# Regression revisited

# Regression

▶ Use regression when one variable is an outcome (*response*, $y$).

▶ See if/how response depends on other variable(s), *explanatory*, $x_1, x_2, ....$

▶ 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

```r
library(MASS) # for Box-Cox, later
library(tidyverse)
library(broom)
library(marginaleffects)
library(conflicted)
conflict_prefer("select", "dplyr")
```

# 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://ritsokiguess.site/datafiles/sleep.txt"
sleep <- read_delim(my_url, " ")
```
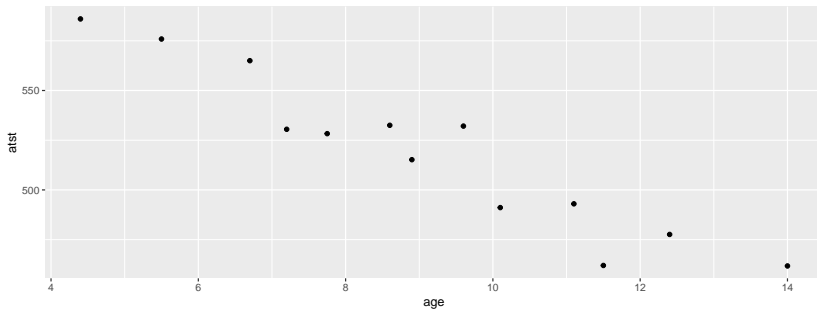
# Check data

```
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()
```
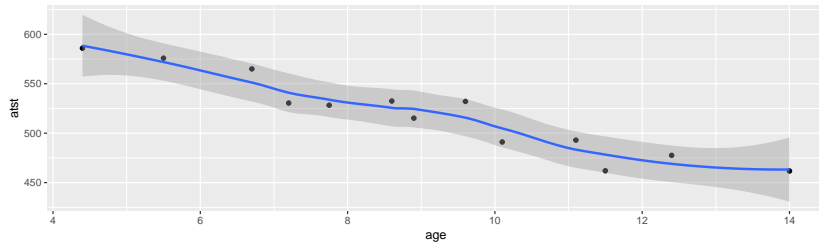
# The regression

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)
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)    646.      12.9      50.0 2.49e-14
2 age           -14.0       1.37    -10.3 5.70e- 7
```

also one-line summary of model:

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

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic     p.value    df
      <dbl>         <dbl> <dbl>     <dbl>       <dbl> <dbl>
1     0.905         0.897  13.2      105. 0.000000570     1
# i 6 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
#   deviance <dbl>, df.residual <int>, nobs <int>
```

# Broom part 2

```
sleep.1 %>% augment(sleep)
```

```
# A tibble: 13 x 8
     atst   age .fitted .resid   .hat .sigma .cooksd
    <dbl> <dbl>   <dbl>  <dbl>  <dbl>  <dbl>   <dbl>
 1   586   4.4    585.    1.30  0.312   13.8 0.00320
 2   462.  14     450.   11.8   0.341   13.0 0.319
 3   491.  10.1   505.  -13.6   0.0887  13.0 0.0568
 4   565   6.7    552.   12.6   0.137   13.1 0.0844
 5   462  11.5    485.  -23.0   0.141   11.3 0.294
 6   532.  9.6    512.   20.4   0.0801  12.0 0.114
 7   478. 12.4    472.    5.23  0.198   13.7 0.0243
 8   515.  8.9    522.   -6.32  0.0772  13.6 0.0105
 9   493  11.1    491.    2.37  0.122   13.8 0.00258
10   528.  7.75   538.   -9.37  0.0954  13.4 0.0296
11   576.  5.5    569.    6.64  0.214   13.6 0.0441
12   532.  8.6    526.    6.77  0.0792  13.6 0.0124
13   530.  7.2    545.  -14.9   0.114   12.9 0.0933
# i 1 more variable: .std.resid <dbl>
```

# 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.

# The `marginaleffects` package 1/2

To get predictions for specific values, set up a dataframe with those values first:

```
new <- datagrid(model = sleep.1, age = c(10, 5))
new
```

```
       atst age
1 519.3038   10
2 519.3038    5
```

Any variables in the dataframe that you don't specify are set to their mean values (quantitative) or most common category (categorical).

Then feed into `newdata` in `predictions`. This contains a lot of columns, so you probably want only to display the ones you care about:

```
cbind(predictions(sleep.1, newdata = new)) %>%
  select(estimate, conf.low, conf.high, age)
```

```
  estimate conf.low conf.high age
1 506.0729 498.4899  513.6558  10
2 576.2781 563.2588  589.2974   5
```

The confidence limits are a 95% confidence interval for the mean response at that age.

# Prediction intervals

These are obtained (instead) with `predict` as below. Use the same dataframe `new` as before:

```
pp <- predict(sleep.1, new, interval = "p")
cbind(new, pp) %>% select(-atst)
```

```
  age      fit      lwr      upr
1  10 506.0729 475.8982 536.2475
2   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))
```
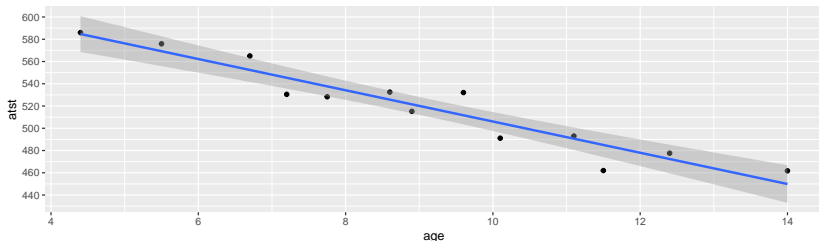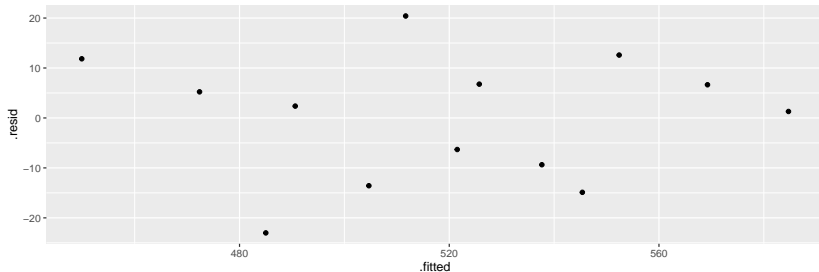
# Diagnostics

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(
```
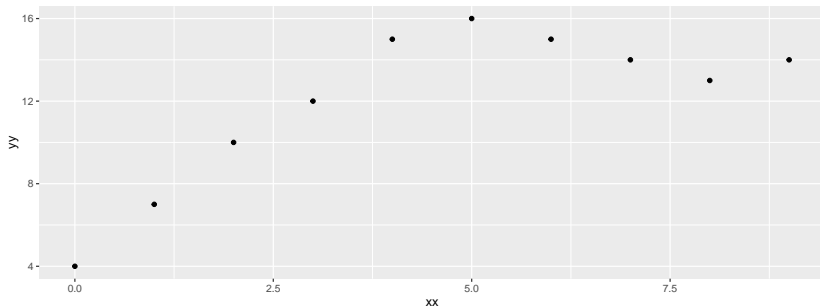
# An inappropriate regression

Different data:

```r
my_url <- "http://ritsokiguess.site/datafiles/curvy.txt"
curvy <- read_delim(my_url, " ")
```

# 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
```
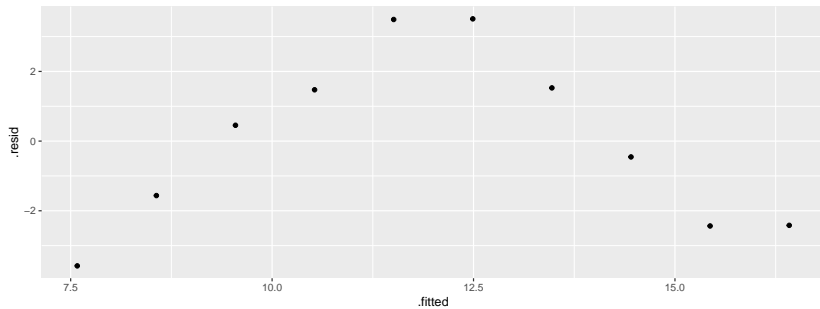
# Residual plot

```
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)
```

```
# A tibble: 3 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)     3.9     0.773       5.04 0.00149
2 xx             3.74     0.400       9.36 0.0000331
3 I(xx^2)       -0.307    0.0428     -7.17 0.000182
```

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

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>
1     0.950         0.936 0.983      66.8 0.0000275     2
# i 6 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
#   deviance <dbl>, df.residual <int>, nobs <int>
```
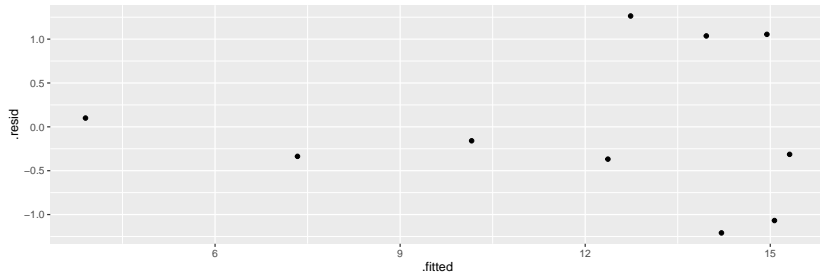
# Comments

- $x$-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
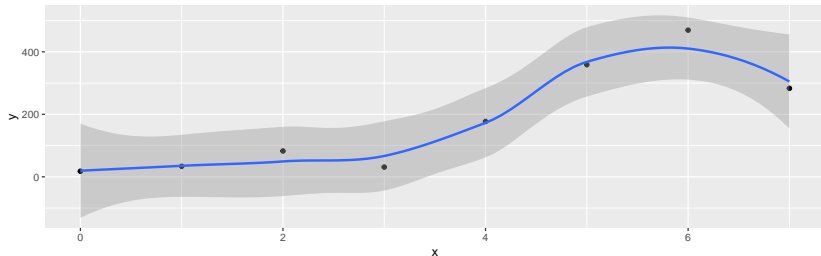
```
my_url <- "http://ritsokiguess.site/datafiles/madeup2.csv"
madeup <- read_csv(my_url)
madeup
```

```
# A tibble: 8 x 3
   ...1     x     y
  <dbl> <dbl> <dbl>
1     1     0  17.9
2     2     1  33.6
3     3     2  82.7
4     4     3  31.2
5     5     4 177.
6     6     5 359.
7     7     6 469.
8     8     7 283.
```

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()
```
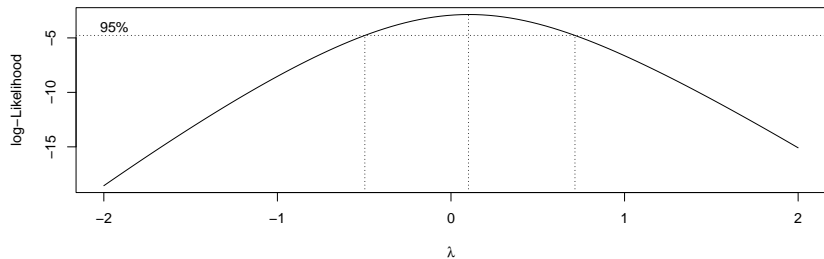
# Running Box-Cox

▶ `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

▶ $\lambda$ (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 $\lambda$ (here about 0.1).

▶ Outer dotted lines mark 95% CI for $\lambda$, 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 $\lambda$ 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()
```
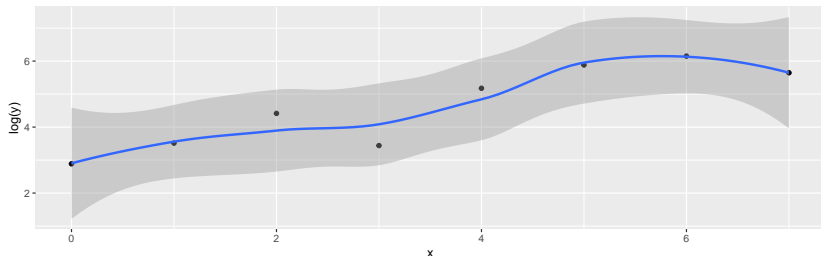


Looks much straighter.

# Regression with transformed $y$

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

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>
1     0.811         0.779 0.588      25.7 0.00228     1
# i 6 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
#   deviance <dbl>, df.residual <int>, nobs <int>
```

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

```
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)     3.03     0.379      7.98 0.000206
2 x               0.460    0.0907     5.07 0.00228
```

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 $x$s 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://ritsokiguess.site/datafiles/regressx.txt"
visits <- read_delim(my_url, " ")
```

# Check data

```
visits
```

```
# A tibble: 465 x 5
   subjno timedrs phyheal menheal stress
    <dbl>   <dbl>   <dbl>   <dbl>  <dbl>
 1      1       1       5       8    265
 2      2       3       4       6    415
 3      3       0       3       4     92
 4      4      13       2       2    241
 5      5      15       3       6     86
 6      6       3       5       5    247
 7      7       2       5       6     13
 8      8       0       4       5     12
 9      9       7       5       4    269
10     10       4       3       9    391
# i 455 more rows
```

# Fit multiple regression

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


Call:
lm(formula = timedrs ~ phyheal + menheal + stress, data = v

Residuals:
    Min      1Q  Median      3Q     Max
-14.792  -4.353  -1.815   0.902  65.886

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.704848   1.124195  -3.296 0.001058 **
phyheal      1.786948   0.221074   8.083 5.6e-15 ***
menheal     -0.009666   0.129029  -0.075 0.940318
stress       0.013615   0.003612   3.769 0.000185 ***
```

# 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`.

▶ The physical health and stress variables definitely 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)
summary(visits.2)


Call:
lm(formula = timedrs ~ menheal, data = visits)

Residuals:
    Min      1Q  Median      3Q     Max
-13.826  -5.150  -2.818   1.177  72.513

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.8159     0.8702   4.385 1.44e-05 ***
menheal       0.6672     0.1173   5.688 2.28e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.6 on 463 degrees of freedom
Multiple R-squared:  0.06532,   Adjusted R-squared:  0.0633
F-statistic: 32.35 on 1 and 463 DF,  p-value: 2.279e-08
```

# 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.
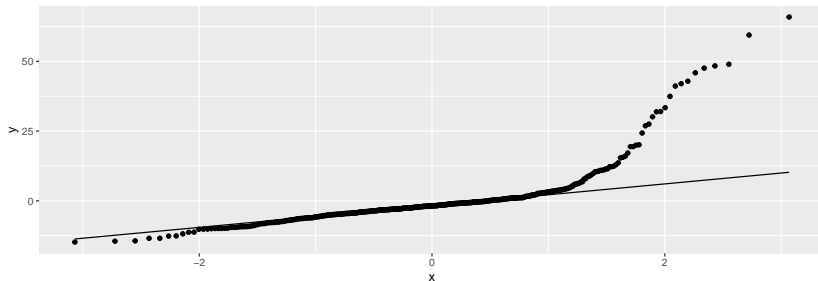
# Residual plot (from `timedrs` on all)

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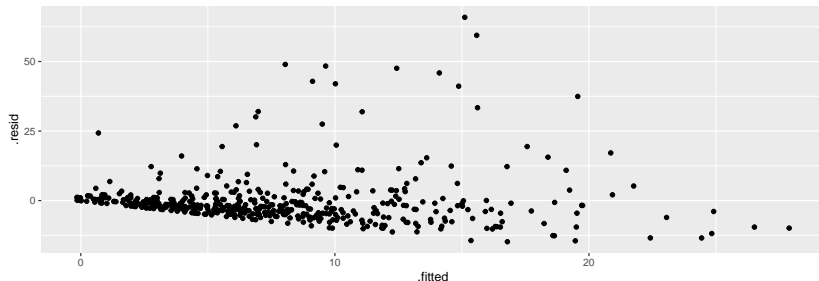


Apparently random. But…

# Normal quantile plot of residuals

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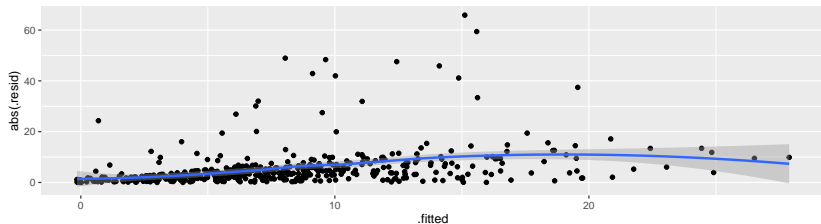


Not normal at all; upper tail is way too long.

# Absolute residuals

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

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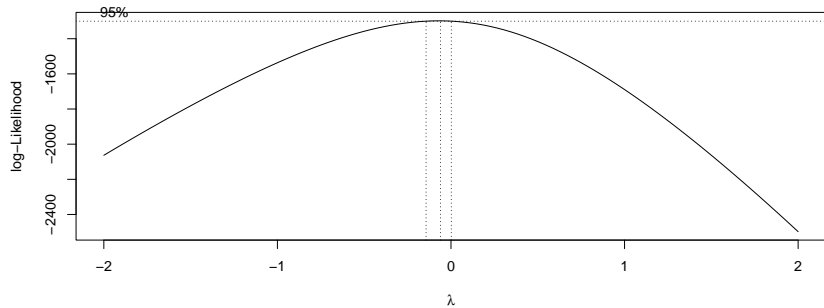
# Comments

▶ 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 = vis
```
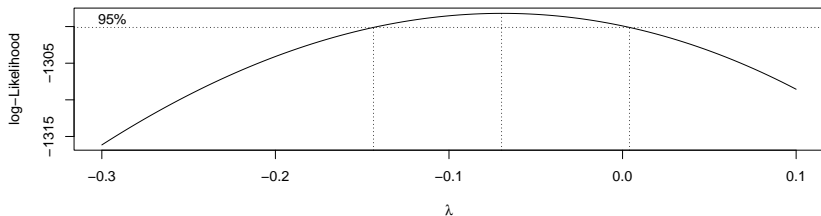
# Try 1

# Comments on try 1

▶ Best: $\lambda$ just less than zero.

▶ Hard to see scale.

▶ Focus on $\lambda$ 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: $\lambda$ just about $-0.07$.
- CI for $\lambda$ 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 + stres
  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
```
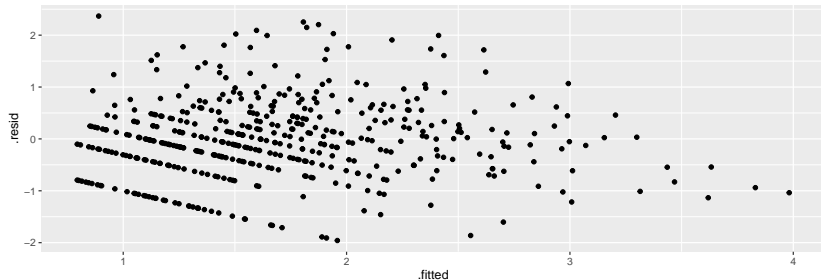
# Comments

▶ 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?
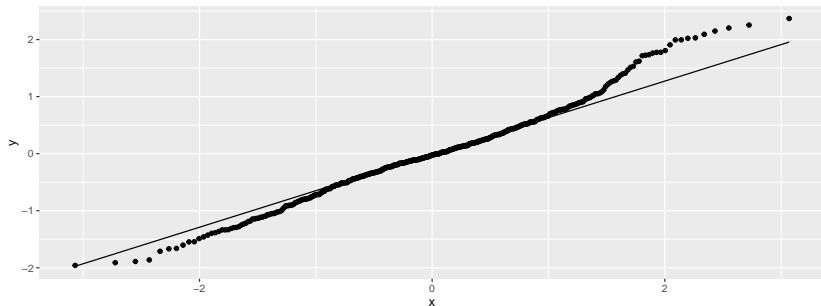
# Residuals against fitted values

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

# Normal quantile plot of residuals

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

# Absolute residuals against fitted

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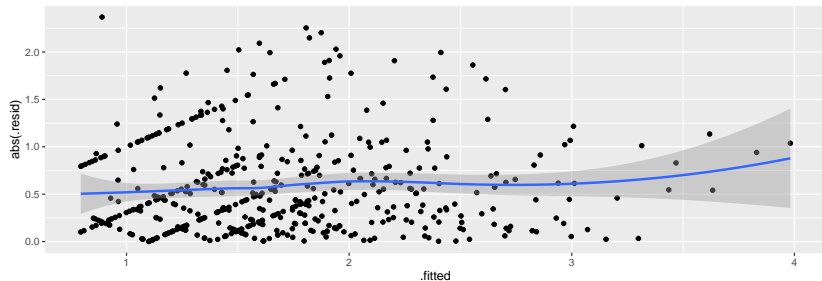
# Comments

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

# Results of tests

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

Analysis of Variance Table

Model 1: log(timedrs + 1) ~ stress
Model 2: log(timedrs + 1) ~ phyheal + menheal + stress
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    463 371.47
2    461 268.01  2    103.46 88.984 < 2.2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

▶ 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://ritsokiguess.site/datafiles/punting.txt"
punting <- read_table(my_url)
```

# The data

```
punting
```

```
# A tibble: 13 x 4
    left right  punt  fred
   <dbl> <dbl> <dbl> <dbl>
 1   170   170  162.   171
 2   130   140  144    136
 3   170   180  174.   174
 4   160   160  164.   161
 5   150   170  192    159
 6   150   150  172.   151
 7   180   170  162    174
 8   110   110  105.   111
 9   110   120  106.   114
10   120   130  118.   126
11   140   120  140.   129
12   130   140  150.   136
13   150   160  165.   154
```

# Regression and output

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

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>
1     0.778         0.704  14.7      10.5 0.00267     3
# i 6 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
#   deviance <dbl>, df.residual <int>, nobs <int>
```

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

```
# A tibble: 4 x 5
  term        estimate std.error statistic p.value
  <chr>          <dbl>     <dbl>     <dbl>   <dbl>
1 (Intercept)    -4.69      29.1    -0.161   0.876
2 left            0.268      2.11    0.127   0.902
3 right           1.05       2.15    0.490   0.636
4 fred           -0.267      4.23   -0.0632  0.951
```

## 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)

Analysis of Variance Table

Model 1: punt ~ right
Model 2: punt ~ left + right + fred
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     11 1962.5
2      9 1938.2  2    24.263 0.0563 0.9456
```

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)
```

```
# A tibble: 2 x 5
  term         estimate std.error statistic   p.value
  <chr>           <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)     -3.69      25.3    -0.146 0.886
2 right            1.04       0.169   6.16  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
```

```
# A tibble: 13 x 10
    left right  punt  fred .fitted .resid   .hat .sigma
   <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>  <dbl>  <dbl>
 1   170   170  162.   171    174. -11.1  0.157    13.5
 2   130   140  144    136    142.   1.72 0.0864   14.0
 3   170   180  174.   174    184.  -9.49 0.244    13.6
 4   160   160  164.   161    163.   0.366 0.101   14.0
 5   150   170  192    159    174.  18.4  0.157    12.5
 6   150   150  172.   151    153.  19.0  0.0778   12.5
 7   180   170  162    174    174. -11.6  0.157    13.4
 8   110   110  105.   111    111.  -6.17 0.305    13.8
 9   110   120  106.   114    121. -15.8  0.2      12.9
10   120   130  118.   126    132. -14.3  0.127    13.1
11   140   120  140.   129    121.  18.8  0.2      12.3
12   130   140  150.   136    142.   7.89 0.0864   13.8
13   150   160  165.   154    163.   2.04 0.101    14.0
# i 2 more variables: .cooksd <dbl>, .std.resid <dbl>
```
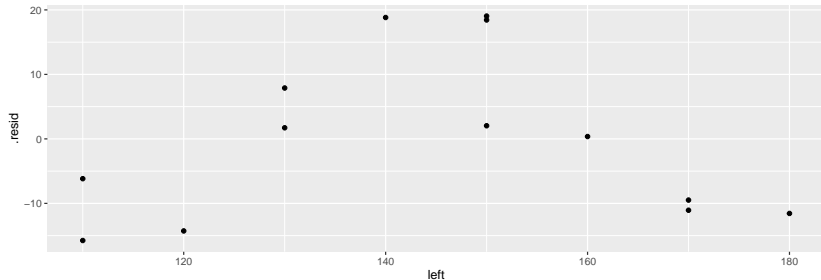
# Residuals against `left`

```
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.