Exercise – Understanding the consistency of the OLS estimator using simulation

1. Aim of the exercise

The aim of this exercise is to understand the consistency of the OLS estimator using simulation.

2. Theory

The consistency of the OLS estimator means that as the sample size increases, the estimator converges to the true parameter value. In other words, with a sufficiently large sample, the OLS estimator provides an estimate that is close to the actual population parameter. This concept of consistency is underpinned by the notion of the probability limit – see the simulation exercise on the law of large numbers which illustrates the probability limit. Here we provide the proof for consistency. Consider the standard linear regression model

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$$

where $\boldsymbol{X}$ is matrix of variables. It is $n \times K$. The bold font indicates multiple observations. The big font indicates multiple variables. Replacing the model in the OLS estimator gives

$$\begin{aligned}
\hat{\boldsymbol{\beta}} &= (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y} \\
&= (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'(\boldsymbol{X\beta} + \boldsymbol{\varepsilon}) \\
&= (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X\beta} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{\varepsilon} \\
&= \boldsymbol{\beta} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{\varepsilon}. \\
&= \boldsymbol{\beta} + \left(\frac{1}{n}\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\frac{1}{n}\boldsymbol{X}'\boldsymbol{\varepsilon}.
\end{aligned}$$

The product of the transpose of the design matrix with itself can be expressed as a sum of outer products of individual data points. This representation is part of what's called matrix decomposition or quadratic forms in linear algebra.

$$\begin{aligned}
\boldsymbol{X}'\boldsymbol{X} &= \begin{bmatrix} 1 & 1 & \dots & 1 & \dots & 1 \\ x_{21} & x_{22} & \dots & x_{2i} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{j1} & x_{j2} & \dots & x_{ji} & \dots & x_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{k1} & x_{k2} & \dots & x_{ki} & \dots & x_{kn} \end{bmatrix} \begin{bmatrix} 1 & x_{21} & \dots & x_{j1} & \dots & x_{k1} \\ 1 & x_{22} & \dots & x_{j2} & \dots & x_{k2} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_{2i} & \dots & x_{ji} & \dots & x_{ki} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_{2n} & \dots & x_{jn} & \dots & x_{kn} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \dots & \boldsymbol{x}_i & \dots & \boldsymbol{x}_n \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1' \\ \boldsymbol{x}_2' \\ \vdots \\ \boldsymbol{x}_i' \\ \vdots \\ \boldsymbol{x}_n' \end{bmatrix} \\
&= \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i'.
\end{aligned}$$

Similarly, we have

$$
\begin{aligned}
\boldsymbol{X}'\boldsymbol{\varepsilon} &= \begin{bmatrix} 1 & 1 & \dots & 1 & \dots & 1 \\ x_{21} & x_{22} & \dots & x_{2i} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{j1} & x_{j2} & \dots & x_{ji} & \dots & x_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{k1} & x_{k2} & \dots & x_{ki} & \dots & x_{kn} \end{bmatrix} \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_i \\ \vdots \\ \varepsilon_n \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \dots & \boldsymbol{x}_i & \dots & \boldsymbol{x}_n \end{bmatrix} \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_i \\ \vdots \\ \varepsilon_n \end{bmatrix} \\
&= \sum_{i=1}^{n} \boldsymbol{x}_i \varepsilon_i.
\end{aligned}
$$

Hence,

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \left(\frac{1}{n}\boldsymbol{X}'\boldsymbol{X}\right)^{-1} \frac{1}{n}\boldsymbol{X}'\boldsymbol{\varepsilon}$$

becomes

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\varepsilon_i.$$

We now take the plim of both sides of the equation. When taking the plim, we do not condition on $\boldsymbol{X}$. To derive asymptotic results, we do not need the technical simplification brought by fixing $\boldsymbol{X}$ in repeated samples. Using the sum rule of plim (Greene, Theorem D.14),

$$\text{plim } \hat{\boldsymbol{\beta}} = \text{plim } \boldsymbol{\beta} + \text{plim } \left[\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\varepsilon_i\right].$$

Using the product rule of plim (Greene, Theorem D.14),

$$\text{plim } \hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \text{plim } \left[\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i'\right)^{-1}\right] \text{plim } \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\varepsilon_i.$$

Assuming that x_i is i.i.d. (A5), and using the WLLN (Greene, Theorem D.5),

$$\text{plim} \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i' = \text{E}\left[\boldsymbol{x}_i\boldsymbol{x}_i'\right].$$

Using the ratio rule of plim (Greene, Theorem D.14), and assuming that $(\boldsymbol{X}'\boldsymbol{X})^{-1}$ exists,

$$\text{plim} \left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i'\right)^{-1} = \left(\text{E}\left[\boldsymbol{x}_i\boldsymbol{x}_i'\right]\right)^{-1}.$$

Assuming that x_i is i.i.d., and using the WLLN (Greene, Theorem D.5),

$$\operatorname{plim} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i' = \mathrm{E}\left[\boldsymbol{x}_i \boldsymbol{x}_i'\right].$$

Assuming that ε_i is i.i.d., assuming that $\mathrm{E}\left[\boldsymbol{x}_i \varepsilon_i\right] = \mathbf{0}$ (A3), and using the WLLN,

$$\operatorname{plim} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \varepsilon_i = \mathrm{E}\left[\boldsymbol{x}_i \varepsilon_i\right] = \mathbf{0}.$$

Hence, we have

$$\operatorname{plim} \hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \underbrace{\operatorname{plim} \left[\left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i'\right)^{-1}\right]}_{\left(\mathrm{E}\left[\boldsymbol{x}_i \boldsymbol{x}_i'\right]\right)^{-1}} \underbrace{\operatorname{plim} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \varepsilon_i}_{\mathrm{E}\left[\boldsymbol{x}_i \varepsilon_i\right] = \mathbf{0}}$$

Hence,

$$\operatorname{plim} \hat{\boldsymbol{\beta}} = \boldsymbol{\beta}.$$

3. Application

3.1. Clear the memory

Set the number of simulations to be carried out.

```
% 3.1. Clear the memory
clear;
```

3.2. Set the number of simulations

Set the number of simulations to be carried out.

```
% 3.2. Set the number of simulations
N_sim = 1000;
```

3.3. Set the sample size

Assume that we have a linear regression model that contains a constant term and an independent variable. Assume also that we have `N_obs` observations for the variables of this model.

```
% 3.3. Set the sample size
N_obs = [1000 10000 100000];
N_obs_j = size(N_obs,2);
```

3.4. Set true values for the coefficients of the intercept and the independent variable

Assume that we know the true values of the coefficients of the variables of the linear regression model we consider, and that these values are as indicated at the end of the section.

```
% 3.4. Set true values for the coefficients
B_true = [0.2 0.5]';
```

3.5. Define the number of coefficients to be simulated

Define the number of coeffcients to be simulated

```
% 3.5. Define the number of coefficients to be simulated
N_par = 1;
```

3.6. Preallocate matrices for storing the simulated OLS coefficient estimates

The code presented at the end of the section creates an empty vector, and an empty matrix. The empty vector is `N_sim` × `N_par` because the vector is to store `N_sim` coefficient estimates of the only independent variable `x_1` from a given simulation using a certain number of observations (sample size). The empty matrix is to store in each of its columns the `N_sim` simulated coefficient estimates from three different scenarios featuring different numbers of observations. Therefore the matrix is `N_sim` × `N_obs_j`, where `N_obs_j` is the number of different scenarios of numbers of observations.

```
% 3.6. Preallocate matrices for storing the simulated statistics
B_hat_1_sim = NaN(N_sim,N_par);
B_hat_1_sim_j = NaN(N_sim,N_obs_j);
```

3.7. Create sampling distributions for the OLS coefficient estimates using different sample sizes

The for loop presented at the end of the section creates three different sampling distributions for the OLS coefficient estimates using three different sample sizes. A note for avoiding a computational hurdle is the following. The presented for loop makes use of the user-written function `exercisefunctionlss`. The function calculates a set of OLS statistics. However, the for loop used here only needs the OLS coefficient estimates, and therefore other OLS statistics need not be calculated. To avoid an unnecessary waiting time for the for loop to finish its iterations, go to the function file, and mask the code except the part of it calculating the OLS coefficient estimates. You can mask the code by selecting the code, and then by pressing the 'Comment' button located on the toolbar of the Editor of your open script file.

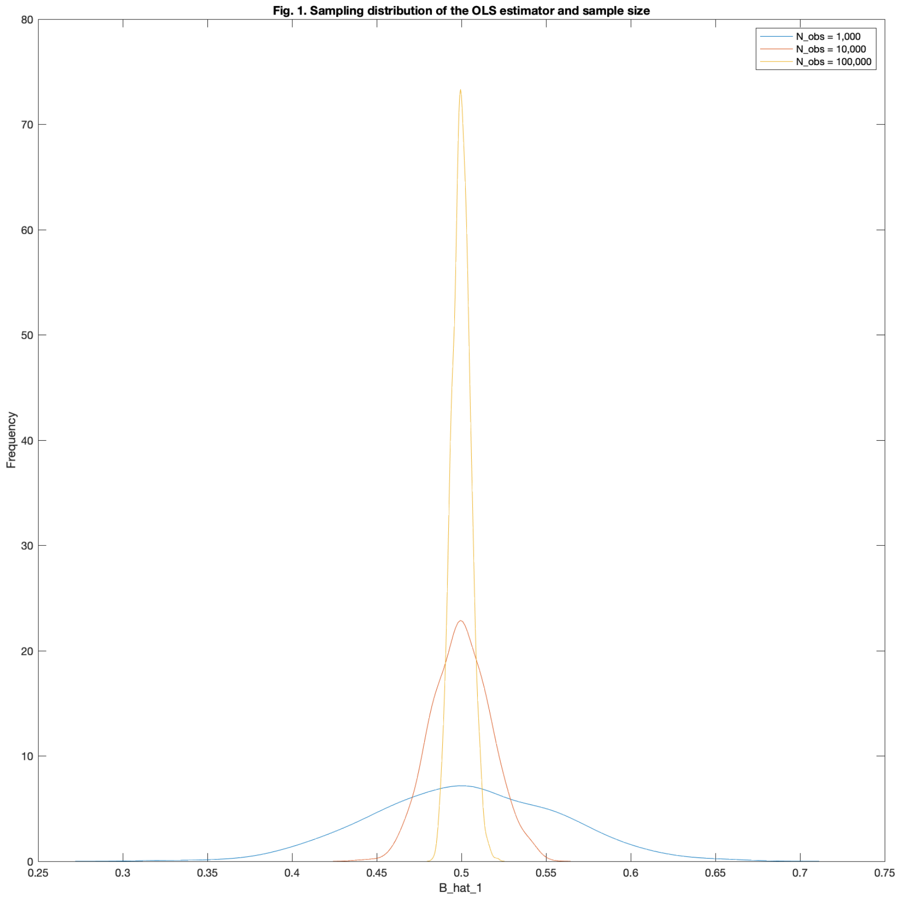
```
% 3.7. Create sampling distributions for the OLS estimator
for j = 1:N_obs_j
    for i = 1:N_sim
        u = random('Normal',0,1,[N_obs(1,j) 1]);
        x_0 = ones(N_obs(1,j),1);
        x_1 = random('Uniform',-1,1,[N_obs(1,j) 1]);
        X = [x_0 x_1];
        y = X*B_true+u;
        LSS = exercisefunctionlss(y,X);
        B_hat_1_sim(i,1) = LSS.B_hat(2,1);
        B_hat_1_sim_j(:,j) = B_hat_1_sim(:,1);
    end
```

```
end
```

4. Plot the sampling distributions of the OLS coefficient estimates at different sample sizes

Here we plot three different sampling distributions featuring three different sample sizes. It demonstrates the consistency property of the OLS estimator. Note that this exercise shows consistency of the estimator without doing any theoretical derivation such as taking the probability limit of the estimator. Indeed, when it is not possible to theoretically prove the consistency, or some other property, of an estimator, one can rely on a simulation study, as done here, to investigate the properties of the estimator of interest.

```
%% 4. Plot the sampling distributions of the OLS estimator
ksdensity(B_hat_1_sim_j(:,1))
hold on
ksdensity(B_hat_1_sim_j(:,2))
hold on
ksdensity(B_hat_1_sim_j(:,3))
title(['Fig. 1. Sampling distribution of the OLS estimator ' ...
    'and sample size'])
legend('N\_obs = 1,000','N\_obs = 10,000','N\_obs = 100,000')
ylabel('Frequency')
xlabel('B\_hat\_1')
```



5. Final notes

The proof of consistency makes use theorems presented in Greene, W. H., 2018. Econometric Analysis, 8th edition, Pearson Education Limited. This file is prepared and copyrighted by Tunga Kantarcı. This file and the accompanying MATLAB files are available on GitHub and can be accessed via this link.