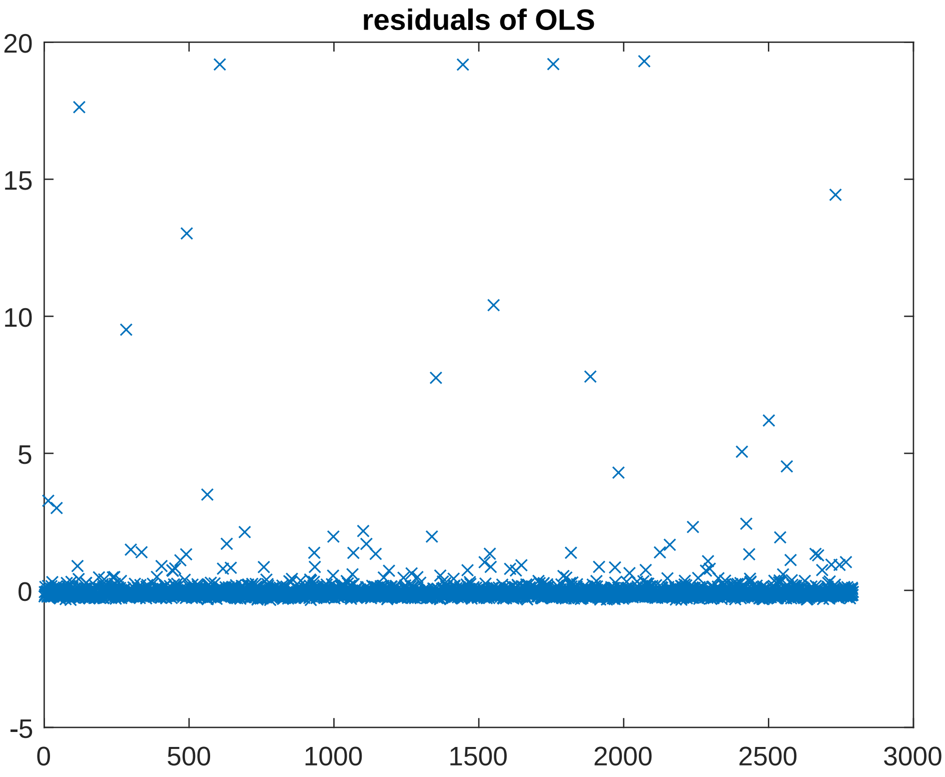
Relationship between alcohol and health

Daxuan Deng

December 1, 2019

Our goal is to figure out the relationship between alcohol use and health condition. First, we use the regular way: running OLS.

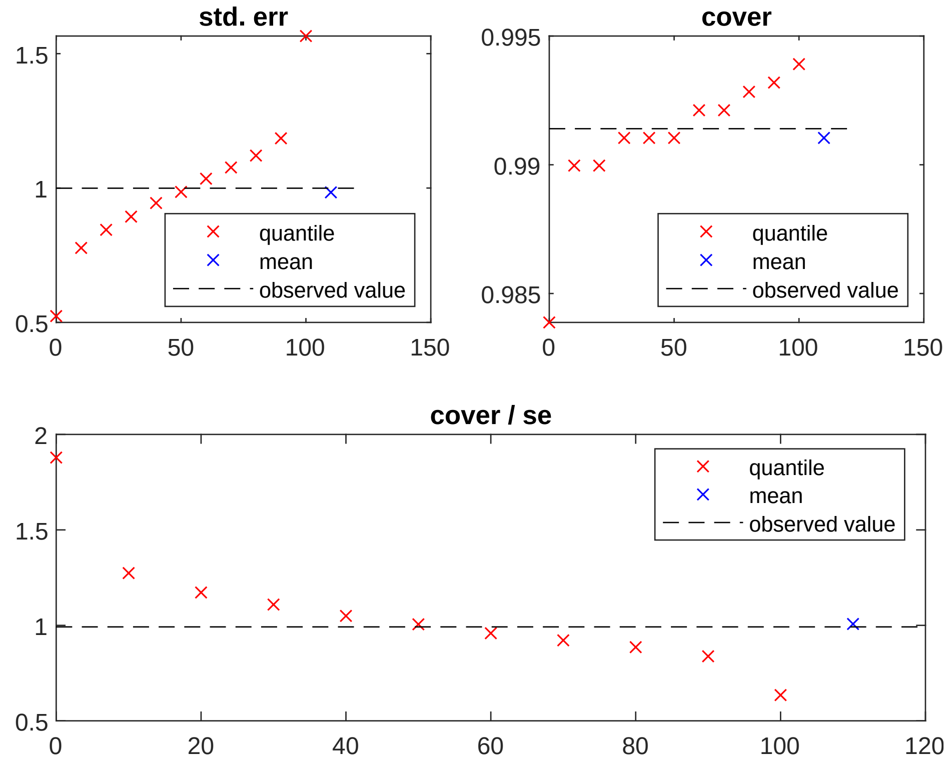
```
% Title:  Matlab script for group project, STATS 506
% Author: Daxuan Deng
% Funtion: explore the relationship between health condition and drinking
%          alcohol, using data NHANES 2005—2006 data.
% Date: 11/30/2019

% load data
alq = xptread('ALQ_D.XPT');
demo = xptread('DEMO_D.XPT');
hsq = xptread('HSQ_D.XPT');

% select variables
alq = alq(:, {'SEQN', 'ALQ120Q'});
demo = demo(:, {'SEQN', 'RIAGENDR', 'RIDAGEYR', 'DMDEDUC2', 'INDFMPIR'});
hsq = hsq(:, {'SEQN', 'HSD010'});

% merge data
dt = join(alq, demo, 'Keys', 'SEQN');
dt = join(dt, hsq, 'Keys', 'SEQN');

% rename columns
dt.Properties.VariableNames = ...
["id", "alcohol", "sex", "yr", "edu", "pir", "health"];

% drop invalid values
dt = rmmissing(dt);
dt(dt.alcohol > 365, :) = [];
dt(dt.yr < 21, :) = [];
dt(dt.edu > 5, :) = [];
dt(dt.health > 3, :) = [];

% centralize and factorize
dt.alcohol = (dt.alcohol - mean(dt.alcohol)) ./ std(dt.alcohol);
dt.sex = categorical(dt.sex);
dt.yr = (dt.yr - mean(dt.yr)) ./ std(dt.yr);
dt.edu = categorical(dt.edu);
dt.pir = (dt.pir - mean(dt.pir)) ./ std(dt.pir);
dt.health = categorical(dt.health);
dt.health = reordercats(dt.health, {’3’,’2’,’1’});

% run OLS
md = fitlm(dt, ’alcohol ˜ sex + yr + edu + pir + health’);

% extract fitted values and residuals
fit = predict(md, dt(:, 3:7));
res = md.Residuals.Raw;

% calculate observed coverage probability
se = sqrt(sum(res .ˆ2) / (height(dt) - width(dt) -1));
ci = [fit - norminv(0.975) * se, fit + norminv(0.975) * se];
cover = sum(dt.alcohol > ci(:,1) & dt.alcohol < ci(:,2)) / length(ci);

% plot residuals
plot(1:length(ci), res, ’x’), title(’residuals of OLS’)
```



We could learn from this plot that the residuals are skewed. As a result, the regular way to estimate SE may fail. So we use bootstrap method to resample residuals.

```
% bootstrap
rng(506);
nboot = 1000;

func_se = @(x)sqrt(sum(x .^2) / (height(dt) - width(dt) -1));
se_boot = bootstrp(nboot, func_se, res);
cover_boot = zeros(nboot, 1);

for i=1:nboot
    ci_boot = [fit - norminv(0.975) * se_boot(i), ...
        fit + norminv(0.975) * se_boot(i)];
    cover_boot(i) = sum(dt.alcohol > ci_boot(:,1) & ...
        dt.alcohol < ci_boot(:,2)) / length(ci);
end

% consider 0, 10, ... , 100 quantile, mean and the original observed value
% as estimate candidates
se_list = ([quantile(se_boot, (0:10) / 10), mean(se_boot), se])' ;
cover_list = zeros(13, 1);

for i=1:13
    ci_list = [fit - norminv(0.975) * se_list(i), ...
        fit + norminv(0.975) * se_list(i)];
    cover_list(i) = sum(dt.alcohol > ci_list(:,1) & ...
        dt.alcohol < ci_list(:,2)) / length(ci);
end

ratio_list = cover_list ./ se_list;

% plot se
subplot(2,2,1)
plot(linspace(0,100,11), se_list(1:11), 'rx', ...
     110, se_list(12), 'bx', ...
     linspace(0,120,13), se_list(13) * ones(13,1), 'k--'), ...
     title('std. err'), ...
     legend('quantile', 'mean', 'observed value', 'Location','southeast')

% plot coverage (probability)
subplot(2,2,2)
plot(linspace(0,100,11), cover_list(1:11), 'rx', ...
     110, cover_list(12), 'bx', ...
     linspace(0,120,13), cover_list(13) * ones(13,1), 'k--'), ...
     title('cover'), ...
     legend('quantile', 'mean', 'observed value', 'Location','southeast')

% plot cover / se
subplot(2,2,[3,4])
```

```
plot(linspace(0,100,11), ratio_list(1:11), 'rx', ...
     110, ratio_list(12), 'bx', ...
     linspace(0,120,13), ratio_list(13) * ones(13,1), 'k--'), ...
     title('cover / se'), ...
     legend('quantile', 'mean', 'observed value', 'Location','northeast')
```



The plot indicates that the coverage probabilities are similar among all estimates, but the lengths of SE varies. If we define the ratio of coverage and SE as the efficiency of estimtaor, the bigger the better. Then we can conclude that the minimum of SE has the best performance.

Naturally, we want to ask whether the minimum reaches the highest efficiency by coincidence. To address this problem, we use Monte Carlo to see the frequency that the minimum is the best.

```
%test
nrep = 1000;

count = 0;

for i=1:nrep
    se_boot = bootstrp(nboot, func_se, res);
    cover_boot = zeros(nboot, 1);
    for j=1:nboot
        ci_test = [fit - norminv(0.975) * se_boot(j), ...
            fit + norminv(0.975) * se_boot(j)];
        cover_boot(j) = sum(dt.alcohol > ci_test(:,1) & ...
            dt.alcohol < ci_test(:,2)) / length(ci);
    end

    se_list = ([quantile(se_boot , (0:10) / 10), mean(se_boot), se])' ;
    cover_list = zeros(13, 1);

    for j=1:13
        ci_list = [fit - norminv(0.975) * se_list(j), ...
        fit + norminv(0.975) * se_list(j)];
        cover_list(j) = sum(dt.alcohol > ci_list(:,1) & ...
        dt.alcohol < ci_list(:,2)) / length(ci);
    end

    ratio_list = cover_list ./ se_list;

    if( max(ratio_list) == ratio_list(1) )
        count = count +1;
    end
end

prob = count / nrep;

prob
```

```
prob =

     1
```

Indeed, the minimum is always the best. It is not surprising because of the distribution of residuals. It is obvious from the residual plot that, most points are concentrated around 0, but there are some outliers far away from 0. Each of them could make RSS bigger dramatically. So the best way to avoid it is to decrease the weight of outliers, which means choosing the minimum of SE in empirical distribution.