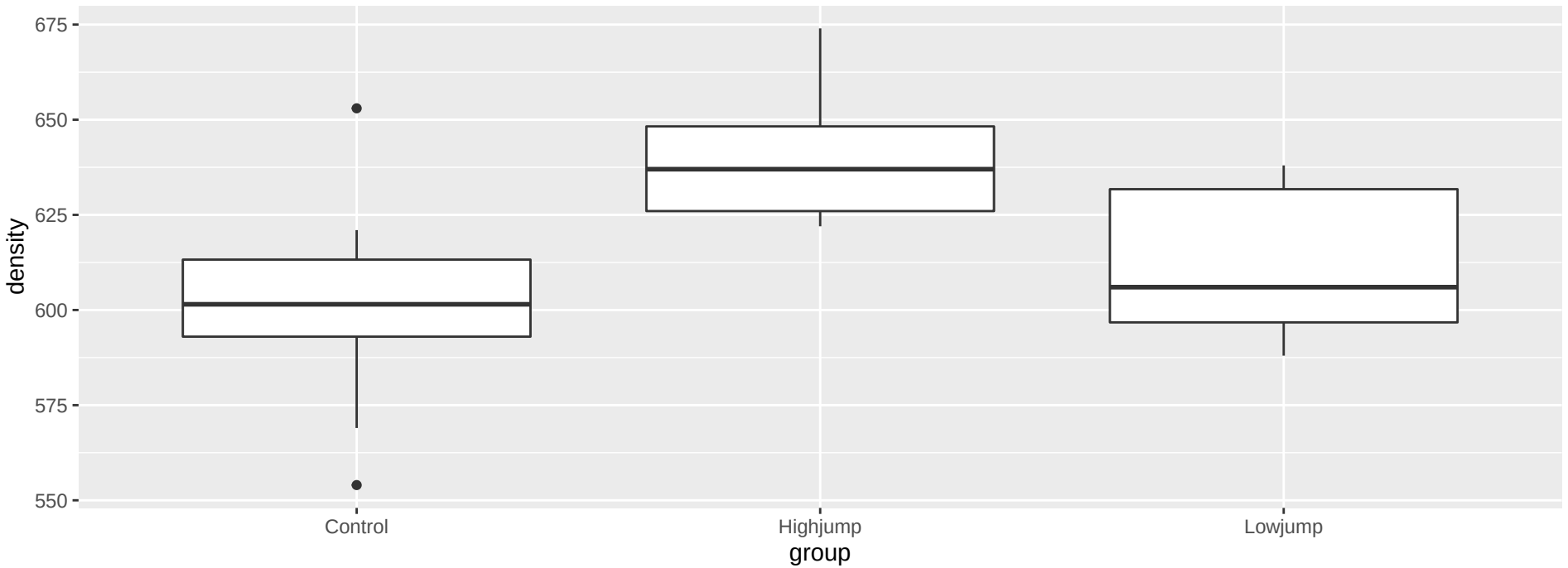
The data (some random rows)

```
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

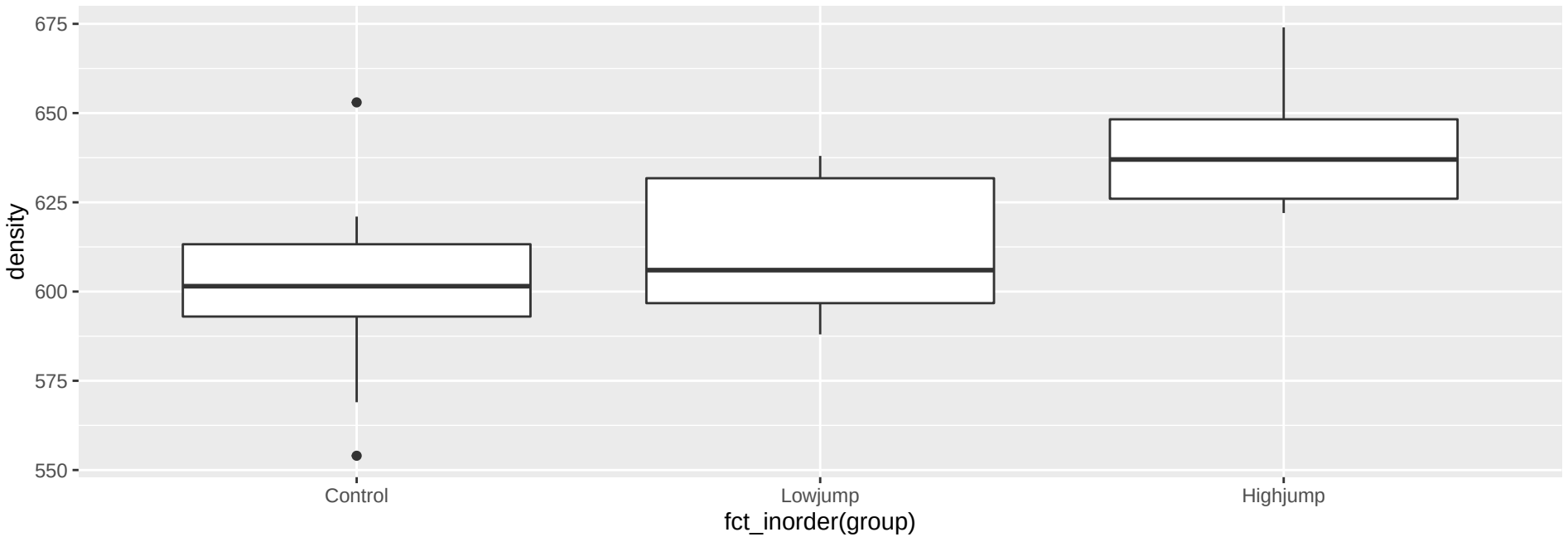
Boxplots

```
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 equ
## 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 equ
## 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 equ
## 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

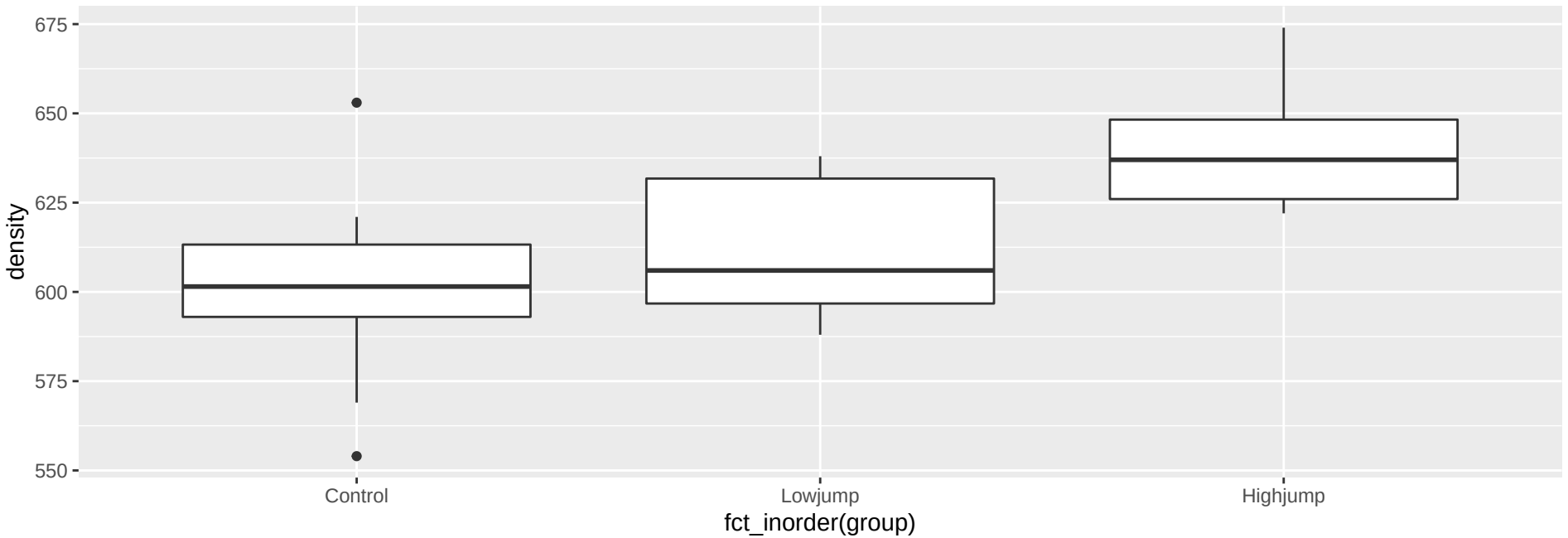
Look at P-values for the two tests:

```
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

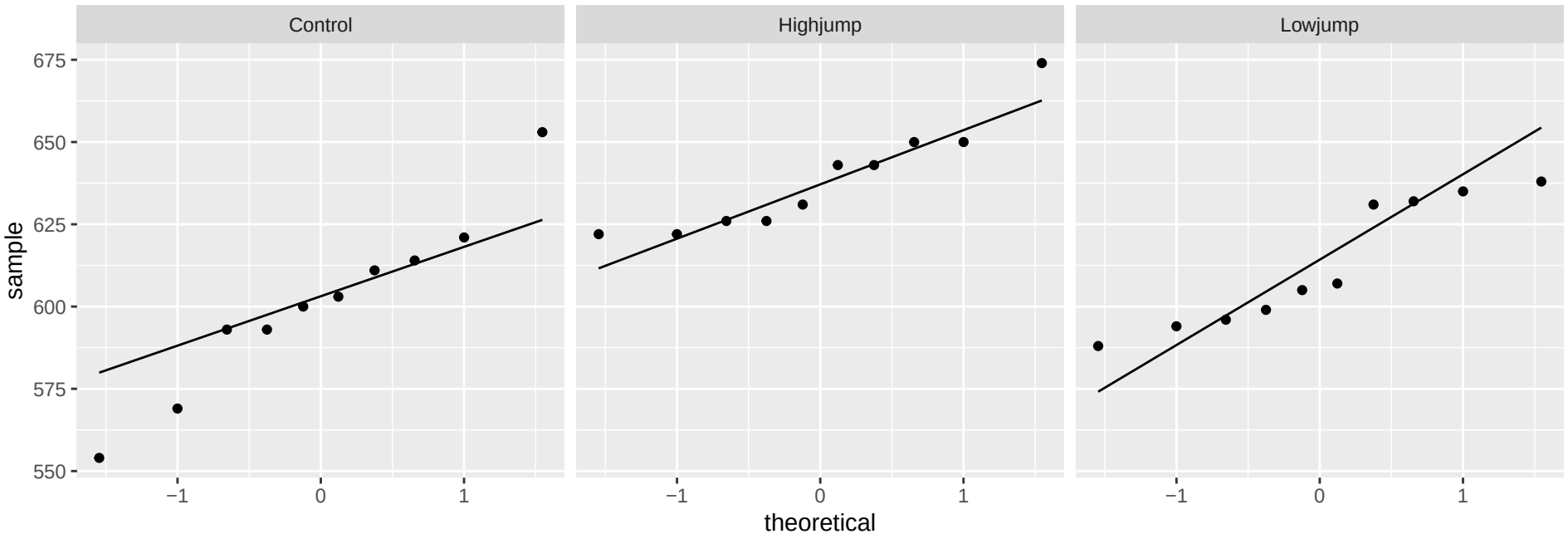
```
ggplot(rats,aes(y=density,x=fct_inorder(group)))+
geom_boxplot()
```



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
```