Tutorial 10: Diagnostics, IV, Matching, Heckman

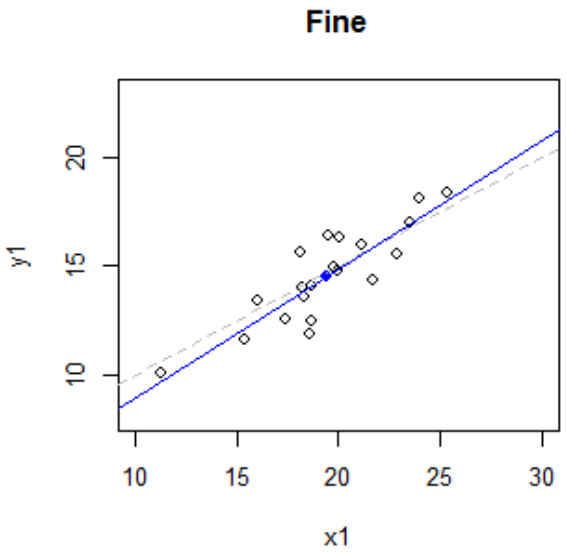
Anh Le

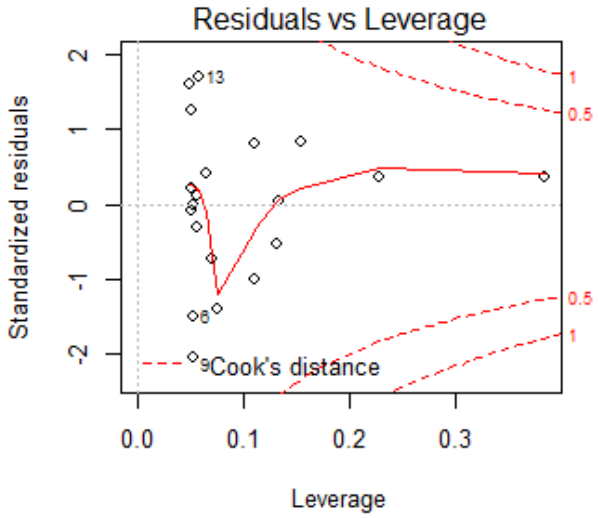
October 28, 2015

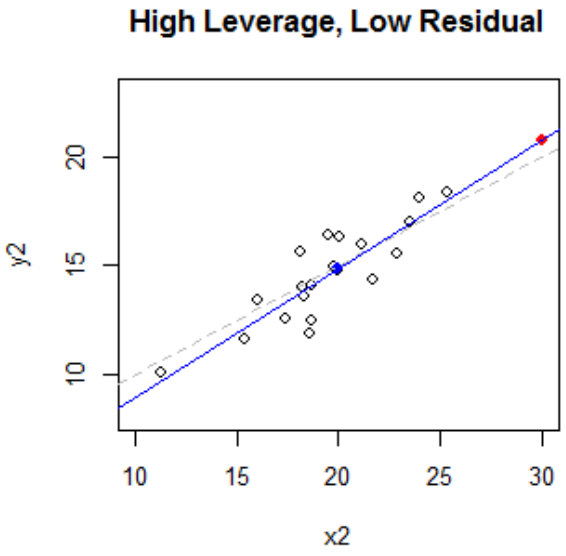
2. Diagnostics (DFbeta, partial regression plot, plot lm)
3. 2SLS (by package and by hand)
4. Heckman
5. Matching

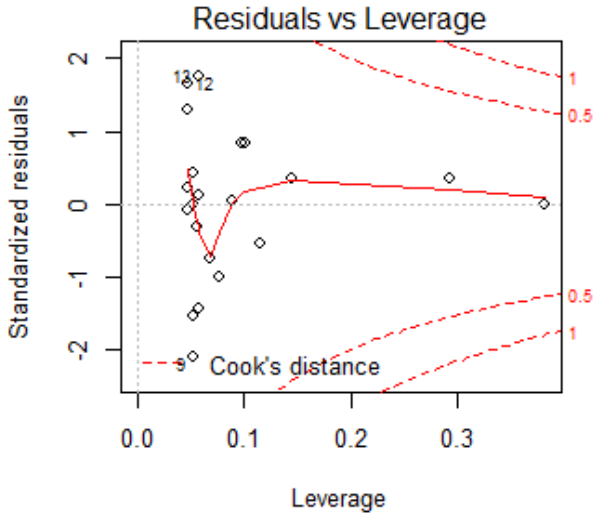
Outlier Diagnostics (graphical)

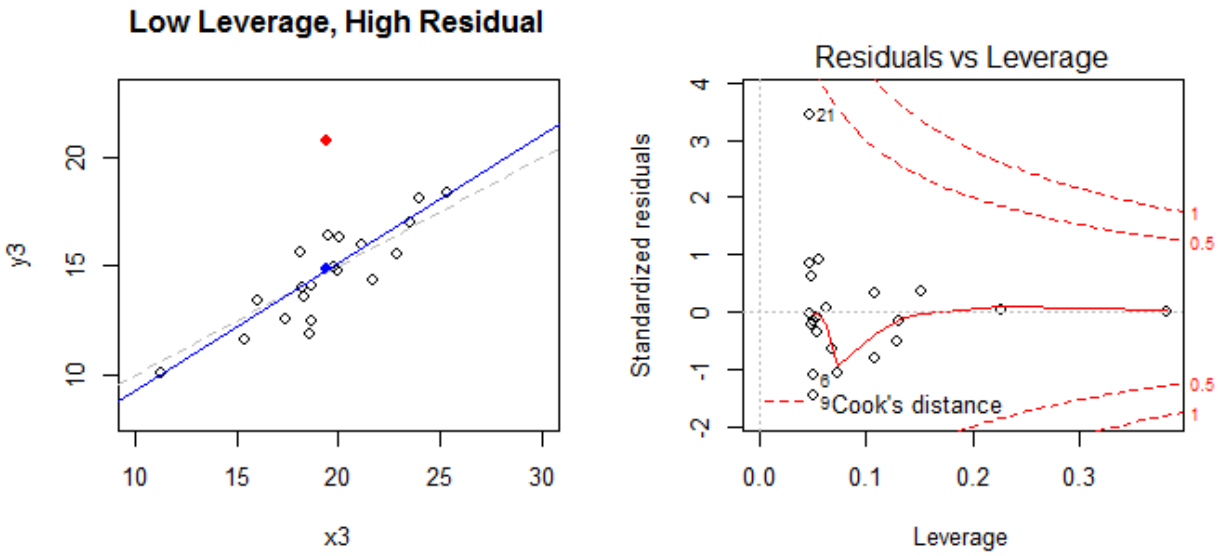
The following plots distinguish the concepts of `leverage` and `residual` (aka `discrepancy` as in the class slides). It also explains the last plot in `plot.lm()` that you haven't learned yet (i.e. the `Residuals vs Leverage` plot, aka Cook's Distance plot)

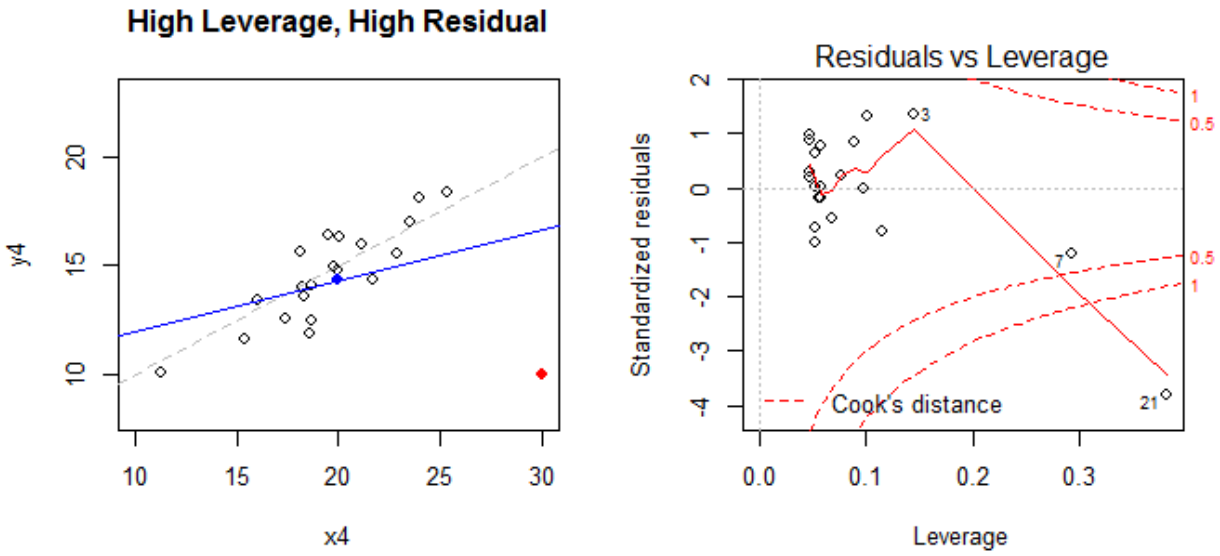

Fine
y1
x1
10
15
20
25
30


Residuals vs Leverage
Standardized residuals
Leverage
Cook's distance
13
6
9
0.0
0.1
0.2
0.3
-2
-1
0
1
2
0.5
1


High Leverage, Low Residual
y2
x2
10
15
20
25
30


Residuals vs Leverage
Standardized residuals
Leverage
Cook's distance
13
12
9
0.0
0.1
0.2
0.3
-2
-1
0
1
2
0.5
1





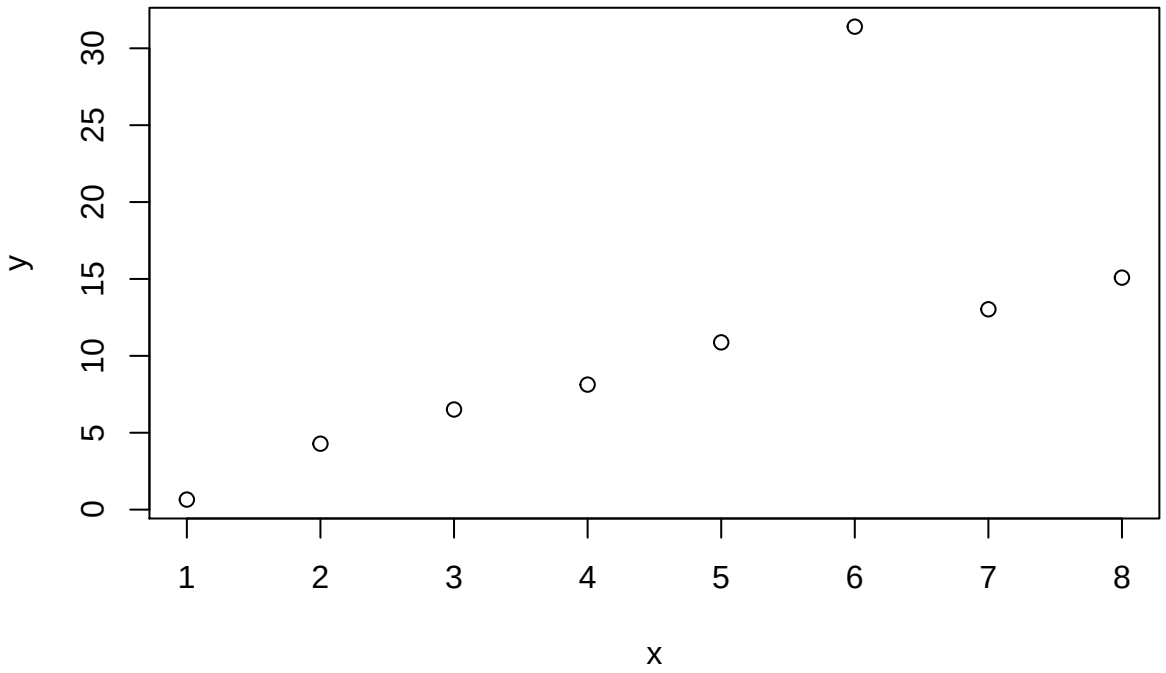
Credit: http://stats.stackexchange.com/questions/58141/interpreting-plot-lm

Outlier Diagnostics (DFBeta & Partial Regression Plot)

DFBeta

Let's create a mock dataset with the 6th observation being the outlier

```
x <- seq(1, 8)
y <- 2 * x + rnorm(8) # y is a linear function of x with added noise
y[6] <- 2 * x[6] + 20 + rnorm(1) # create the outlier
plot(x, y)
```



Here's how we detect the outlier:

```
m <- lm(y ~ x)
influence.measures(m)
```

```
## Influence measures of
##   lm(formula = y ~ x) :
##
##    dfb.1_   dfb.x   dffit    cov.r   cook.d   hat inf
## 1 -0.1837  0.1546 -0.1848 2.422016 0.020303 0.417   *
## 2 -0.0301  0.0230 -0.0312 1.980903 0.000584 0.274
## 3 -0.0427  0.0270 -0.0492 1.745254 0.001451 0.179
## 4 -0.0592  0.0199 -0.0935 1.619231 0.005180 0.131
## 5 -0.0223 -0.0188 -0.0881 1.623359 0.004611 0.131
## 6 -0.9750  4.9234  8.9889 0.000309 0.643517 0.179   *
## 7  0.1598 -0.3363 -0.4561 1.608377 0.112422 0.274
## 8  0.4032 -0.6787 -0.8112 1.760238 0.333367 0.417
```

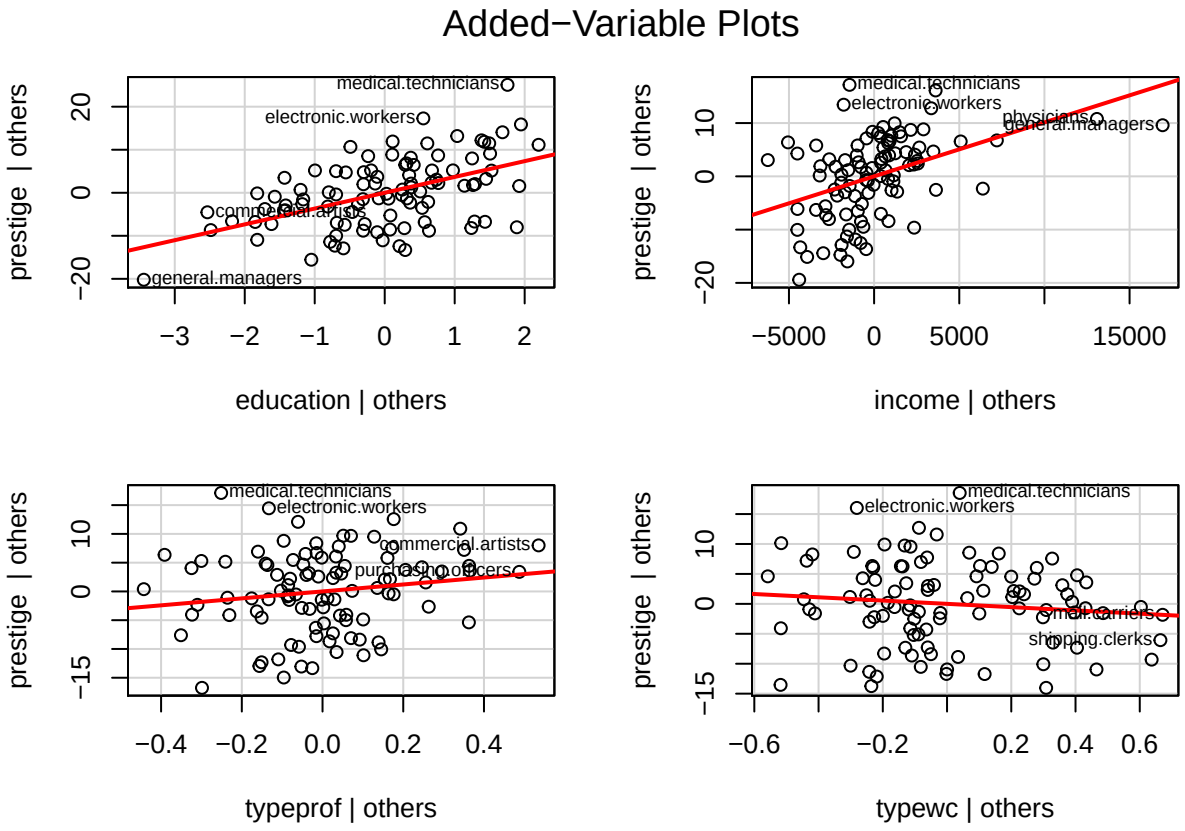
```
# dfbetas is scaled. It's = dfbeta / SE(beta), same as in class slide
dfbetas(m)
```

```
##   (Intercept)           x
## 1 -0.18373797  0.15463855
## 2 -0.03007757  0.02301277
## 3 -0.04271360  0.02696164
## 4 -0.05918244  0.01992378
## 5 -0.02232013 -0.01878519
## 6 -0.97498833  4.92344972
## 7  0.15981348 -0.33625771
## 8  0.40318532 -0.67866202
```

Partial correlation plot

```
library(car) # install if necessary

# Data set on the prestige of an occupation
reg1 <- lm(prestige ~ education + income + type, data = Prestige)
avPlots(reg1, id.n=2, id.cex=0.7)
```




```
# id.n - id most influential observation
# id.cex - font size for id.
```

2SLS

Y is outcome, X is endogenous, O is the omitted variable (importantly, X and O are correlated). And Z is the instrument for X.

Example: Y is `grade`, X is `attendance`, O is `time spent studying`. X and O are often correlated – hard-working students attend class more and also spend more time studying. Z is instrument for `attendance` (What could Z be?)

```
library(AER) # for ivreg
library(mvtnorm) # to generate multivariate normal

data <- rmvnorm(100, mean = c(0, 0, 0, 0),
```

```
                sigma = matrix(c(1, 0.5, 0.5, 0.5,
                                 0.5, 1, 0.5, 0,
                                 0.5, 0.5, 1, 0,
                                 0.5, 0, 0, 1), ncol = 4))
X <- data[, 1]
Z1 <- data[, 2]
Z2 <- data[, 3]
O <- data[, 4]

# Notice the correlation structure of the data
cor(data)
```

```
##           [,1]       [,2]        [,3]        [,4]
## [1,] 1.0000000  0.4119118  0.48647353  0.51321417
## [2,] 0.4119118  1.0000000  0.52038461 -0.12151744
## [3,] 0.4864735  0.5203846  1.00000000 -0.07420623
## [4,] 0.5132142 -0.1215174 -0.07420623  1.00000000
```

```
# Generate Y
Y <- 2 * X + 3 * O + rnorm(100)

# Run normal regression
summary(lm(Y ~ X))
```

```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3004 -1.8211 -0.4605  1.8321  8.0155
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06841    0.27381   -0.25    0.803
## X            3.40100    0.27290   12.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.738 on 98 degrees of freedom
## Multiple R-squared:  0.6131, Adjusted R-squared:  0.6092
## F-statistic: 155.3 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
# Run IV regression
# recall that true value of beta X is 2
summary(ivreg(Y ~ X | Z1 + Z2),
        diagnostics = TRUE)
```

```
##
## Call:
## ivreg(formula = Y ~ X | Z1 + Z2)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0758 -2.4139 -0.1881  1.6793  8.2547
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06094    0.33737  -0.181   0.8570
## X            1.45637    0.64565   2.256   0.0263 *
##
## Diagnostic tests:
##                  df1 df2 statistic  p-value
## Weak instruments   2  97    18.050 2.17e-07 ***
## Wu-Hausman         1  97    23.173 5.44e-06 ***
## Sargan             1  NA     0.209    0.647
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.374 on 98 degrees of freedom
## Multiple R-Squared: 0.4127,  Adjusted R-squared: 0.4067
## Wald test: 5.088 on 1 and 98 DF,  p-value: 0.02631
```

```
# Weak instrument: F-test. Null hypothesis = instruments are not correlated with X
# Wu-Hausman: check the endogeneity of X. Null hypothesis = X is not endogenous
# Sargan test: over-identification test, only runnable when there are more instruments than endogenous
```

```
# Run IV regression by hand
m_1ststage <- lm(X ~ Z1 + Z2)
xhat <- predict(m_1ststage)
m_2ndstage <- lm(Y ~ xhat)
summary(m_2ndstage)
```

```
##
## Call:
## lm(formula = Y ~ xhat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8715 -3.1138 -0.3046  2.8339 10.2979
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06094    0.43346  -0.141   0.8885
## xhat         1.45637    0.82953   1.756   0.0823 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.334 on 98 degrees of freedom
## Multiple R-squared:  0.03049,    Adjusted R-squared:  0.0206
## F-statistic: 3.082 on 1 and 98 DF,  p-value: 0.08227
```

Notice that running IV using package and by hand give the same coefficient estimate, but different standard error. It's because if we run the second stage like above, we don't take into account the uncertainty in estimating $\hat{x}$ (Notice how we just plug in $\hat{x}$, paying no attention to the standard error in the first stage.)

So, in real research, just use a package.

Heckman

$$y^o = y \times s \tag{1}$$
$$y = x + \epsilon_1 \tag{2}$$
$$s = 1 \text{ if } (x^s + \epsilon_2) > 0 \tag{3}$$
$$= 0 \text{ if } (x^s + \epsilon_2) < 0 \tag{4}$$

```
library(sampleSelection)

set.seed(0)
library(mvtnorm)

# Generate 2 epsilons so that they are correlated
eps <- rmvnorm(500, c(0, 0), matrix(c(1, -0.7, -0.7, 1), 2, 2))

x <- runif(500) # explanatory var for the outcome
y <- x + eps[, 1] # latent outcome variable

xs <- runif(500) # explanatory var for the selection
s <- (xs + eps[, 2]) > 0 # selection variable

yo <- y * (s > 0) # observable outcome, which has a bunch of 0 due to truncation

# Note that xs and x are independent, satisfying the exclusion restriction
```

Regular regression

```
summary(lm(yo ~ x))
```

```
##
## Call:
## lm(formula = yo ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.13942 -0.39944  0.00288  0.32733  2.28314
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.32711    0.06312  -5.182 3.19e-07 ***
## x            0.80920    0.11092   7.295 1.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7266 on 498 degrees of freedom
## Multiple R-squared:  0.09656,    Adjusted R-squared:  0.09474
## F-statistic: 53.22 on 1 and 498 DF,  p-value: 1.181e-12
```

Here's how you run a Heckman model

```r
summary(heckit(s ~ xs + x, yo ~ x))
```

```
## --------------------------------------------
## Tobit 2 model (sample selection model)
## 2-step Heckman / heckit estimation
## 500 observations (162 censored and 338 observed)
## 8 free parameters (df = 493)
## Probit selection equation:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.13947    0.15332   0.910 0.363452
## xs           0.68262    0.20488   3.332 0.000927 ***
## x           -0.01944    0.20000  -0.097 0.922623
## Outcome equation:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1870     0.3358   0.557    0.578
## x             1.2460     0.2096   5.946 5.21e-09 ***
## Multiple R-Squared:0.1647,   Adjusted R-Squared:0.1597
## Error terms:
##               Estimate Std. Error t value Pr(>|t|)
## invMillsRatio  -1.3259     0.6123  -2.166   0.0308 *
## sigma           1.2744         NA      NA       NA
## rho            -1.0404         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## --------------------------------------------
```

Matching

We can use the library `MatchIt`, which can run a lot of matching method.

```r
library(MatchIt)

data(lalonde)
m.out <- matchit(treat ~ educ + black + hispan, data = lalonde,
                 method = "cem")
```

```
##
## Using 'treat'='1' as baseline group
```

```r
summary(m.out) # to check balance
```

```
##
## Call:
## matchit(formula = treat ~ educ + black + hispan, data = lalonde,
##     method = "cem")
##
## Summary of balance for all data:
##          Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean
## distance        0.5576        0.1908     0.2298    0.3668  0.5197   0.3678
## educ           10.3459       10.2354     2.8552    0.1105  1.0000   0.7027
```

```
## black           0.8432        0.2028     0.4026    0.6404  1.0000   0.6432
## hispan          0.0595        0.1422     0.3497   -0.0827  0.0000   0.0811
##          eQQ Max
## distance  0.5845
## educ      4.0000
## black     1.0000
## hispan    1.0000
##
##
## Summary of balance for matched data:
##          Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean
## distance        0.5576        0.5587     0.2015   -0.0011  0.5074   0.3576
## educ           10.3459       10.4379     2.0532   -0.0920  0.0000   0.2919
## black           0.8432        0.8432     0.3641    0.0000  1.0000   0.6216
## hispan          0.0595        0.0595     0.2368    0.0000  0.0000   0.0541
##          eQQ Max
## distance  0.5879
## educ      2.0000
## black     1.0000
## hispan    1.0000
##
## Percent Balance Improvement:
##          Mean Diff.  eQQ Med eQQ Mean eQQ Max
## distance    99.6877   2.3629   2.7693 -0.5729
## educ        16.7970 100.0000  58.4615 50.0000
## black      100.0000   0.0000   3.3613  0.0000
## hispan     100.0000   0.0000  33.3333  0.0000
##
## Sample sizes:
##           Control Treated
## All           429     185
## Matched       380     185
## Unmatched      49       0
## Discarded       0       0
```

```
# Get the matched data
lalonde_matched <- match.data(m.out)
names(lalonde_matched) # there's a weight variable in here
```

```
##  [1] "treat"    "age"      "educ"     "black"    "hispan"   "married"
##  [7] "nodegree" "re74"     "re75"     "re78"     "distance" "weights"
## [13] "subclass"
```

```
# Run weighted regression to get the causal treatment effect
lm(re78 ~ treat + age + educ + black + hispan,
   data = lalonde_matched, weights = lalonde_matched$weights)
```

```
##
## Call:
## lm(formula = re78 ~ treat + age + educ + black + hispan, data = lalonde_matched,
##     weights = lalonde_matched$weights)
##
## Coefficients:
```

```
## (Intercept)        treat          age         educ        black
##     -971.89       951.25        63.09       610.36     -1904.55
##      hispan
##      543.53
```