432 Class 21 Slides

github.com/THOMASELOVE/2019-432

2019-04-16

Preliminaries

```r
library(skimr); library(MASS); library(robustbase)
library(quantreg); library(lmtest); library(sandwich)
library(boot); library(rms); library(survival)
library(OIsurv); library(survminer); library(broom)
library(tidyverse)

decim <- function(x, k) format(round(x, k), nsmall=k)
```

Today's Agenda

- Regression on Time-to-event data
    - Cox Proportional Hazards Model
- Robust Linear Regression Methods
    - with Huber weights
    - with bisquare weights (biweights)
    - Bounded Influence Regression & Least Trimmed Squares
    - Penalized Least Squares using `ols` in `rms` package
    - Quantile Regression on the Median

Survival Analysis / Cox Regression

A Survival Analysis Example

Source: Chen and Peace (2011) *Clinical Trial Data Analysis Using R*, CRC Press, section 5.1

```
brca <- read.csv("data/breast_cancer.csv") %>% tbl_df
```

The `brca` trial

The `brca` data describes a parallel randomized trial of three treatments, adjuvant to surgery in the treatment of patients with stage-2 carcinoma of the breast. The three treatment groups are:

- `S+CT` = Surgery plus one year of chemotherapy
- `S+IT` = Surgery plus one year of immunotherapy
- `S+CT+IT` = Surgery plus one year of chemotherapy and immunotherapy

The measure of efficacy were "time to death" in weeks. In addition to `treat`, our variables are:

- `trial_weeks`: time in the study, in weeks, to death or censoring
- `last_alive`: 1 if alive at last follow-up (and thus censored), 0 if dead
- `age`: age in years at the start of the trial

brca tibble

```
# A tibble: 31 x 5
   subject treat   trial_weeks last_alive   age
   <fct>   <fct>         <int>      <int> <int>
 1 A01     S+CT            102          0    55
 2 A02     S+IT            192          0    62
 3 A03     S+CT+IT          73          0    72
 4 A04     S+CT             58          1    48
 5 A05     S+CT             48          1    26
 6 A06     S+IT            182          1    52
 7 A07     S+IT            196          1    50
 8 A08     S+CT            177          1    49
 9 A09     S+IT            191          1    62
10 A10     S+CT+IT          36          0    60
# ... with 21 more rows
```

Analytic Objectives

This is a typical right-censored survival data set with interest in the comparative analysis of the three treatments.

1. Does immunotherapy added to surgery plus chemotherapy improve survival? (Comparing S+CT+IT to S+CT)
2. Does chemotherapy add efficacy to surgery plus immunotherapy? (S+CT+IT vs. S+IT)
3. What is the effect of age on survival?

Create survival object

- `trial_weeks`: time in the study, in weeks, to death or censoring
- `last_alive`: 1 if alive at last follow-up (and thus censored), 0 if dead

So `last_alive` = 0 if the event (death) occurs.

What's next?

Create survival object

- trial_weeks: time in the study, in weeks, to death or censoring
- last_alive: 1 if alive at last follow-up (and thus censored), 0 if dead

So last_alive = 0 if the event (death) occurs.

```
brca$S <- with(brca, Surv(trial_weeks, last_alive == 0))

head(brca$S)
```

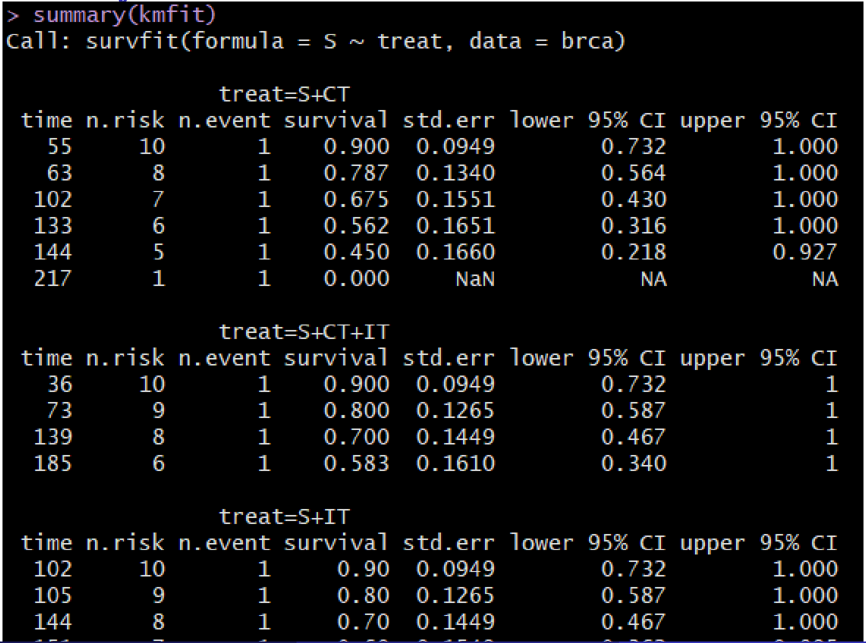
```
[1] 102  192   73   58+  48+ 182+
```

Build Kaplan-Meier Estimator

```r
kmfit <- survfit(S ~ treat, dat = brca)

print(kmfit, print.rmean = TRUE)
```

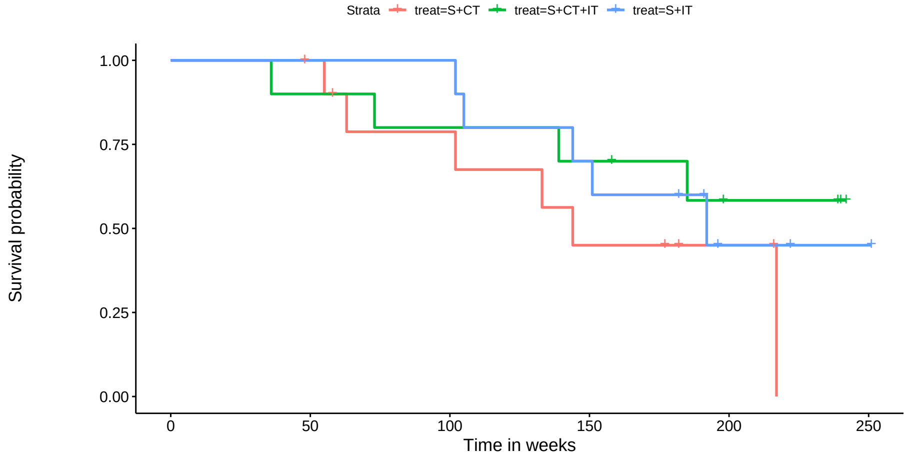
```
Call: survfit(formula = S ~ treat, data = brca)

              n events *rmean *se(rmean) median 0.95LCL
treat=S+CT    11      6    153       21.1    144     102
treat=S+CT+IT 10      4    188       23.7     NA     139
treat=S+IT    10      5    188       17.9    192     144
              0.95UCL
treat=S+CT         NA
treat=S+CT+IT      NA
treat=S+IT         NA
    * restricted mean with upper limit =  242
```

summary(kmfit)



```
> summary(kmfit)
Call: survfit(formula = S ~ treat, data = brca)

                treat=S+CT
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
   55     10       1    0.900  0.0949        0.732        1.000
   63      8       1    0.787  0.1340        0.564        1.000
  102      7       1    0.675  0.1551        0.430        1.000
  133      6       1    0.562  0.1651        0.316        1.000
  144      5       1    0.450  0.1660        0.218        0.927
  217      1       1    0.000     NaN           NA           NA

                treat=S+CT+IT
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
   36     10       1    0.900  0.0949        0.732            1
   73      9       1    0.800  0.1265        0.587            1
  139      8       1    0.700  0.1449        0.467            1
  185      6       1    0.583  0.1610        0.340            1

                treat=S+IT
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
  102     10       1     0.90  0.0949        0.732        1.000
  105      9       1     0.80  0.1265        0.587        1.000
  144      8       1     0.70  0.1449        0.467        1.000
```

K-M Plot via `survminer`



Number at risk

Strata	50	100	150	200	250
treat=S+CT	10	7	4	2	0
treat=S+CT+IT	9	8	7	4	0
treat=S+IT	10	10	7	2	1

Time in weeks

K-M Plot via survminer (code)

```
ggsurvplot(kmfit, data = brca,
           risk.table = TRUE,
           risk.table.height = 0.25,
           xlab = "Time in weeks")
```

Testing the difference between curves

```r
survdiff(S ~ treat, dat = brca)
```

```
Call:
survdiff(formula = S ~ treat, data = brca)

              N Observed Expected (O-E)^2/E (O-E)^2/V
treat=S+CT    11        6     3.80    1.2772    1.7647
treat=S+CT+IT 10        4     5.62    0.4676    0.7725
treat=S+IT    10        5     5.58    0.0605    0.0981

 Chisq= 1.9  on 2 degrees of freedom, p= 0.4
```

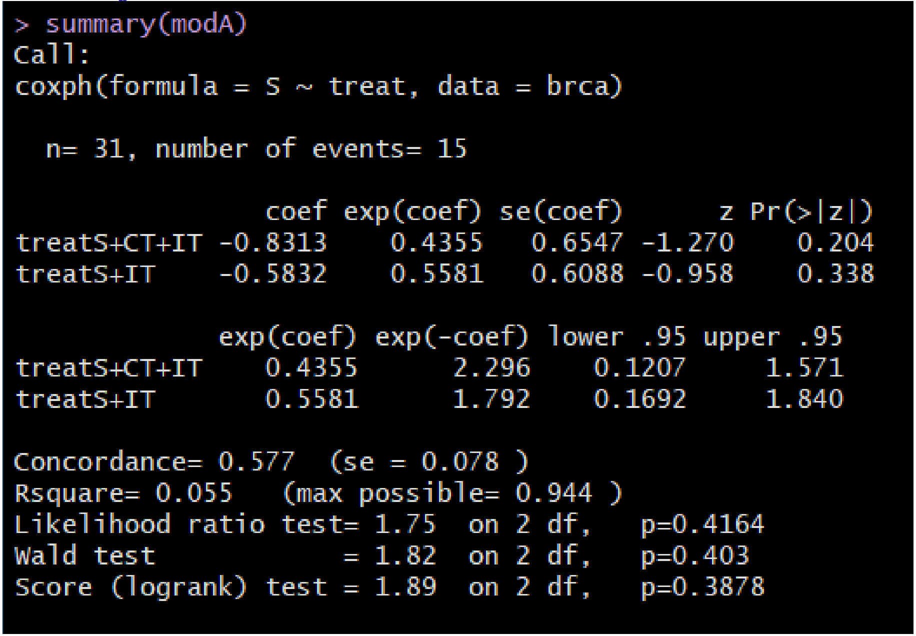
What do we conclude?

Fit Cox Model A: Treatment alone

```r
modA <- coxph(S ~ treat, data = brca)
modA
```

```
Call:
coxph(formula = S ~ treat, data = brca)

                coef exp(coef) se(coef)      z     p
treatS+CT+IT -0.8313    0.4355   0.6547 -1.270 0.204
treatS+IT    -0.5832    0.5581   0.6088 -0.958 0.338

Likelihood ratio test=1.75  on 2 df, p=0.4164
n= 31, number of events= 15
```

summary(modA)

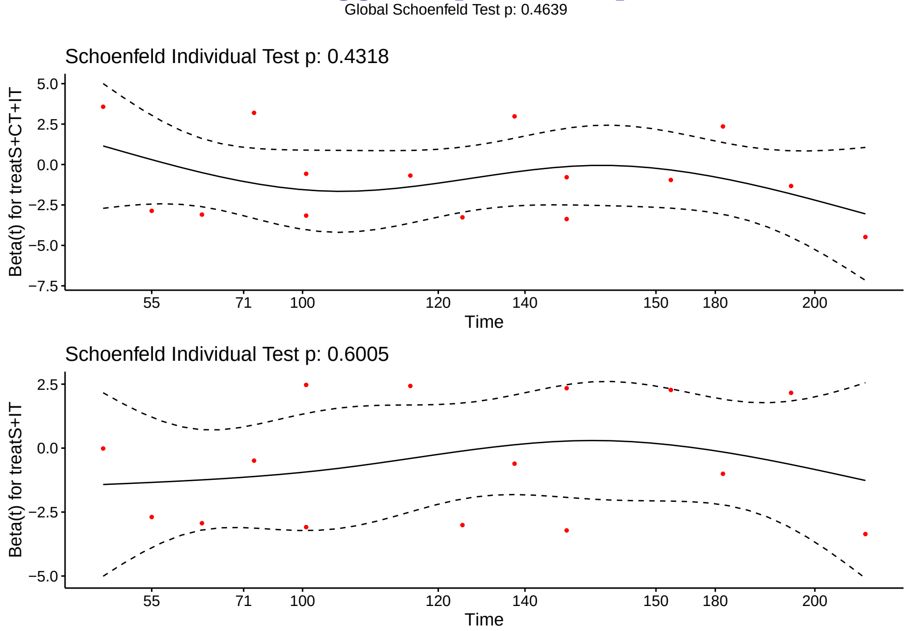

```
> summary(modA)
Call:
coxph(formula = S ~ treat, data = brca)

  n= 31, number of events= 15

                 coef exp(coef) se(coef)      z Pr(>|z|)
treatS+CT+IT -0.8313    0.4355   0.6547 -1.270    0.204
treatS+IT    -0.5832    0.5581   0.6088 -0.958    0.338

             exp(coef) exp(-coef) lower .95 upper .95
treatS+CT+IT    0.4355      2.296    0.1207     1.571
treatS+IT       0.5581      1.792    0.1692     1.840

Concordance= 0.577  (se = 0.078 )
Rsquare= 0.055   (max possible= 0.944 )
Likelihood ratio test= 1.75  on 2 df,   p=0.4164
Wald test            = 1.82  on 2 df,   p=0.403
Score (logrank) test = 1.89  on 2 df,   p=0.3878
```

Check Proportional Hazards Assumption

```r
cox.zph(modA)
```

```
                rho chisq     p
treatS+CT+IT -0.198 0.618 0.432
treatS+IT     0.138 0.274 0.601
GLOBAL           NA 1.536 0.464
```

Graphical PH Test `ggcoxzph(cox.zph(modA))`

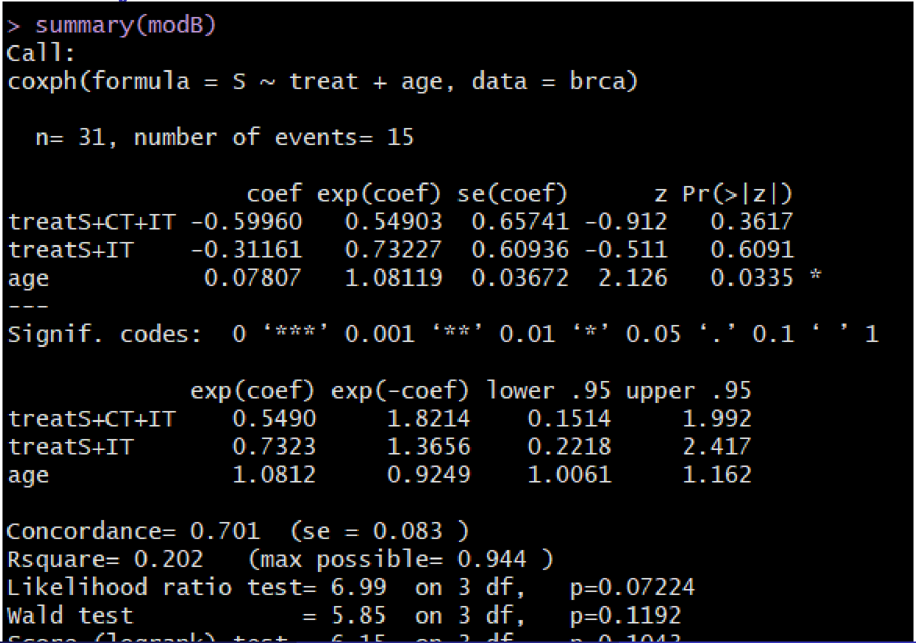


Fit Cox Model B: Treatment + Age

```r
modB <- coxph(S ~ treat + age, data = brca)
modB
```

```
Call:
coxph(formula = S ~ treat + age, data = brca)

                 coef exp(coef) se(coef)      z      p
treatS+CT+IT -0.59960   0.54903  0.65741 -0.912 0.3617
treatS+IT    -0.31161   0.73227  0.60936 -0.511 0.6091
age           0.07807   1.08119  0.03672  2.126 0.0335

Likelihood ratio test=6.99  on 3 df, p=0.07224
n= 31, number of events= 15
```

summary(modB)



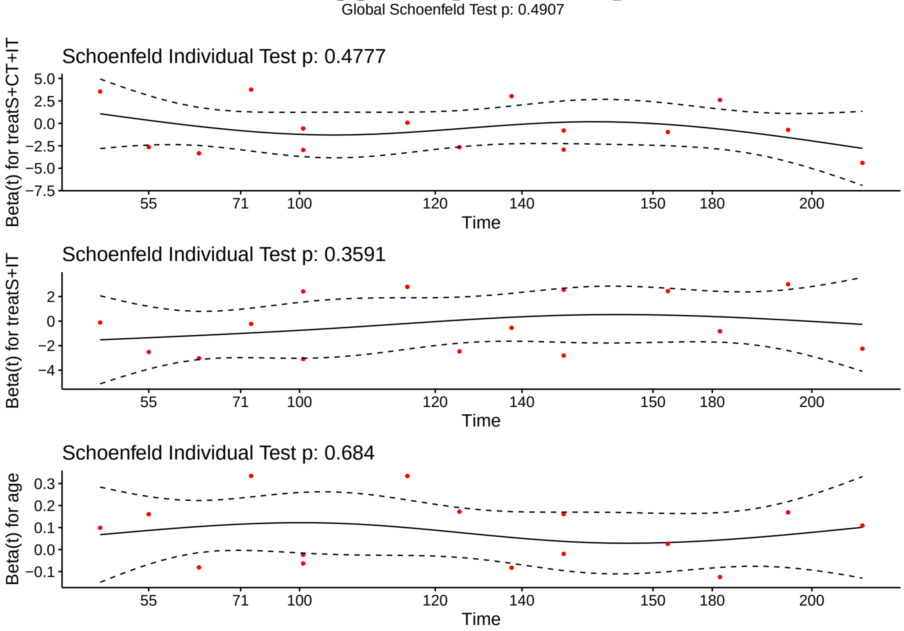
```
> summary(modB)
Call:
coxph(formula = S ~ treat + age, data = brca)

  n= 31, number of events= 15

                 coef exp(coef) se(coef)      z Pr(>|z|)  
treatS+CT+IT -0.59960   0.54903  0.65741 -0.912   0.3617  
treatS+IT    -0.31161   0.73227  0.60936 -0.511   0.6091  
age           0.07807   1.08119  0.03672  2.126   0.0335 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

             exp(coef) exp(-coef) lower .95 upper .95
treatS+CT+IT    0.5490     1.8214    0.1514     1.992
treatS+IT       0.7323     1.3656    0.2218     2.417
age             1.0812     0.9249    1.0061     1.162

Concordance= 0.701  (se = 0.083 )
Rsquare= 0.202   (max possible= 0.944 )
Likelihood ratio test= 6.99  on 3 df,   p=0.07224
Wald test            = 5.85  on 3 df,   p=0.1192
Score (logrank) test = 6.15  on 3 df,   p=0.1043
```

Proportional Hazards Assumption: Model B Check

```r
cox.zph(modB)
```

```
                rho chisq     p
treatS+CT+IT -0.179 0.504 0.478
treatS+IT     0.244 0.841 0.359
age          -0.106 0.166 0.684
GLOBAL           NA 2.416 0.491
```

Graphical PH Test `ggcoxzph(cox.zph(modB))`



What to do if the PH assumption is violated

- If the PH assumption fails on a categorical predictor, fit a Cox model stratified by that predictor (use `strata(var)` rather than `var` in the specification of the `coxph` model.)
- If the PH assumption is violated, this means the hazard isn't constant over time, so we could fit separate Cox models for a series of time intervals.
- Use an extension of the Cox model that permits covariates to vary over time.

Visit https://cran.r-project.org/web/packages/survival/vignettes/timedep.pdf for details on building the relevant data sets and models, with examples.

The `crimestat` data and an OLS fit

The `crimestat` data set

For each of 51 states (including the District of Columbia), we have the state's ID number, postal abbreviation and full name, as well as:

- **crime** - the violent crime rate per 100,000 people
- **poverty** - the official poverty rate (% of people living in poverty in the state/district) in 2014
- **single** - the percentage of households in the state/district led by a female householder with no spouse present and with her own children under 18 years living in the household in 2016
- **trump** - whether Donald Trump won the popular vote in the 2016 presidential election in that state/district (which we'll ignore for today)

The `crimestat` data set

```r
crimestat <- read.csv("data/crimestat.csv") %>% tbl_df
crimestat
```

```
# A tibble: 51 x 7
     sid state crime poverty single trump state.full
   <int> <fct> <dbl>   <dbl>  <dbl> <int> <fct>
 1     1 AL     427.    19.2   9.02     1 Alabama
 2     2 AK     636.    11.4   7.63     1 Alaska
 3     3 AZ     400.    18.2   8.31     1 Arizona
 4     4 AR     480.    18.7   9.41     1 Arkansas
 5     5 CA     396.    16.4   7.25     0 California
 6     6 CO     309.    12.1   6.75     0 Colorado
 7     7 CT     237.    10.8   8.04     0 Connecticut
 8     8 DE     489.    13     6.52     0 Delaware
 9     9 DC    1244.    18.4   8.41     0 District of Colum~
10    10 FL     540.    16.6   8.29     1 Florida
# ... with 41 more rows
```

Modeling `crime` with `poverty` and `single`

Our main goal will be to build a linear regression model to predict **crime** using centered versions of both **poverty** and **single**.

```
crimestat <- crimestat %>%
    mutate(pov_c = poverty - mean(poverty),
           single_c = single - mean(single))
```

Our original (OLS) model

```r
(mod1 <- lm(crime ~ pov_c + single_c, data = crimestat))
```

```

Call:
lm(formula = crime ~ pov_c + single_c, data = crimestat)

Coefficients:
(Intercept)        pov_c     single_c
     364.41        16.11        23.84
```

Significance of our coefficients?

```r
tidy(mod1)
```

```
# A tibble: 3 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)    364.      22.9      15.9  9.48e-21
2 pov_c           16.1      9.62      1.68 1.00e- 1
3 single_c        23.8     18.4       1.30 2.01e- 1
```

Robust Linear Regression with Huber Weights

Robust Linear Regression with Huber weights

There are several ways to do robust linear regression using M-estimation, including weighting using Huber and bisquare strategies.

- Robust linear regression here will make use of a method called iteratively re-weighted least squares (IRLS) to estimate models.
- M-estimation defines a weight function which is applied during estimation.
- The weights depend on the residuals and the residuals depend on the weights, so an iterative process is required.

We'll fit the model, using the default weighting choice: what are called Huber weights, where observations with small residuals get a weight of 1, and the larger the residual, the smaller the weight.

Our robust model (using `MASS::rlm`)

```
rob.huber <- rlm(crime ~ pov_c + single_c, data = crimestat)
```

Summary of the robust (Huber weights) model

```r
tidy(rob.huber)
```

```
# A tibble: 3 x 4
  term        estimate std.error statistic
  <chr>          <dbl>     <dbl>     <dbl>
1 (Intercept)    344.      13.1      26.2
2 pov_c           11.9      5.51      2.16
3 single_c        31.0     10.5       2.94
```

Now, *both* predictors appear to have estimates that exceed twice their standard error. So this is a very different result than ordinary least squares gave us.

Glance at the robust model (vs. OLS)

```r
glance(mod1)
```

```
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic p.value    df
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <int>
1     0.197         0.163  164.      5.88 0.00518     3
# ... with 5 more variables: logLik <dbl>, AIC <dbl>,
#   BIC <dbl>, deviance <dbl>, df.residual <int>
```

```r
glance(rob.huber)
```

```
# A tibble: 1 x 6
  sigma converged logLik   AIC   BIC deviance
  <dbl> <lgl>      <dbl> <dbl> <dbl>    <dbl>
1  59.1 TRUE       -331.  671.  678. 1314784.
```

Understanding the Huber weights a bit

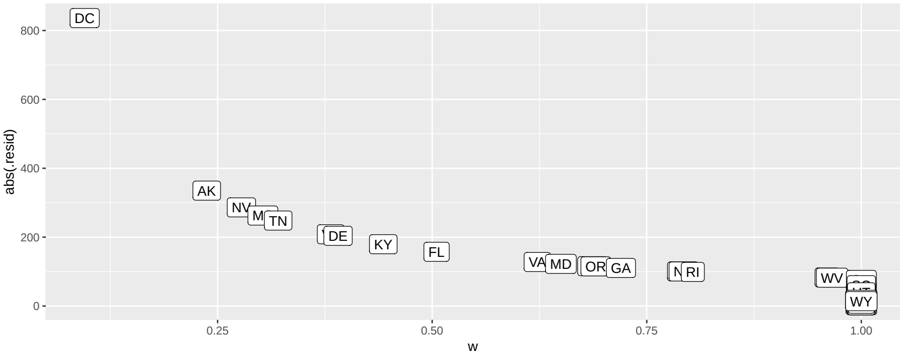
Let's augment the data with results from this model, including the weights used.

```
crime_with_huber <- augment(rob.huber, crimestat) %>%
    mutate(w = rob.huber$w) %>% arrange(w) %>% tbl_df

head(crime_with_huber, 3)
```

```
# A tibble: 3 x 15
    sid state crime poverty single trump state.full  pov_c
  <int> <fct> <dbl>   <dbl>  <dbl> <int> <fct>       <dbl>
1     9 DC    1244.    18.4   8.41     0 District ~  3.53
2     2 AK     636.    11.4   7.63     1 Alaska     -3.47
3    29 NV     636.    15.4   7.66     0 Nevada      0.527
# ... with 7 more variables: single_c <dbl>, .fitted <dbl>,
#   .se.fit <dbl>, .resid <dbl>, .hat <dbl>, .sigma <dbl>,
#   w <dbl>
```

Are cases with large residuals down-weighted?

```
ggplot(crime_with_huber, aes(x = w, y = abs(.resid))) +
    geom_label(aes(label = state))
```



Conclusions from the Plot of Weights

- The district of Columbia will be down-weighted the most, followed by Alaska and then Nevada and Mississippi.
- But many of the observations will have a weight of 1.
- In ordinary least squares, all observations would have weight 1.
- So the more cases in the robust regression that have a weight close to one, the closer the results of the OLS and robust procedures will be.

summary(rob.huber)

```

Call: rlm(formula = crime ~ pov_c + single_c, data = crimestat
Residuals:
     Min       1Q   Median       3Q      Max 
-262.751  -45.641    1.762   36.732  836.244 

Coefficients:
            Value    Std. Error t value
(Intercept) 343.7982  13.1309    26.1823
pov_c        11.9098   5.5058     2.1631
single_c     30.9868  10.5266     2.9437

Residual standard error: 59.14 on 48 degrees of freedom
```

Robust Linear Regression with the bisquare weighting function

Robust Linear Regression with the biweight

As mentioned there are several possible weighting functions - we'll next try the biweight, also called the bisquare or Tukey's bisquare, in which all cases with a non-zero residual get down-weighted at least a little. Here is the resulting fit...

```
(rob.biweight <- rlm(crime ~ pov_c + single_c,
                     data = crimestat, psi = psi.bisquare))
```

```
Call:
rlm(formula = crime ~ pov_c + single_c, data = crimestat, psi
Converged in 13 iterations

Coefficients:
(Intercept)       pov_c    single_c
  336.17015    10.31578    34.70765

Degrees of freedom: 51 total; 48 residual
Scale estimate: 67.3
```

Coefficients and Standard Errors

```
tidy(rob.biweight)
```

```
# A tibble: 3 x 4
  term        estimate std.error statistic
  <chr>          <dbl>     <dbl>     <dbl>
1 (Intercept)    336.      12.7      26.5 
2 pov_c           10.3      5.31      1.94
3 single_c        34.7     10.2       3.42
```

Understanding the biweights weights a bit

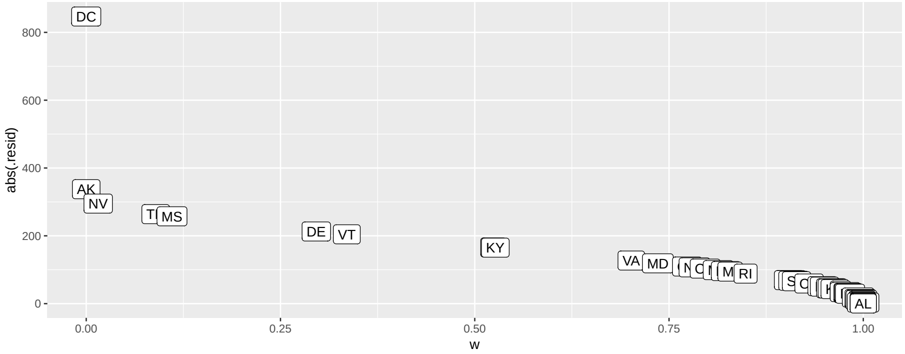
Let's augment the data, as above

```
crime_with_biweights <- augment(rob.biweight, crimestat) %>%
    mutate(w = rob.biweight$w) %>% arrange(w) %>% tbl_df

head(crime_with_biweights, 3)
```

```
# A tibble: 3 x 13
    sid state crime poverty single trump state.full  pov_c
  <int> <fct> <dbl>   <dbl>  <dbl> <int> <fct>       <dbl>
1     2 AK     636.    11.4   7.63     1 Alaska     -3.47
2     9 DC    1244.    18.4   8.41     0 District ~  3.53
3    29 NV     636.    15.4   7.66     0 Nevada      0.527
# ... with 5 more variables: single_c <dbl>, .fitted <dbl>,
#   .se.fit <dbl>, .resid <dbl>, w <dbl>
```

Relationship of Weights and Residuals

```
ggplot(crime_with_biweights, aes(x = w, y = abs(.resid))) +
    geom_label(aes(label = state))
```



Conclusions from the biweights plot

Again, cases with large residuals (in absolute value) are down-weighted generally, but here, Alaska and Washington DC receive no weight at all in fitting the final model.

- We can see that the weight given to DC and Alaska is dramatically lower (in fact it is zero) using the bisquare weighting function than the Huber weighting function and the parameter estimates from these two different weighting methods differ.
- The maximum weight (here, for Alabama) for any state using the biweight is still slightly smaller than 1.

summary(rob.biweight)

```
Call: rlm(formula = crime ~ pov_c + single_c, data = crimestat
Residuals:
    Min      1Q  Median      3Q     Max 
-257.58  -40.53    8.01   45.30  846.81 

Coefficients:
            Value    Std. Error t value
(Intercept) 336.1702  12.6733    26.5259
pov_c        10.3158   5.3139     1.9413
single_c     34.7077  10.1598     3.4162

Residual standard error: 67.27 on 48 degrees of freedom
```

Comparing OLS and the two weighting schemes

```r
glance(mod1) # OLS
```

```
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic p.value    df
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <int>
1     0.197         0.163  164.      5.88 0.00518     3
# ... with 5 more variables: logLik <dbl>, AIC <dbl>,
#   BIC <dbl>, deviance <dbl>, df.residual <int>
```

```r
glance(rob.biweight) # biweights
```

```
# A tibble: 1 x 6
  sigma converged logLik   AIC   BIC deviance
  <dbl> <lgl>      <dbl> <dbl> <dbl>    <dbl>
1  67.3 TRUE       -332.  672.  679. 1339850.
```

```r
glance(rob.huber) # Huber weights
```

```
# A tibble: 1 x 6
```

Bounded-Influence Regression

Bounded-Influence Regression and Least-Trimmed Squares

Under certain circumstances, M-estimators can be vulnerable to high-leverage observations, and so, bounded-influence estimators, like least-trimmed squares (LTS) regression have been proposed. The biweight that we have discussed is often fitted as part of what is called an MM-estimation procedure, by using an LTS estimate as a starting point.

The `ltsReg` function, which is part of the `robustbase` package (Note: **not** the `ltsreg` function from `MASS`) is what I use below to fit a least-trimmed squares model. The LTS approach minimizes the sum of the h smallest squared residuals, where h is greater than $n/2$, and by default is taken to be $(n + p + 1)/2$.

Least Trimmed Squares Model

```
lts1 <- ltsReg(crime ~ pov_c + single_c, data = crimestat)
```

Summarizing the LTS model

```
summary(lts1)$coeff
```

```
           Estimate Std. Error   t value     Pr(>|t|)
Intercept 339.14817  11.616766 29.194715 1.601245e-29
pov_c      16.99322   4.973459  3.416781 1.418337e-03
single_c   24.99819   9.136683  2.736024 9.073473e-03
```

MM estimation

Specifying the argument `method="MM"` to `rlm` requests bisquare estimates with start values determined by a preliminary bounded-influence regression, as follows. . .

```
rob.MM <- rlm(crime ~ pov_c + single_c,
              data = crimestat, method = "MM")

glance(rob.MM)
```

```
# A tibble: 1 x 6
  sigma converged logLik   AIC   BIC deviance
  <dbl> <lgl>      <dbl> <dbl> <dbl>    <dbl>
1  75.8 TRUE       -332.  672.  679. 1337077.
```

summary(rob.MM)

```
Call: rlm(formula = crime ~ pov_c + single_c, data = crimestat
Residuals:
     Min       1Q   Median       3Q      Max
-252.412  -41.096    8.696   47.141  847.128

Coefficients:
            Value    Std. Error t value
(Intercept) 336.3928  13.1929    25.4980
pov_c        10.5579   5.5318     1.9086
single_c     32.7755  10.5763     3.0990

Residual standard error: 75.79 on 48 degrees of freedom
```

Penalized Least Squares

Penalized Least Squares with `rms`

We can apply a penalty to least squares directly through the `ols` function in the `rms` package.

```
d <- datadist(crimestat)
options(datadist = "d")
pls <- ols(crime ~ pov_c + single_c, penalty = 1,
            data = crimestat, x=T, y = T)
```

The pls fit

```
Linear Regression Model

 ols(formula = crime ~ pov_c + single_c, data = crimestat, x =
     y = T, penalty = 1)

                      Model Likelihood      Discrimination
                         Ratio Test            Indexes
 Obs          51     LR chi2       11.18    R2       0.197
 sigma159.1209     d.f.       1.946198    R2 adj   0.164
 d.f.  48.0538     Pr(> chi2)   0.0035    g       89.298

 Residuals

     Min       1Q   Median       3Q      Max
 -284.24  -65.93  -16.68    15.66   807.01


              Coef      S.E.      t      Pr(>|t|)
```

How to Choose the Penalty in Penalized Least Squares?

The problem here is how to choose the penalty - and that's a subject I'll essentially skip today. The most common approach (that we've seen with the lasso) is cross-validation.

Meanwhile, what do we conclude about the fit here from AIC and BIC?

```
AIC(pls); BIC(pls)
```

```
    d.f.
669.5781
```

```
    d.f.
677.2014
```

Quantile Regression (on the Median)

Quantile Regression on the Median

We can use the rq function in the quantreg package to model the **median** of our outcome (violent crime rate) on the basis of our predictors, rather than the mean, as is the case in ordinary least squares.

```
rob.quan <- rq(crime ~ pov_c + single_c, data = crimestat)

glance(rob.quan)
```

```
# A tibble: 1 x 5
    tau logLik   AIC   BIC df.residual
  <dbl>  <dbl> <dbl> <dbl>       <int>
1   0.5  -316.  638.  643.          48
```

summary(rob.quan)

```
Call: rq(formula = crime ~ pov_c + single_c, data = crimestat

tau: [1] 0.5

Coefficients:
            coefficients lower bd  upper bd
(Intercept) 344.75658    336.94534 366.23603
pov_c        10.54757      3.06714  28.95962
single_c     32.27249      4.45889  48.18925
```

Estimating a different quantile (tau = 0.70)

In fact, if we like, we can estimate any quantile by specifying the `tau` parameter (here `tau` = 0.5, by default, so we estimate the median.)

```
(rob.quan70 <- rq(crime ~ pov_c + single_c, tau = 0.70,
                  data = crimestat))
```

```
Call:
rq(formula = crime ~ pov_c + single_c, tau = 0.7, data = crime

Coefficients:
(Intercept)       pov_c    single_c
  379.72818    19.30376    32.15827

Degrees of freedom: 51 total; 48 residual
```

Conclusions

Comparing Five of the Models

Estimating the Mean

Fit	Intercept CI	`pov_c` CI	`single_c` CI
OLS	(318.6, 410.2)	(-3.13, 35.35)	(-12.92, 60.60)
Robust (Huber)	(320.0, 367.6)	(0.89, 22.93)	(9.93, 52.05)
Robust (biweight)	(310.7, 361.5)	(-0.30, 20.94)	(14.39, 55.03)
Robust (MM)	(310.0, 362.8)	(-0.50, 21.62)	(11.62, 53.94)

Note: CIs estimated for OLS and Robust methods as point estimate $\pm$ 2 standard errors

Estimating the Median

Fit	Intercept CI	`pov_c` CI	`single_c` CI
Quantile (Median) Reg	(336.9, 366.2)	(3.07, 28.96)	(4.46, 48,19)

Comparing AIC and BIC

Fit	AIC	BIC
OLS	669.7	677.4
Robust (Huber)	670.8	678.5
Robust (biweight)	671.7	679.4
Robust (MM)	671.6	679.3
Quantile (median)	637.5	643.3

Some General Thoughts

1. When comparing the results of a regular OLS regression and a robust regression for a data set which displays outliers, if the results are very different, you will most likely want to use the results from the robust regression.
    - Large differences suggest that the model parameters are being highly influenced by outliers.
2. Different weighting functions have advantages and drawbacks.
    - Huber weights can have difficulties with really severe outliers.
    - Bisquare weights can have difficulties converging or may yield multiple solutions.
    - Quantile regression approaches have some nice properties, but describe medians (or other quantiles) rather than means.