Regression revisited

Regression

- Use regression when one variable is an outcome (*response*, y).
- See if/how response depends on other variable(s), *explanatory*, $x_1, x_2, \ldots$.
- Can have *one* or *more than one* explanatory variable, but always one response.
- Assumes a *straight-line* relationship between response and explanatory.
- Ask:
  - *is there* a relationship between y and x's, and if so, which ones?
  - what does the relationship look like?

Packages

```
library(MASS) # for Box-Cox, later
library(tidyverse)
library(broom)
```

A regression with one x

13 children, measure average total sleep time (ATST, mins) and age (years) for each. See if ATST depends on age. Data in `sleep.txt`, ATST then age. Read in data:

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/sleep.txt"
sleep <- read_delim(my_url, " ")
```

```
##
## -- Column specification --------------------------
## cols(
##   atst = col_double(),
##   age = col_double()
## )
```

Check data

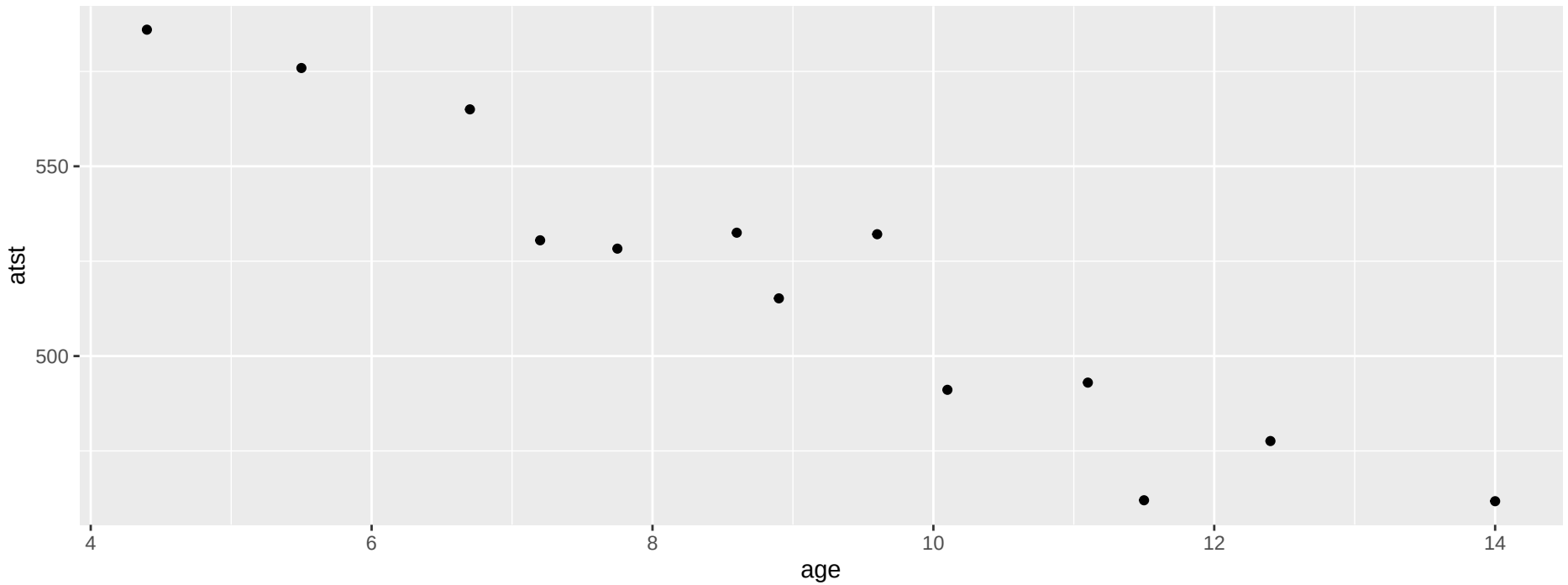
```
summary(sleep)
```

```
##       atst            age
##  Min.   :461.8   Min.   : 4.400
##  1st Qu.:491.1   1st Qu.: 7.200
##  Median :528.3   Median : 8.900
##  Mean   :519.3   Mean   : 9.058
##  3rd Qu.:532.5   3rd Qu.:11.100
##  Max.   :586.0   Max.   :14.000
```

Make scatter plot of ATST (response) vs. age (explanatory) using code overleaf:

The scatterplot

```
ggplot(sleep, aes(x = age, y = atst)) + geom_point()
```



Correlation

- Measures how well a straight line fits the data:

```
with(sleep, cor(atst, age))
```

```
## [1] -0.9515469
```

- 1 is perfect upward trend, -1 is perfect downward trend, 0 is no trend.
- This one close to perfect downward trend.
- Can do correlations of all pairs of variables:

```
cor(sleep)
```

```
##              atst        age
## atst  1.0000000 -0.9515469
## age  -0.9515469  1.0000000
```

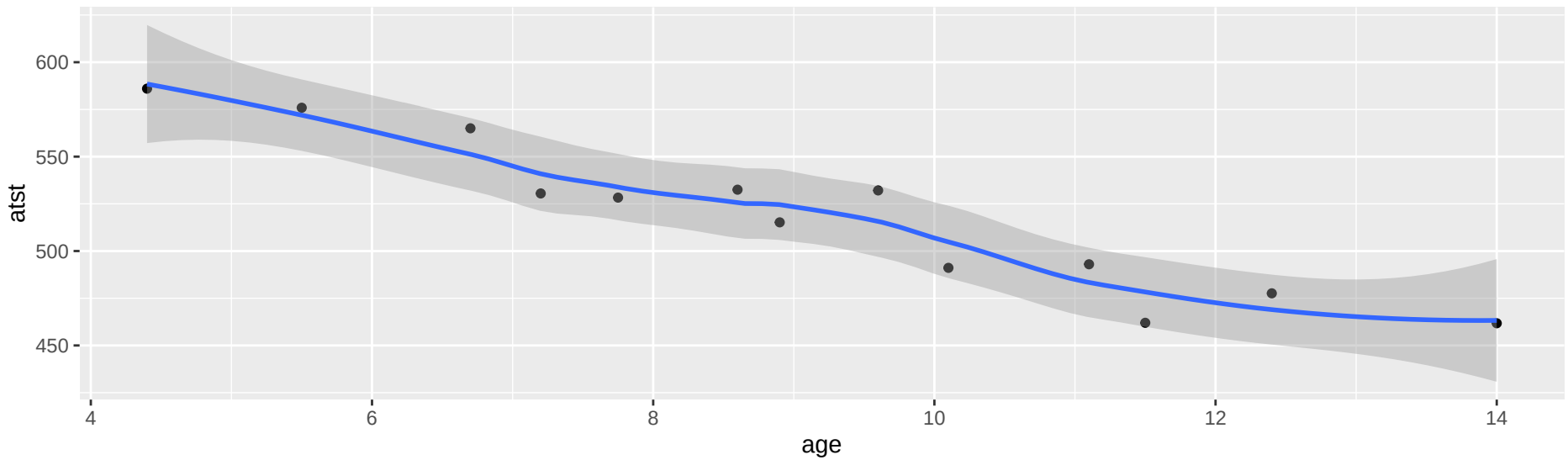
Lowess curve

- Sometimes nice to guide the eye: is the trend straight, or not?
- Idea: *lowess curve*. "Locally weighted least squares", not affected by outliers, not constrained to be linear.
- Lowess is a *guide*: even if straight line appropriate, may wiggle/bend a little. Looking for *serious* problems with linearity.
- Add lowess curve to plot using `geom_smooth`:

Plot with lowess curve

```
ggplot(sleep, aes(x = age, y = atst)) + geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The regression

Scatterplot shows no obvious curve, and a pretty clear downward trend. So we can run the regression:

```
sleep.1 <- lm(atst ~ age, data = sleep)
```

The output

```
summary(sleep.1)
```

```
##
## Call:
## lm(formula = atst ~ age, data = sleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.011  -9.365   2.372   6.770  20.411
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  646.483     12.918   50.05 2.49e-14 ***
## age          -14.041      1.368  -10.26 5.70e-07 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.15 on 11 degrees of freedom
## Multiple R-squared:  0.9054, Adjusted R-squared:  0.8968
## F-statistic: 105.3 on 1 and 11 DF,  p-value: 5.7e-07
```

Conclusions

- The relationship appears to be a straight line, with a downward trend.
- F-tests for model as a whole and t-test for slope (same) both confirm this (P-value $5.7 \times 10^{-7} = 0.00000057$).
- Slope is -14, so a 1-year increase in age goes with a 14-minute decrease in ATST on average.
- R-squared is correlation squared (when one x anyway), between 0 and 1 (1 good, 0 bad).
- Here R-squared is 0.9054, pleasantly high.

Doing things with the regression output

- Output from regression (and eg. t-test) is all right to look at, but hard to extract and re-use information from.
- Package `broom` extracts info from model output in way that can be used in pipe (later):

```
tidy(sleep.1)
```

term	estimate	std.error	statistic	p.value
(Intercept)	646.48334	12.917726	50.04622	0e+00
age	-14.04105	1.368116	-10.26305	6e-07

also one-line summary of model:

```
glance(sleep.1)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.905401	0.8969845	13.15238	105.3306	2e-07	1	-50.85618	107.7124	109.4072	1902.835	11

Broom part 2

```
sleep.1 %>% augment(sleep) %>% slice(1:8)
```

atst	age	.fitted	.resid	.std.resid	.hat	.sigma
586.00	4.4	584.7027	1.297271	0.1188843	0.3116588	13.78546
461.75	14.0	449.9087	11.841335	1.1092444	0.3412231	12.99996
491.10	10.1	504.6688	-13.568753	-1.0806868	0.0886783	13.04151
565.00	6.7	552.4083	12.591682	1.0306037	0.1370698	13.11145
462.00	11.5	485.0113	-23.011286	-1.8882414	0.1414645	11.34048
532.10	9.6	511.6893	20.410722	1.6180245	0.0801053	12.04143
477.60	12.4	472.3743	5.225658	0.4436029	0.1977964	13.67039
515.20	8.9	521.5180	-6.318011	-0.5000582	0.0771921	13.63664

Useful for plotting residuals against an x-variable.

CI for mean response and prediction intervals

Once useful regression exists, use it for prediction:

- To get a single number for prediction at a given x, substitute into regression equation, eg. age 10: predicted ATST is $646.48 - 14.04(10) = 506$ minutes.
- To express uncertainty of this prediction:
- *CI for mean response* expresses uncertainty about mean ATST for all children aged 10, based on data.
- *Prediction interval* expresses uncertainty about predicted ATST for a new child aged 10 whose ATST not known. More uncertain.
- Also do above for a child aged 5.

Intervals

- Make new data frame with these values for age

```
my.age <- c(10, 5)
ages.new <- tibble(age = my.age)
ages.new
```

age
10
5

- Feed into `predict`:

```
pc <- predict(sleep.1, ages.new, interval = "c")
pp <- predict(sleep.1, ages.new, interval = "p")
```

The intervals

Confidence intervals for mean response:

```
cbind(ages.new, pc)
```

age	fit	lwr	upr
10	506.0729	497.5574	514.5883
5	576.2781	561.6578	590.8984

Prediction intervals for new response:

```
cbind(ages.new, pp)
```

age	fit	lwr	upr
10	506.0729	475.8982	536.2475
5	576.2781	543.8474	608.7088

Comments

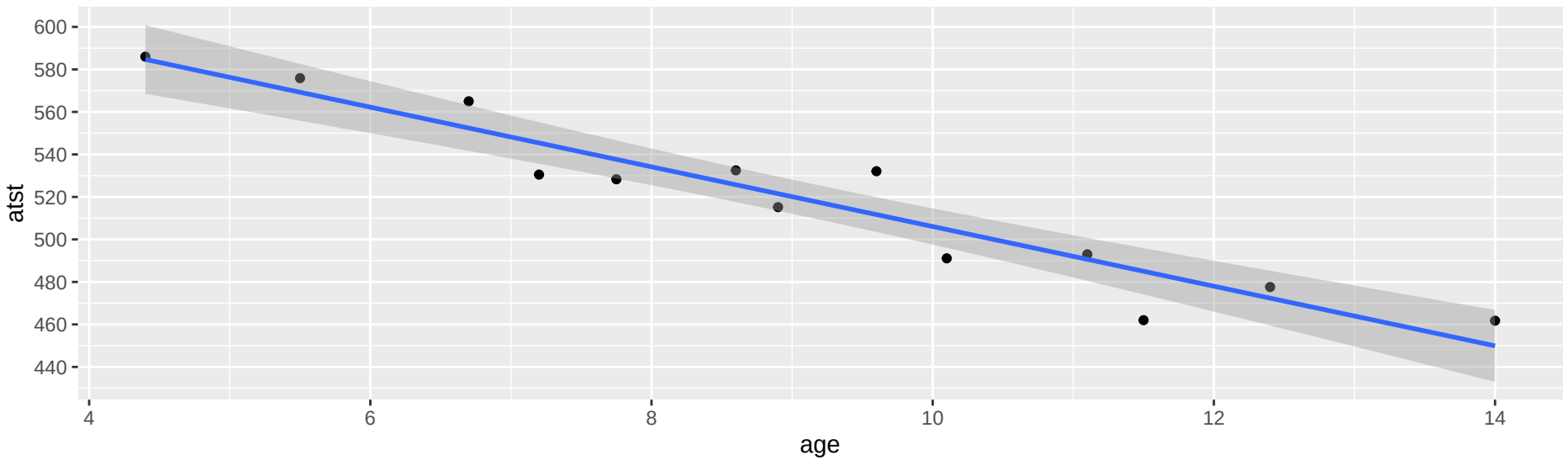
- Age 10 closer to centre of data, so intervals are both narrower than those for age 5.
- Prediction intervals bigger than CI for mean (additional uncertainty).
- Technical note: output from `predict` is R `matrix`, not data frame, so Tidyverse `bind_cols` does not work. Use base R `cbind`.

That grey envelope

Marks confidence interval for mean for all x:

```
ggplot(sleep, aes(x = age, y = atst)) + geom_point() +
  geom_smooth(method = "lm") +
  scale_y_continuous(breaks = seq(420, 600, 20))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Diagnostics

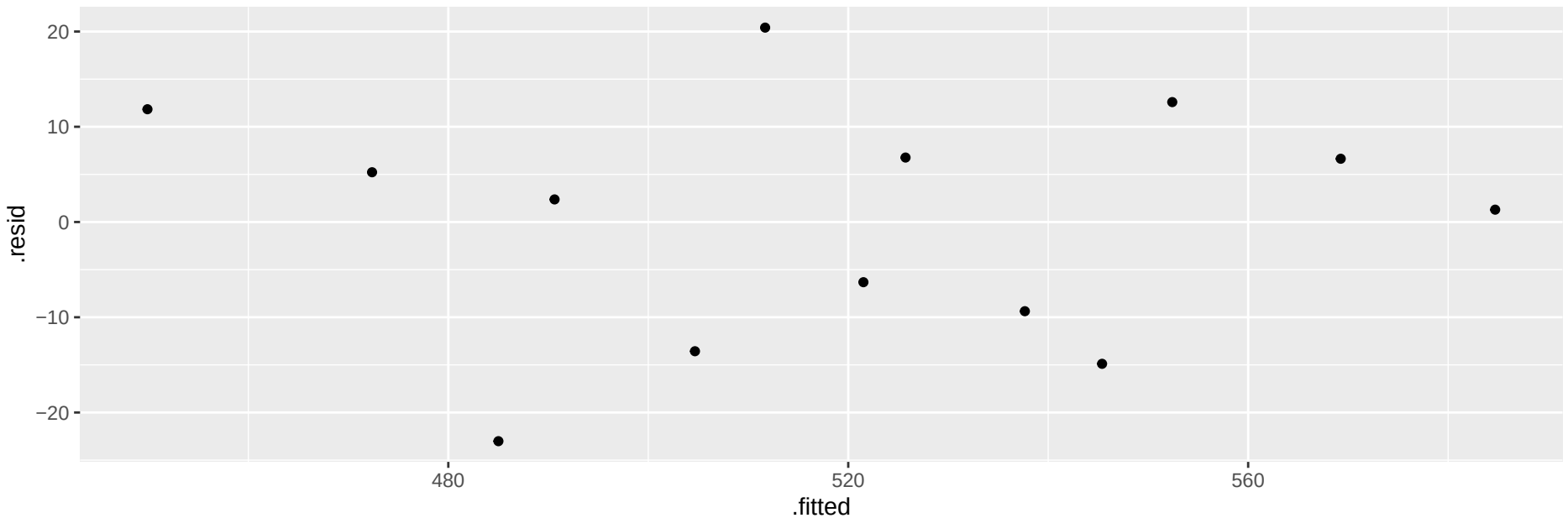
How to tell whether a straight-line regression is appropriate?

- Before: check scatterplot for straight trend.
- After: plot *residuals* (observed minus predicted response) against predicted values. Aim: a plot with no pattern.

Residual plot

Not much pattern here — regression appropriate.

```
ggplot(sleep.1, aes(x = .fitted, y = .resid)) + geom_point()
```



An inappropriate regression

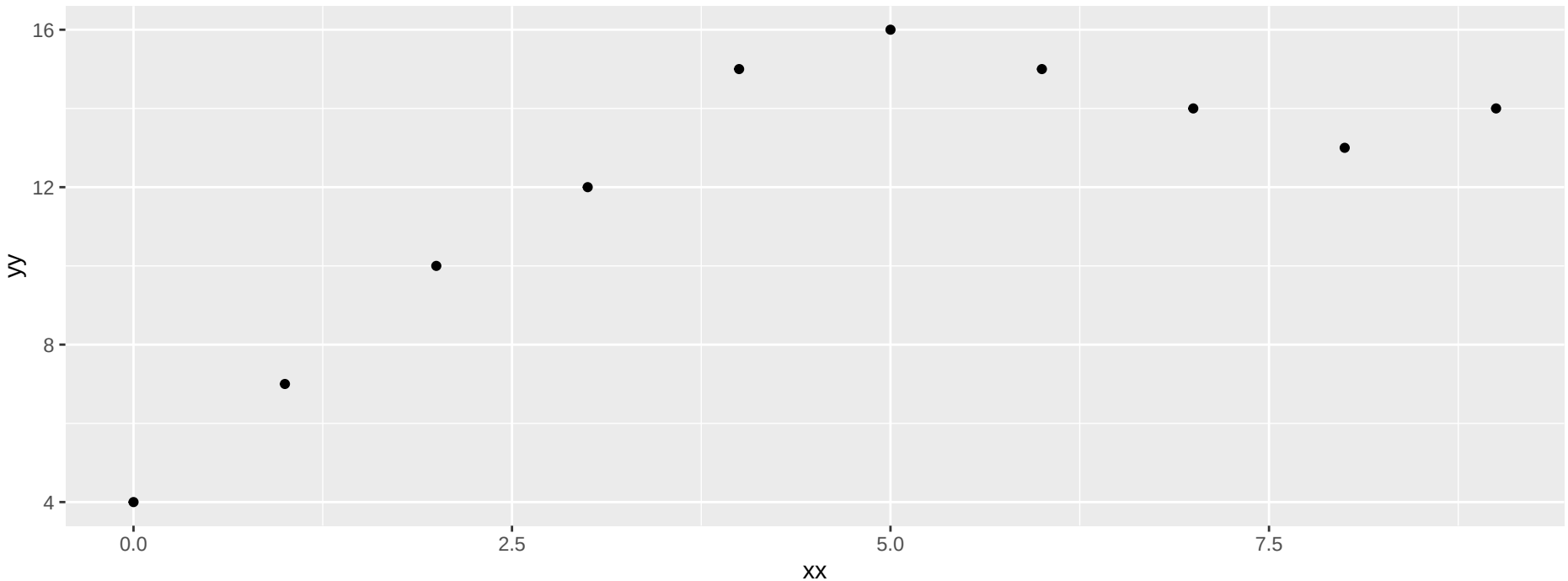
Different data:

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/curvy.txt"
curvy <- read_delim(my_url, " ")
```

```
##
## -- Column specification --------------------------------
## cols(
##   xx = col_double(),
##   yy = col_double()
## )
```

Scatterplot

```
ggplot(curvy, aes(x = xx, y = yy)) + geom_point()
```



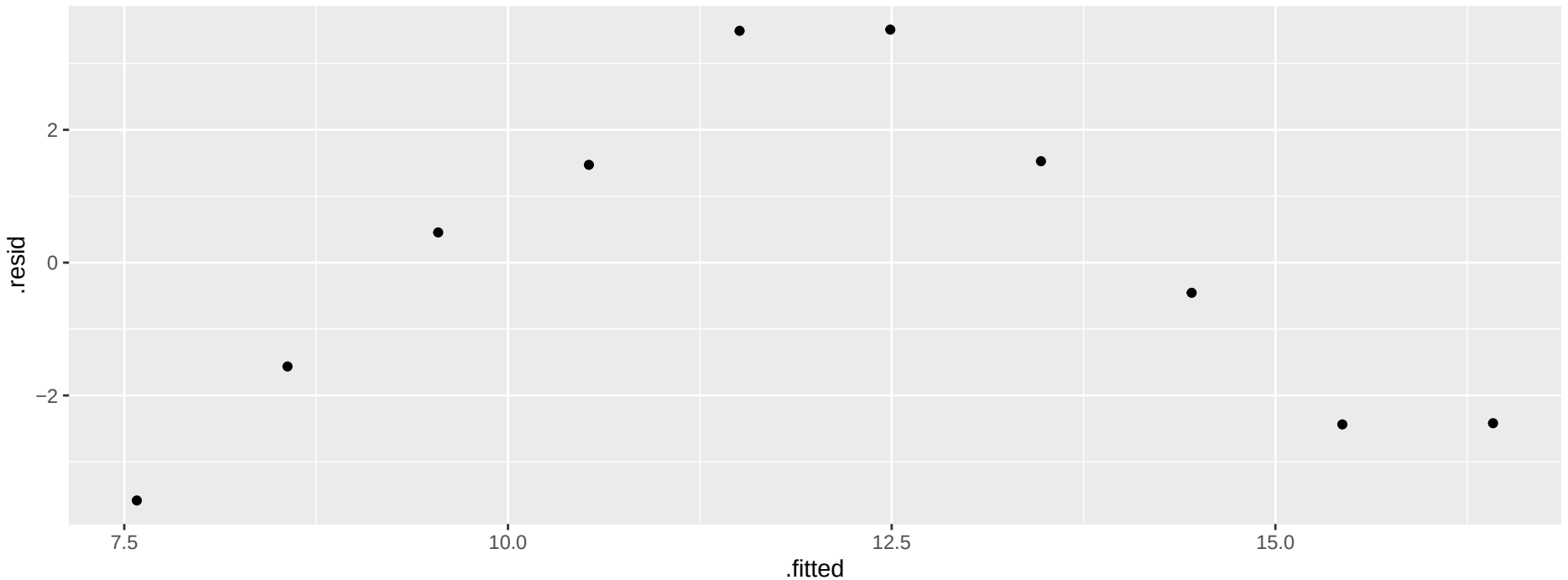
Regression line, anyway

```
curvy.1 <- lm(yy ~ xx, data = curvy)
summary(curvy.1)
```

```
##
## Call:
## lm(formula = yy ~ xx, data = curvy)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.582 -2.204  0.000  1.514  3.509
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.5818     1.5616   4.855  0.00126 **
## xx            0.9818     0.2925   3.356  0.00998 **
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.657 on 8 degrees of freedom
## Multiple R-squared:  0.5848, Adjusted R-squared:  0.5329
## F-statistic: 11.27 on 1 and 8 DF,  p-value: 0.009984
```

Residual plot

```
ggplot(curvy.1, aes(x = .fitted, y = .resid)) + geom_point()
```



No good: fixing it up

- Residual plot has *curve*: middle residuals positive, high and low ones negative. Bad.
- Fitting a curve would be better. Try this:

```
curvy.2 <- lm(yy ~ xx + I(xx^2), data = curvy)
```

- Adding `xx`-squared term, to allow for curve.
- Another way to do same thing: specify how model *changes*:

```
curvy.2a <- update(curvy.1, . ~ . + I(xx^2))
```

Regression 2

```
tidy(curvy.2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.9000000	0.7731168	5.044516	0.0014889
xx	3.7431818	0.4000606	9.356538	0.0000331
I(xx^2)	-0.3068182	0.0427927	-7.169866	0.0001822

```
glance(curvy.2) #
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.9502349	0.9360151	0.9833662	66.82902	2.875e-05	2	-12.23762	32.47524	33.68556	6.768182	7

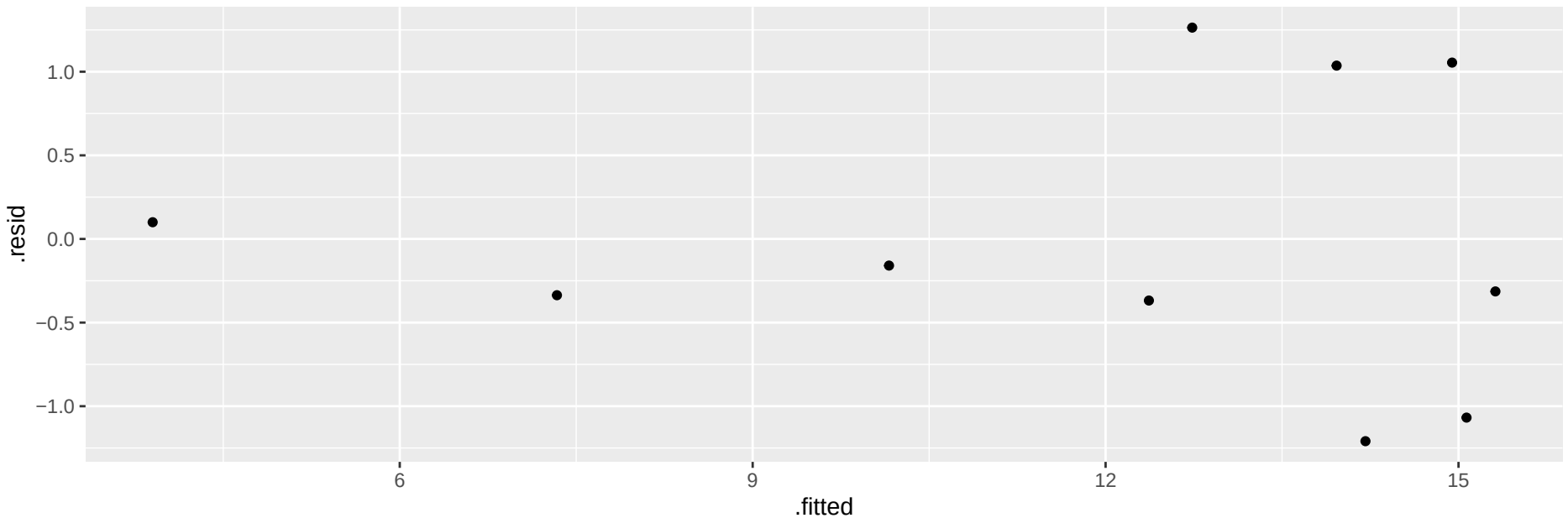
Comments

- xx-squared term definitely significant (P-value 0.000182), so need this curve to describe relationship.
- Adding squared term has made R-squared go up from 0.5848 to 0.9502: great improvement.
- This is a definite curve!

The residual plot now

No problems any more:

```
ggplot(curvy.2, aes(x = .fitted, y = .resid)) + geom_point()
```



Another way to handle curves

- Above, saw that changing x (adding x^2) was a way of handling curved relationships.
- Another way: change y (transformation).
- Can guess how to change y, or might be theory:
- example: relationship $y = ae^{bx}$ (exponential growth):
- take logs to get $\ln y = \ln a + bx$.
- Taking logs has made relationship linear ($\ln y$ as response).
- Or, *estimate* transformation, using Box-Cox method.

Box-Cox

- Install package `MASS` via `install.packages("MASS")` (only need to do *once*)
- Every R session you want to use something in `MASS`, type `library(MASS)`

Some made-up data

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/madeup.csv"
madeup <- read_csv(my_url)
madeup
```

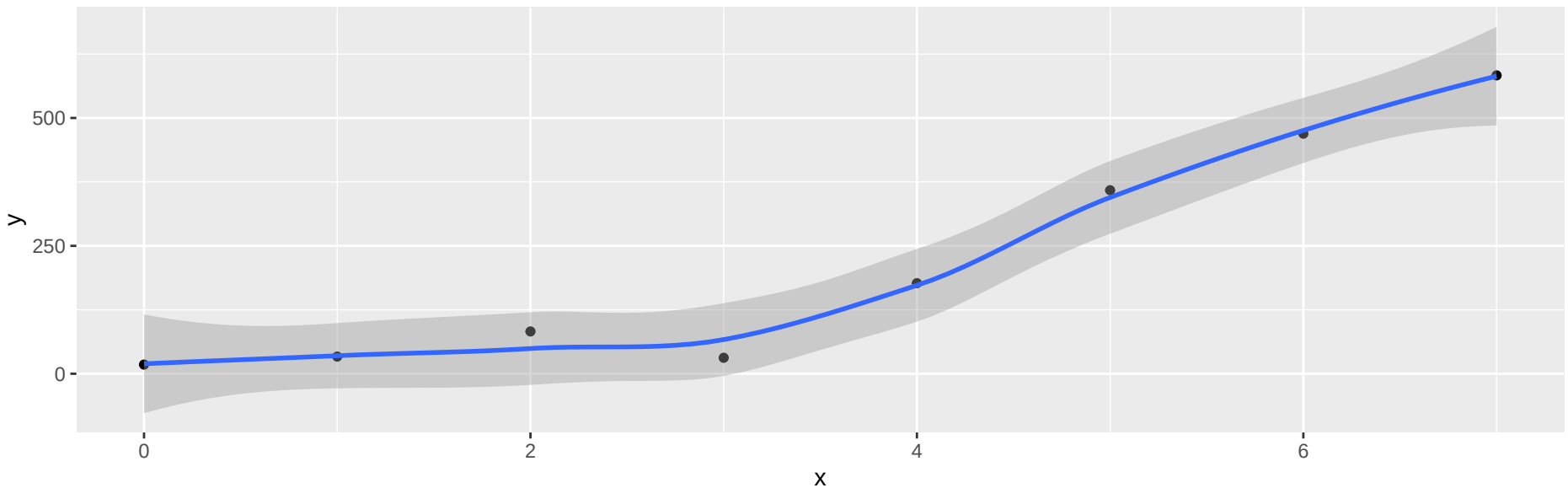
row	x	y
1	0	17.92576
2	1	33.58480
3	2	82.69371
4	3	31.19415
5	4	177.07919
6	5	358.70001
7	6	469.30232
8	7	583.24106

Seems to be faster-than-linear growth, maybe exponential growth.

Scatterplot: faster than linear growth

```
ggplot(madeup, aes(x = x, y = y)) + geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

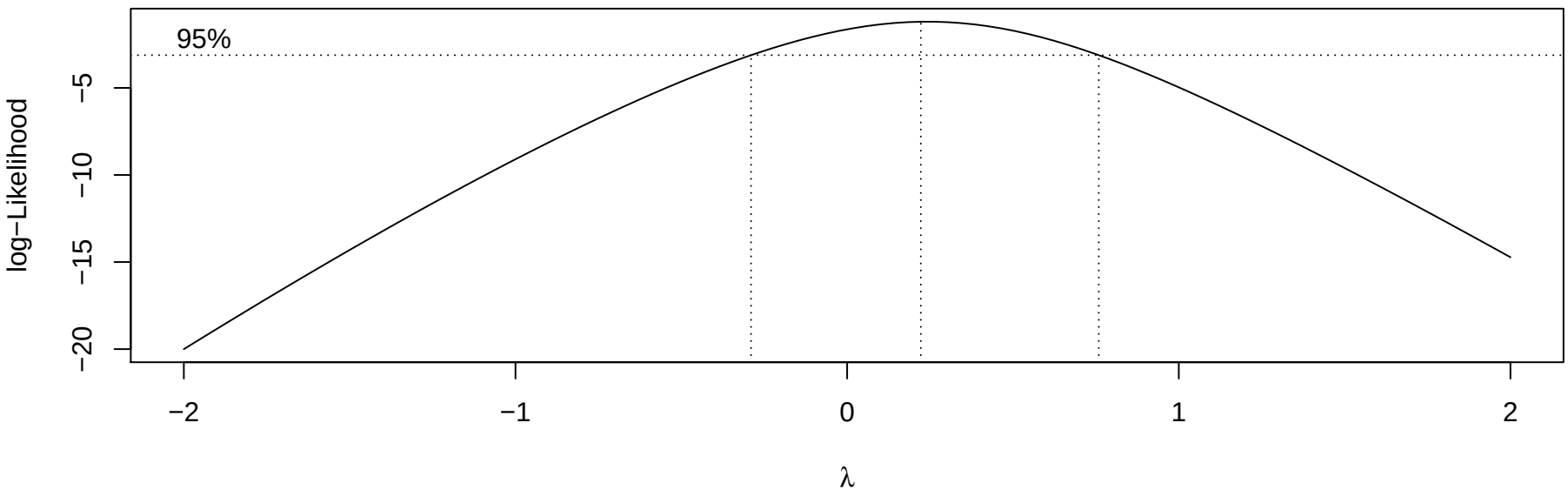


Running Box-Cox

- `library(MASS)` first.
- Feed `boxcox` a model formula with a squiggle in it, such as you would use for `lm`.
- Output: a graph (next page):

```
boxcox(y ~ x, data = madeup)
```

The Box-Cox output



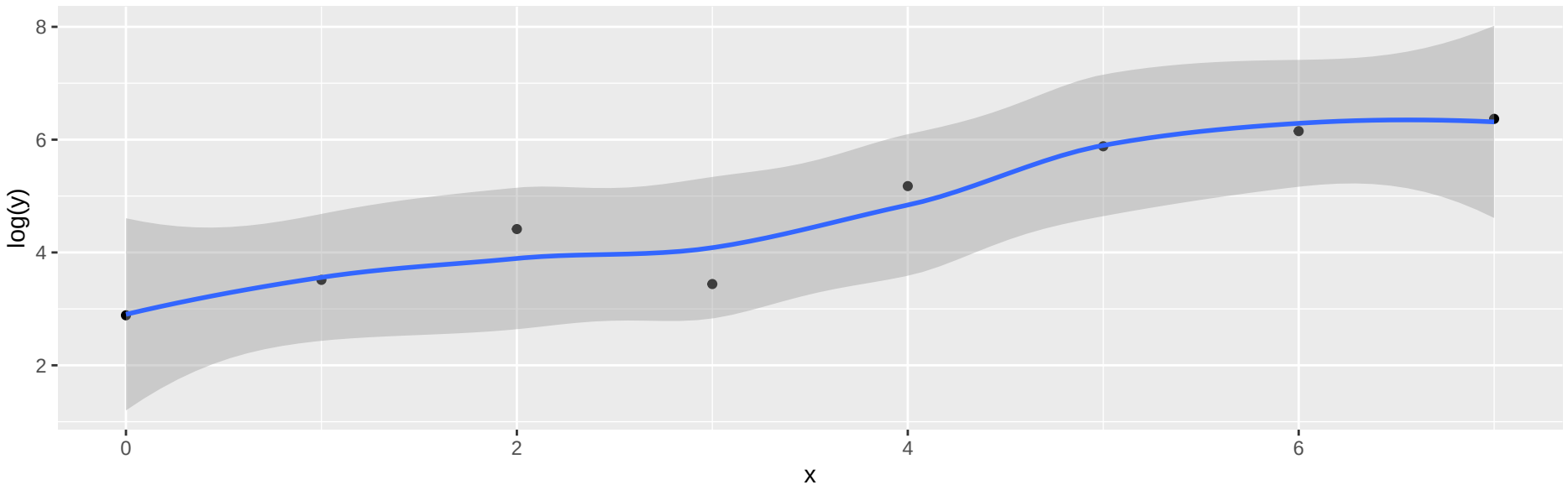
Comments

- λ (lambda) is the power by which you should transform y to get the relationship straight (straighter). Power 0 is "take logs"
- Middle dotted line marks best single value of λ (here about 0.1).
- Outer dotted lines mark 95% CI for λ, here -0.3 to 0.7, approx. (Rather uncertain about best transformation.)
- Any power transformation within the CI supported by data. In this case, log ($\lambda = 0$) and square root ($\lambda = 0.5$) good, but no transformation ($\lambda = 1$) not.
- Pick a "round-number" value of λ like $2, 1, 0.5, 0, -0.5, -1$. Here 0 and 0.5 good values to pick.

Did transformation straighten things?

- Plot transformed y against x. Here, log:

```
ggplot(madeup, aes(x = x, y = log(y))) + geom_point() +
  geom_smooth()
```



Looks much straighter.

Regression with transformed y

```
madeup.1 <- lm(log(y) ~ x, data = madeup)
glance(madeup.1)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.8830453	0.8635528	0.5009294	45.3019	0.0005235	1	-4.670459	15.34092	15.57924	1.505582	6

```
tidy(madeup.1)
```

term	estimate	std.error	statistic	p.value
(Intercept)	2.9084607	0.3233485	8.994816	0.0001056
x	0.5202476	0.0772951	6.730668	0.0005235

R-squared now decently high.

Multiple regression

- What if more than one x? Extra issues:
  - Now one intercept and a slope for each x: how to interpret?
  - Which x-variables actually help to predict y?
  - Different interpretations of "global" F-test and individual t-tests.
  - R-squared no longer correlation squared, but still interpreted as "higher better".
  - In `lm` line, add extra xs after `~`.
  - Interpretation not so easy (and other problems that can occur).

Multiple regression example

Study of women and visits to health professionals, and how the number of visits might be related to other variables:

timedrs: number of visits to health professionals (over course of study)

phyheal: number of physical health problems

menheal: number of mental health problems

stress: result of questionnaire about number and type of life changes

`timedrs` response, others explanatory.

The data

```
my_url <-
  "http://www.utsc.utoronto.ca/~butler/d29/regressx.txt"
visits <- read_delim(my_url, " ")
```

```
##
## -- Column specification --------------------------------
## cols(
##   subjno = col_double(),
##   timedrs = col_double(),
##   phyheal = col_double(),
##   menheal = col_double(),
##   stress = col_double()
## )
```

Check data

```
visits
```

subjno	timedrs	phyheal	menheal	stress
1	1	5	8	265
2	3	4	6	415
3	0	3	4	92
4	13	2	2	241
5	15	3	6	86
6	3	5	5	247
7	2	5	6	13
8	0	4	5	12
9	7	5	4	269
10	4	3	9	391
11	15	6	3	237
12	0	3	5	13
13	2	3	10	84
14	13	6	9	144
15	2	3	2	135

Fit multiple regression

```
visits.1 <- lm(timedrs ~ phyheal + menheal + stress,
  data = visits)
glance(visits.1)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.218770	0.213694	2.708454	43.03373	0	3	-1714.741	3439.483	3460.193	3451.134	461

The slopes

Model as a whole strongly significant even though R-sq not very big (lots of data). At least one of the x's predicts `timedrs`.

```
tidy(visits.1)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-3.7048477	1.1241951	-3.2955560	0.0010581
phyheal	1.7869481	0.2210735	8.0830488	0.0000000
menheal	-0.0096656	0.1290286	-0.0749106	0.9403184
stress	0.0136145	0.0036121	3.7690914	0.0001851

The physical health and stress variables initely help to predict the number of visits, but *with those in the model* we don't need `menheal`. However, look at prediction of `timedrs` from `menheal` by itself:

Just menheal

```
visits.2 <- lm(timedrs ~ menheal, data = visits)
glance(visits.2)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	log-Lik	AIC	BIC	deviance	df.residual
0.065316	0.0632974	10.59632	32.35466	0	1	-1756.44	3518.88	3531.306	51986.6	463

```
tidy(visits.2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.8158808	0.8702158	4.384982	1.44e-05
menheal	0.6672341	0.1173032	5.688116	0.00e+00

`menheal` by itself

- `menheal` by itself *does* significantly help to predict `timedrs`.
- But the R-sq is much less (6.5% vs. 22%).
- So other two variables do a better job of prediction.
- With those variables in the regression (`phyheal` and `stress`), don't need `menheal` *as well*.

Investigating via correlation

Leave out first column (subjno):

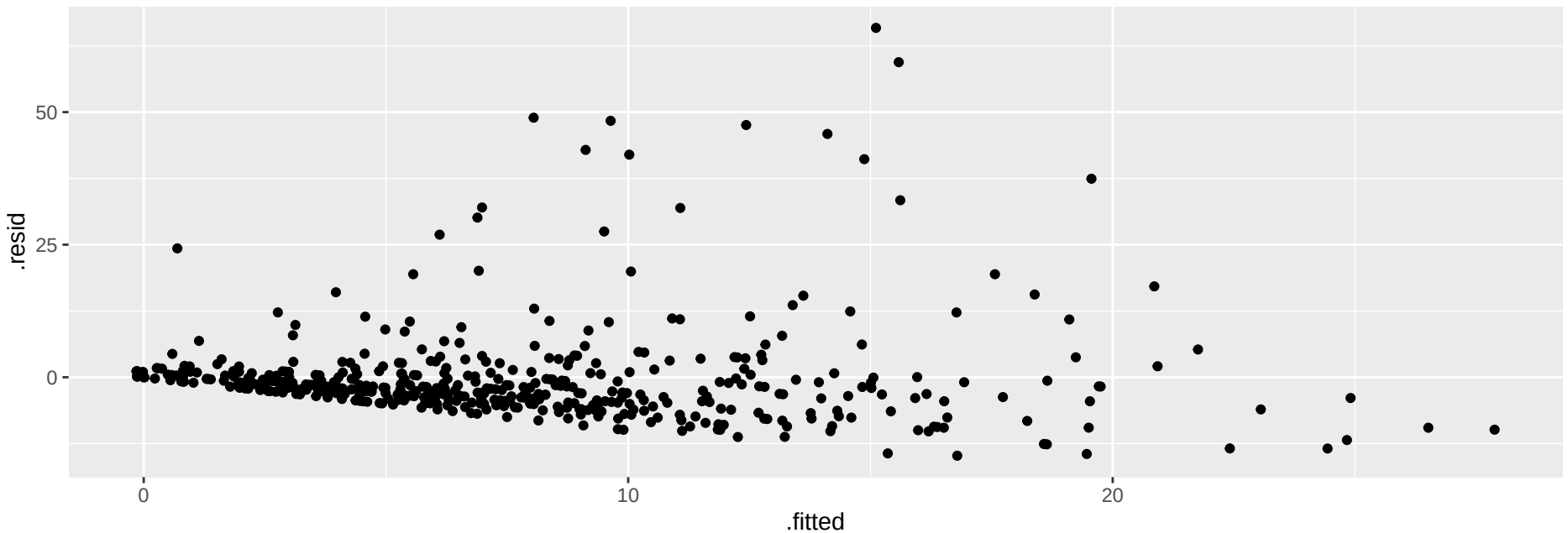
```
visits %>% select(-subjno) %>% cor()
```

```
##              timedrs   phyheal   menheal    stress
## timedrs 1.0000000 0.4395293 0.2555703 0.2865951
## phyheal 0.4395293 1.0000000 0.5049464 0.3055517
## menheal 0.2555703 0.5049464 1.0000000 0.3697911
## stress  0.2865951 0.3055517 0.3697911 1.0000000
```

- phyheal most strongly correlated with timedrs.
- Not much to choose between other two.
- But menheal has higher correlation with phyheal, so not as much to *add* to prediction as stress.
- Goes to show things more complicated in multiple regression.

Residual plot (from `timedrs` on all)

```
ggplot(visits.1, aes(x = .fitted, y = .resid)) + geom_point()
```

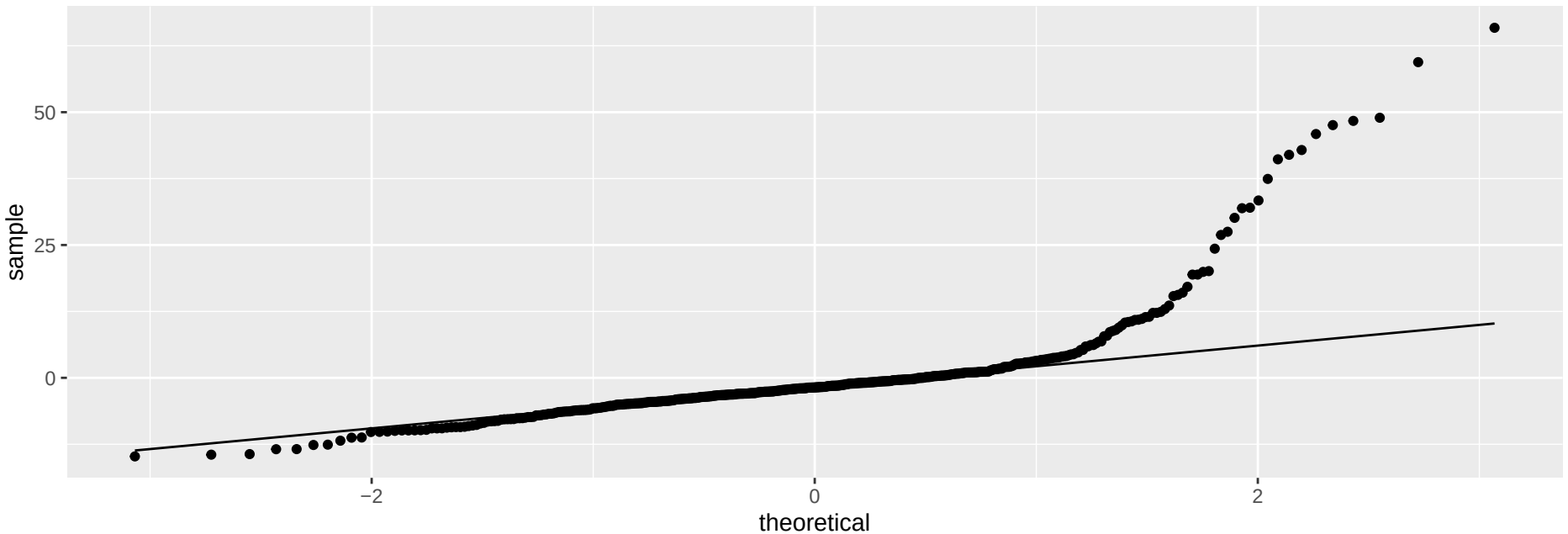


Comment

Apparently random. But...

Normal quantile plot of residuals

```
ggplot(visits.1, aes(sample = .resid)) + stat_qq() + stat_qq_l
```

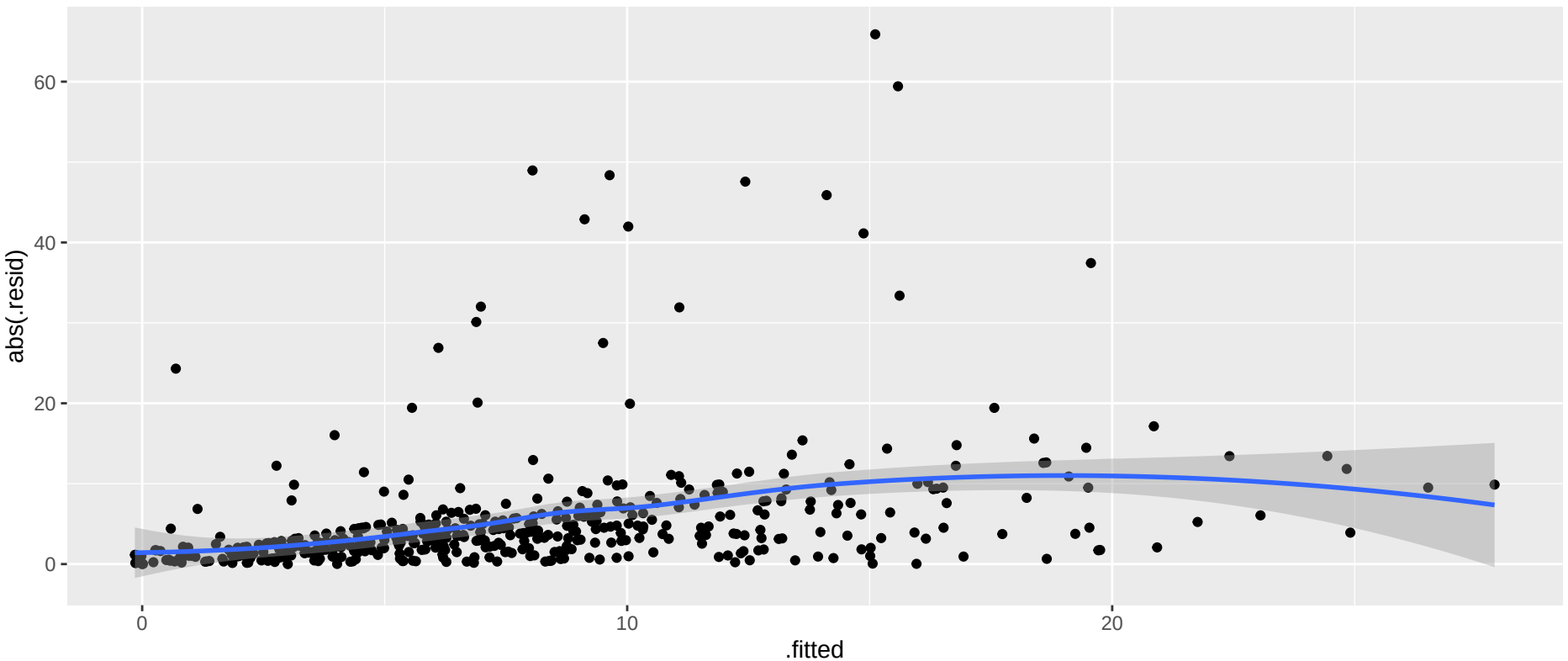


Absolute residuals

Is there trend in *size* of residuals (fan-out)? Plot *absolute value* of residual against fitted value (graph next page):

```
g <- ggplot(visits.1, aes(x = .fitted, y = abs(.resid))) +
  geom_point() + geom_smooth()
```

The plot



Comments

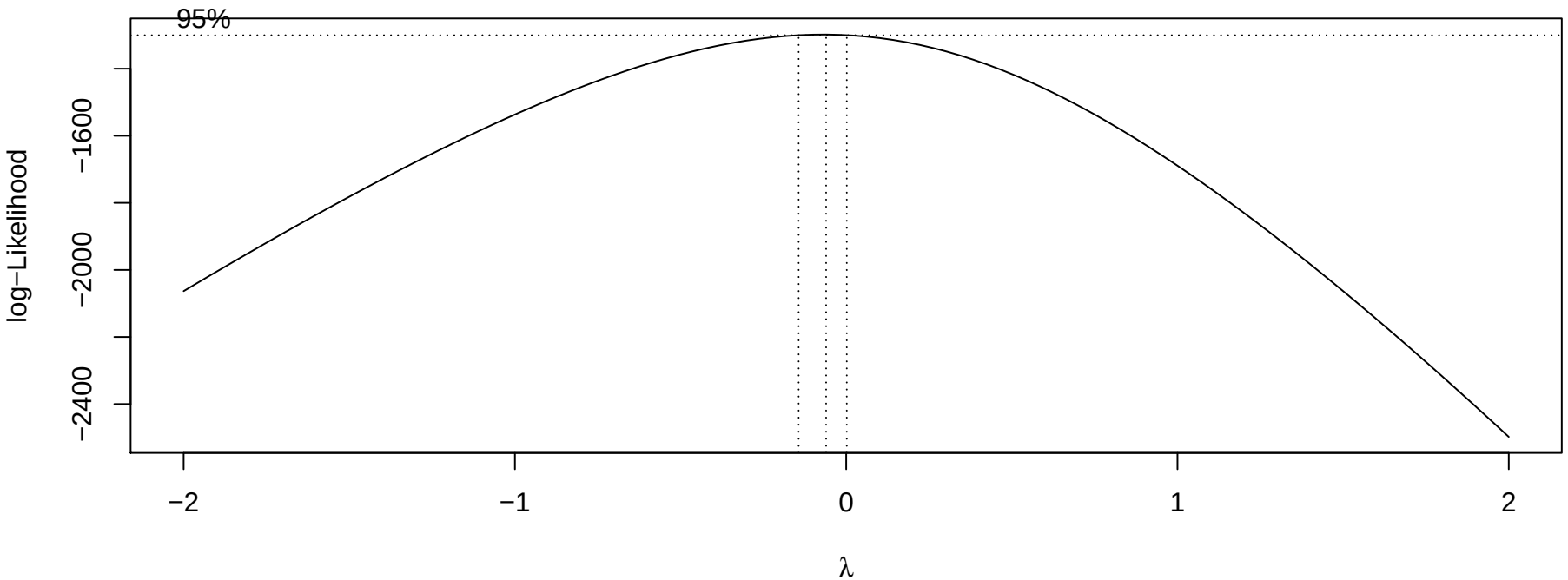
- On the normal quantile plot:
  - highest (most positive) residuals are *way* too high
  - distribution of residuals skewed to right (not normal at all)
- On plot of absolute residuals:
  - size of residuals getting bigger as fitted values increase
  - predictions getting more variable as fitted values increase
  - that is, predictions getting *less accurate* as fitted values increase, but predictions should be equally accurate all way along.
- Both indicate problems with regression, of kind that transformation of response often fixes: that is, predict *function* of response `timedrs` instead of `timedrs` itself.

Box-Cox transformations

- Taking log of `timedrs` and having it work: lucky guess. How to find good transformation?
- Box-Cox again.
- Extra problem: some of `timedrs` values are 0, but Box-Cox expects all $+$. Note response for `boxcox`:

```
boxcox(timedrs + 1 ~ phyheal + menheal + stress, data = visits
```

Try 1



Comments on try 1

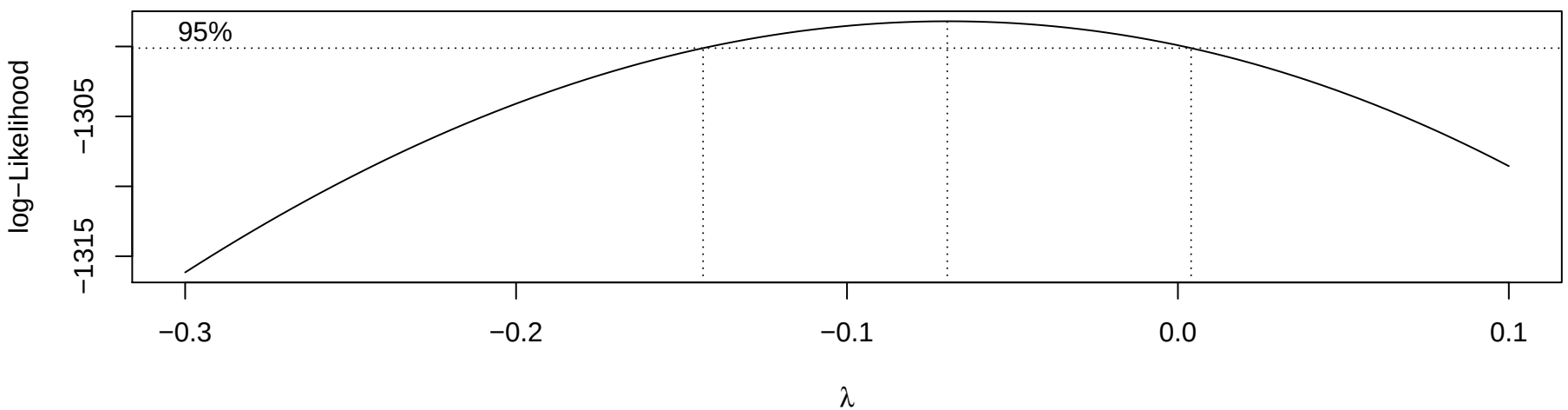
- Best: λ just less than zero.
- Hard to see scale.
- Focus on λ in $(-0.3, 0.1)$:

```
my.lambda <- seq(-0.3, 0.1, 0.01)
my.lambda
```

```
##  [1] -0.30 -0.29 -0.28 -0.27 -0.26 -0.25 -0.24 -0.23 -0.22
## [10] -0.21 -0.20 -0.19 -0.18 -0.17 -0.16 -0.15 -0.14 -0.13
## [19] -0.12 -0.11 -0.10 -0.09 -0.08 -0.07 -0.06 -0.05 -0.04
## [28] -0.03 -0.02 -0.01  0.00  0.01  0.02  0.03  0.04  0.05
## [37]  0.06  0.07  0.08  0.09  0.10
```

Try 2

```
boxcox(timedrs + 1 ~ phyheal + menheal + stress,
  lambda = my.lambda,
  data = visits
)
```



Comments

- Best: λ just about -0.07.
- CI for λ about $(-0.14, 0.01)$.
- Only nearby round number: $\lambda = 0$, log transformation.

Fixing the problems

- Try regression again, with transformed response instead of original one.
- Then check residual plot to see that it is OK now.

```
visits.3 <- lm(log(timedrs + 1) ~ phyheal + menheal + stress,
  data = visits
)
```

- `timedrs+1` because some `timedrs` values 0, can't take log of 0.
- Won't usually need to worry about this, but when response could be zero/negative, fix that before transformation.

Output

```
summary(visits.3)
```

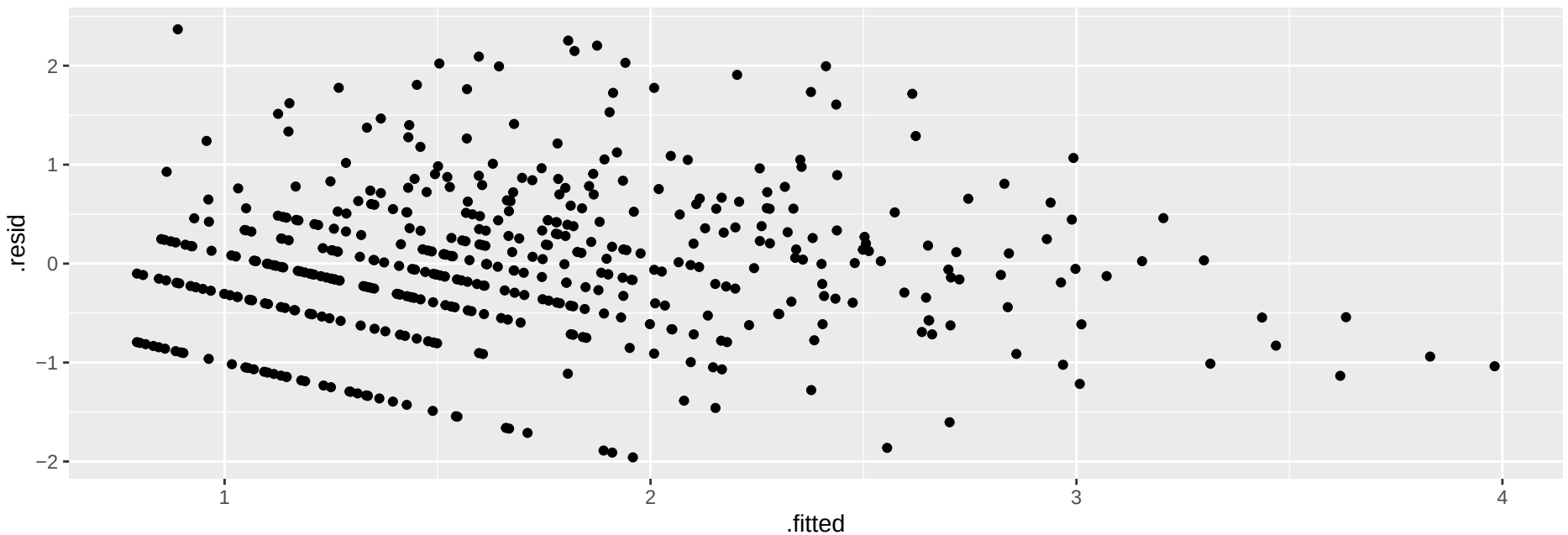
```
##
## Call:
## lm(formula = log(timedrs + 1) ~ phyheal + menheal + stress, data = visits)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.95865 -0.44076 -0.02331  0.42304  2.36797
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.3903862  0.0882908   4.422 1.22e-05 ***
## phyheal     0.2019361  0.0173624  11.631  < 2e-16 ***
## menheal     0.0071442  0.0101335   0.705    0.481
## stress      0.0013158  0.0002837   4.638 4.58e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7625 on 461 degrees of freedom
## Multiple R-squared:  0.3682, Adjusted R-squared:  0.3641
## F-statistic: 89.56 on 3 and 461 DF,  p-value: < 2.2e-16
```

Comments

- Model as a whole strongly significant again
- R-sq higher than before (37% vs. 22%) suggesting things more linear now
- Same conclusion re `menheal`: can take out of regression.
- Should look at residual plots (next pages). Have we fixed problems?

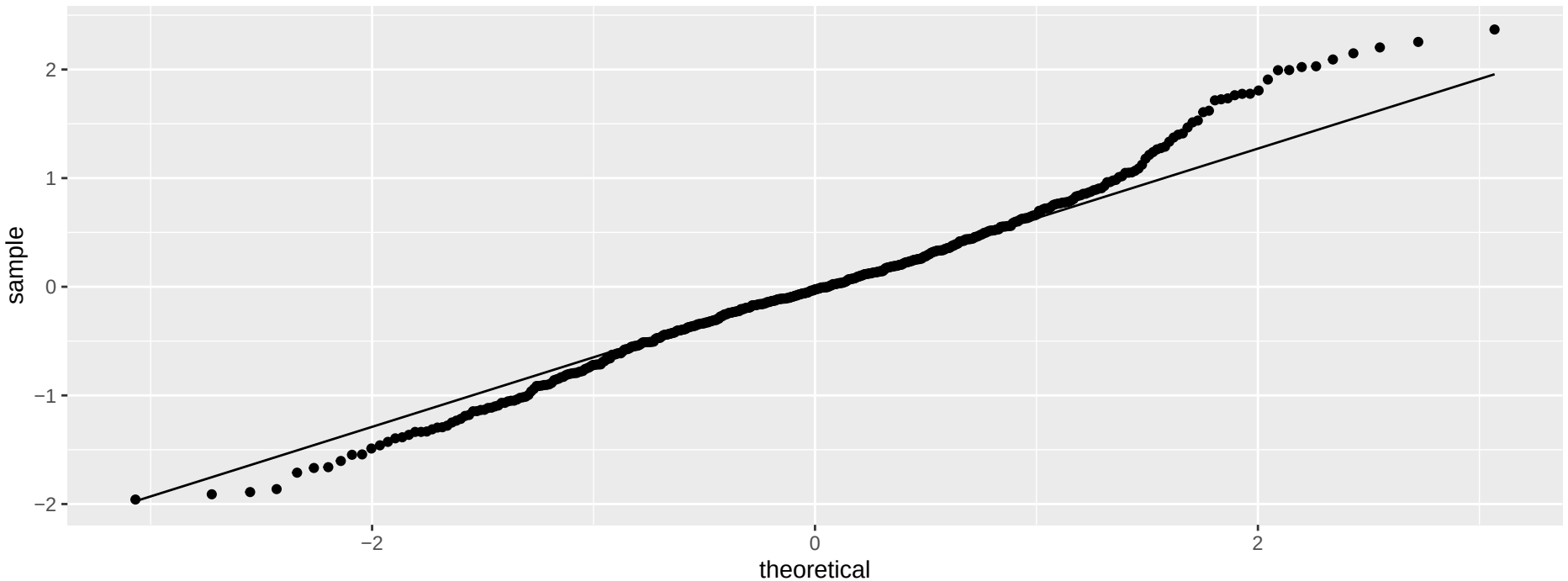
Residuals against fitted values

```
ggplot(visits.3, aes(x = .fitted, y = .resid)) +
  geom_point()
```



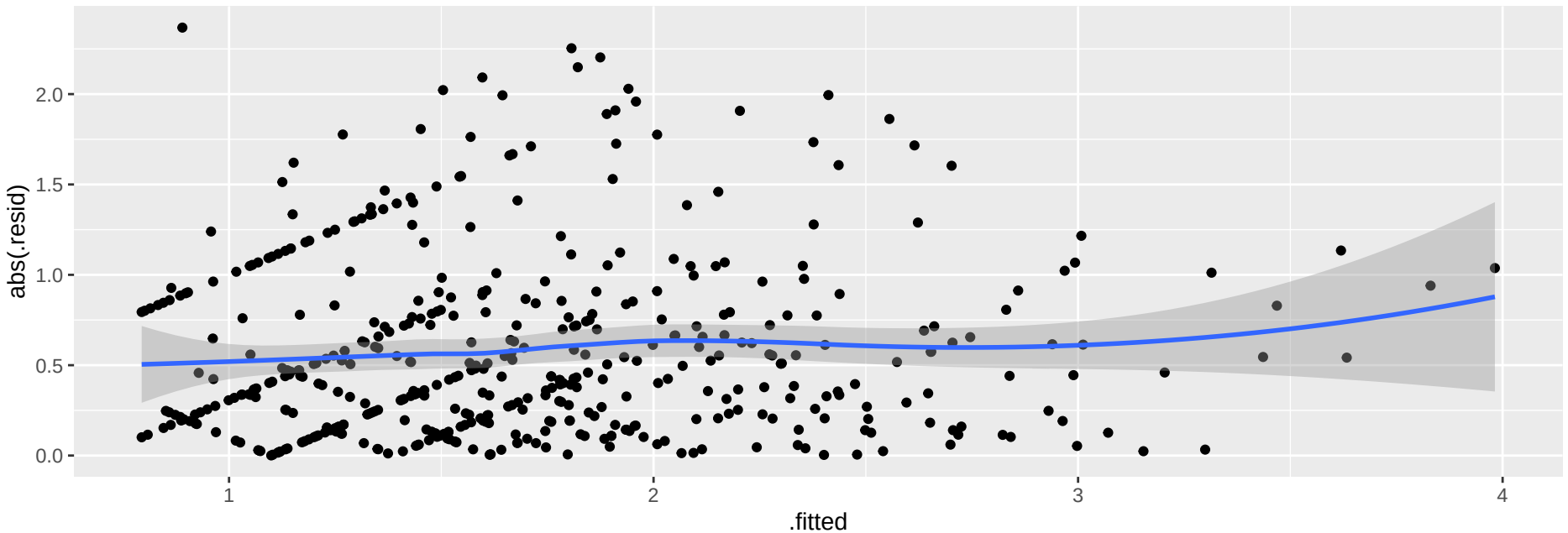
Normal quantile plot of residuals

```
ggplot(visits.3, aes(sample = .resid)) + stat_qq() + stat_qq_l
```



Absolute residuals against fitted

```
ggplot(visits.3, aes(x = .fitted, y = abs(.resid))) +
  geom_point() + geom_smooth()
```



Comments

- Residuals vs. fitted looks a lot more random.
- Normal quantile plot looks a lot more normal (though still a little right-skewness)
- Absolute residuals: not so much trend (though still some).
- Not perfect, but much improved.

Testing more than one x at once

- The t-tests test only whether one variable could be taken out of the regression you're looking at.
- To test significance of more than one variable at once, fit model with and without variables
  - then use `anova` to compare fit of models:

```
visits.5 <- lm(log(timedrs + 1) ~ phyheal + menheal + stress,
               data = visits)
visits.6 <- lm(log(timedrs + 1) ~ stress, data = visits)
```

Results of tests

```
anova(visits.6, visits.5)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
463	371.4729	NA	NA	NA	NA
461	268.0087	2	103.4642	88.98405	0

- Models don't fit equally well, so bigger one fits better.
- Or "taking both variables out makes the fit worse, so don't do it".
- Taking out those x's is a mistake. Or putting them in is a good idea.

The punting data

Data set `punting.txt` contains 4 variables for 13 right-footed football kickers (punters): left leg and right leg strength (lbs), distance punted (ft), another variable called "fred". Predict punting distance from other variables:

left	right	punt	fred
170	170	162.50	171
130	140	144.0	136
170	180	174.50	174
160	160	163.50	161
150	170	192.0	159
150	150	171.75	151
180	170	162.0	174
110	110	104.83	111
110	120	105.67	114
120	130	117.58	126
140	120	140.25	129
130	140	150.17	136
150	160	165.17	154

Reading in

- Separated by *multiple spaces* with *columns lined up*:

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/punting.txt
punting <- read_table(my_url)
```

```
##
## -- Column specification --------------------------------
## cols(
##   left = col_double(),
##   right = col_double(),
##   punt = col_double(),
##   fred = col_double()
## )
```

The data

```
punting
```

left	right	punt	fred
170	170	162.50	171
130	140	144.00	136
170	180	174.50	174
160	160	163.50	161
150	170	192.00	159
150	150	171.75	151
180	170	162.00	174
110	110	104.83	111
110	120	105.67	114
120	130	117.58	126
140	120	140.25	129
130	140	150.17	136
150	160	165.17	154

Regression and output

```
punting.1 <- lm(punt ~ left + right + fred, data = punting)
glance(punting.1)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	log-Lik	AIC	BIC	deviance	df.residual
0.778140	0.704186	14.6752	10.52204	4.002670	3	-50.97606	111.9521	114.7769	1938.254	9

```
tidy(punting.1)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-4.6855434	29.117224	-0.1609200	0.8757106
left	0.2678651	2.111096	0.1268844	0.9018215
right	1.0524055	2.147710	0.4900129	0.6358478
fred	-0.2672438	4.226613	-0.0632288	0.9509663

Comments

- Overall regression strongly significant, R-sq high.
- None of the x's significant! Why?
- t-tests only say that you could take any one of the x's out without damaging the fit; doesn't matter which one.
- Explanation: look at *correlations*.

The correlations

```
cor(punting)
```

```
##              left     right      punt      fred
## left  1.0000000 0.8957224 0.8117368 0.9722632
## right 0.8957224 1.0000000 0.8805469 0.9728784
## punt  0.8117368 0.8805469 1.0000000 0.8679507
## fred  0.9722632 0.9728784 0.8679507 1.0000000
```

- *All* correlations are high: x's with `punt` (good) and with each other (bad, at least confusing).
- What to do? Probably do just as well to pick one variable, say `right` since kickers are right-footed.

Just `right`

```
punting.2 <- lm(punt ~ right, data = punting)
anova(punting.2, punting.1)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
11	1962.516	NA	NA	NA	NA
9	1938.254	2	24.26262	0.05633	0.9455577

No significant loss by dropping other two variables.

Comparing R-squareds

```
summary(punting.1)$r.squared
```

```
## [1] 0.7781401
```

```
summary(punting.2)$r.squared
```

```
## [1] 0.7753629
```

Basically no difference. In regression (over), `right` significant:

Regression results

```
tidy(punting.2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-3.693037	25.2648659	-0.1461728	0.8864293
right	1.042672	0.1692152	6.1618066	0.0000709

But...

- Maybe we got the *form* of the relationship with `left` wrong.
- Check: plot *residuals* from previous regression (without `left`) against `left`.
- Residuals here are "punting distance adjusted for right leg strength''.
- If there is some kind of relationship with `left`, we should include in model.
- Plot of residuals against original variable: `augment` from `broom`.

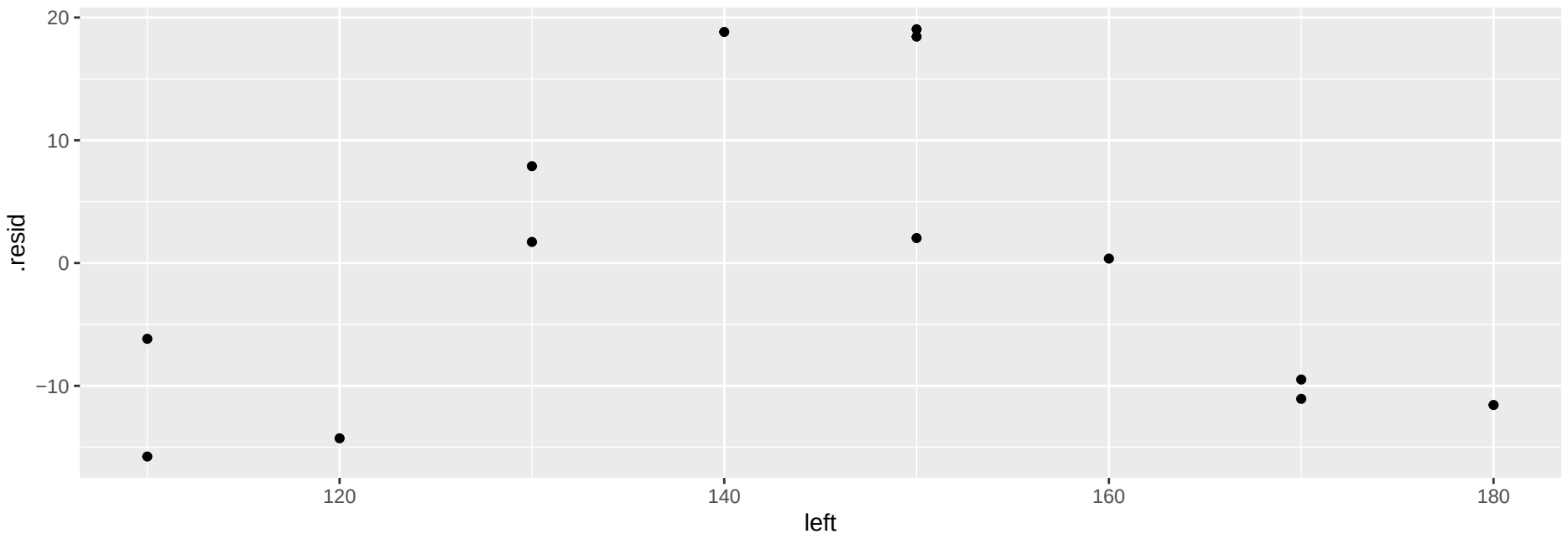
Augmenting punting.2

```
punting.2 %>% augment(punting) -> punting.2.aug
punting.2.aug %>% slice(1:8)
```

left	right	punt	fred	.fitted	.resid	.std.resid	.hat	.sig
170	170	162.50	171	173.5611	-11.0611358	-0.9018235	0.1567901	13.48
130	140	144.00	136	142.2810	1.7190123	0.1346466	0.0864198	13.99
170	180	174.50	174	183.9879	-9.4878519	-0.8171925	0.2444444	13.57
160	160	163.50	161	163.1344	0.3655802	0.0288702	0.1012346	14.00
150	170	192.00	159	173.5611	18.4388642	1.5033358	0.1567901	12.48
150	150	171.75	151	152.7077	19.0422963	1.4845376	0.0777778	12.52
180	170	162.00	174	173.5611	-11.5611358	-0.9425890	0.1567901	13.43
110	110	104.83	111	111.0008	-6.1708395	-0.5541435	0.3049383	13.81

Residuals against `left`

```
ggplot(punting.2.aug, aes(x = left, y = .resid)) +
  geom_point()
```



Comments

- There is a *curved* relationship with `left`.
- We should add `left`-squared to the regression (and therefore put `left` back in when we do that):

```
punting.3 <- lm(punt ~ left + I(left^2) + right,
  data = punting
)
```

Regression with `left-squared`

```
summary(punting.3)
```

```
##
## Call:
## lm(formula = punt ~ left + I(left^2) + right, data = punting)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3777  -5.3599   0.0459   4.5088  13.2669
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.623e+02  9.902e+01  -4.669  0.00117 **
## left         6.888e+00  1.462e+00   4.710  0.00110 **
## I(left^2)   -2.302e-02  4.927e-03  -4.672  0.00117 **
## right        7.396e-01  2.292e-01   3.227  0.01038 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.931 on 9 degrees of freedom
## Multiple R-squared:  0.9352, Adjusted R-squared:  0.9136
## F-statistic:  43.3 on 3 and 9 DF,  p-value: 1.13e-05
```

Comments

- This was definitely a good idea (R-squared has clearly increased).
- We would never have seen it without plotting residuals from `punting.2` (without `left`) against `left`.
- Negative slope for `leftsq` means that increased left-leg strength only increases punting distance up to a point: beyond that, it decreases again.