Exercise – Understanding the method of paired bootstrap

1. Aim of the exercise

The aim of this exercise is to understand how to correctly estimate standard errors of regression coefficients when the error variance is not constant (i.e., heteroskedasticity is present). Standard inference methods can be misleading in such cases. We use the paired bootstrap method, which resamples observations as (y, X) pairs, to account for heteroskedasticity and compare its performance to sampling directly from the population.

2. Theory

Consider the linear regression model

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

where $\mathbf{y}$ is an $n \times 1$ vector of outcome, and $\mathbf{X}$ is an $n \times K$ matrix of regressors. The parameter vector $\boldsymbol{\beta}$ is $K \times 1$ and unknown. Assume the zero conditional mean assumption holds:

$$\mathrm{E}\left[\boldsymbol{\varepsilon} \mid \mathbf{X}\right] = 0,$$

but allow for heteroskedastic errors such that

$$\mathrm{E}\left[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}' \mid \mathbf{X}\right] = \boldsymbol{\Omega},$$

where $\boldsymbol{\Omega} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$. The ordinary least squares (OLS) estimator of $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}.$$

To account for heteroskedasticity, the variance-covariance matrix of $\hat{\boldsymbol{\beta}}$ can be estimated using the heteroskedasticity-consistent estimator (often called White's estimator):

$$\widehat{\mathrm{Var}}\left[\hat{\boldsymbol{\beta}}\right] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\hat{\boldsymbol{\Omega}}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1},$$

where $\hat{\boldsymbol{\Omega}} = \mathrm{diag}(\hat{u}_1^2, \hat{u}_2^2, \ldots, \hat{u}_n^2)$, and $\hat{u}_i$ are the OLS residuals.

While the heteroskedasticity-consistent estimator is asymptotically valid, it can suffer from finite-sample biases, particularly in the presence of high-leverage observations. In such cases, the paired bootstrap offers an alternative by resampling entire $(\mathbf{x}_i, y_i)$ observation pairs and approximating the finite-sample distribution of the OLS estimator without the need to specify or estimate the heteroskedastic error-covariance matrix $\boldsymbol{\Omega}$.

Here, a 'pair' refers to the combined observation $(y_i, \mathbf{x}_i)$, where y_i is the response variable and $\mathbf{x}_i$ is a vector of K covariates for the ith observation. Let $(y_i, \mathbf{x}_i)$, for $i = 1, \ldots, n$, be an i.i.d. sample from the linear model described above. To implement the paired bootstrap, we treat the dataset as a collection of observation pairs $\{(y_1, \mathbf{x}_1), (y_2, \mathbf{x}_2), \ldots, (y_n, \mathbf{x}_n)\}$. For each bootstrap replication $b = 1, \ldots, B$, we draw n pairs with replacement from this set to form a bootstrap sample $\{(y_1, \mathbf{x}_1)^b, (y_2, \mathbf{x}_2)^b, \ldots, (y_n, \mathbf{x}_n)^b\}$. Using this resampled dataset, we compute the OLS estimator

$$\hat{\boldsymbol{\beta}}^b = \left(\mathbf{X}^{b\prime}\mathbf{X}^b\right)^{-1}\mathbf{X}^{b\prime}\mathbf{y}^b$$

and store the result. After B replications, we obtain the collection $\{\hat{\boldsymbol{\beta}}^b\}_{b=1}^B$. The empirical distribution of these bootstrap estimates can then be used to assess key statistical properties of the estimator, including its standard error, bias, and to construct confidence intervals.

As discussed in the exercise on bootstrap theory, under suitable regularity conditions, (i) as the number of bootstrap samples increase:

$$\hat{G}_n(\tau, F_n) \xrightarrow{p} G_n(\tau, F_n)$$

and (ii) as the size of the initial sample increase:

$$G_n(\tau, F_n) \xrightarrow{p} G_n(\tau, F)$$

meaning that the distribution of a statistic under the empirical distribution F_n converges in probability to its true sampling distribution under the population distribution F. Equivalently, for the OLS estimator:

$$\sqrt{n}(\hat{\beta}^b - \hat{\beta}) \xrightarrow{d^*} \sqrt{n}(\hat{\beta} - \beta)$$

where d^* denotes convergence of distributions under the bootstrap measure. That is, the distribution of the OLS estimator computed from bootstrap resamples $\hat{\beta}^b$ converges to the sampling distribution of the original estimator $\hat{\beta}$. Moreover, the bootstrap estimates can be used to compute the multivariate sample variance-covariance matrix of the OLS estimator:

$$\widehat{\text{Var}}\left[\hat{\boldsymbol{\beta}}^b\right] = \frac{1}{B-1}\sum_{b=1}^{B}(\hat{\boldsymbol{\beta}}^b - \bar{\boldsymbol{\beta}})(\hat{\boldsymbol{\beta}}^b - \bar{\boldsymbol{\beta}})',$$

where

$$\bar{\boldsymbol{\beta}} = \frac{1}{B}\sum_{b=1}^{B}\hat{\boldsymbol{\beta}}^b.$$

This estimator reflects both the variances of the individual coefficients and their covariances, thereby providing a comprehensive characterization of the estimator's sampling uncertainty. The key is that this estimator does not rely on specific assumptions about the error structure like the standard variance estimator of the OLS estimator which is given by

$$\widehat{\text{Var}}\left[\hat{\boldsymbol{\beta}}\right] = s^2(X'X)^{-1}.$$

It also circumvents asymptotic approximations, which can perform poorly in finite samples or in the presence of complex forms of heteroskedasticity.

3. Set values for the parameters of the simulation

Clear all variables from memory and define the total number of simulations or samples to be generated.

```
%% 3. Set values for the parameters of the simulation

% 3.1. Clear the memory
clear;

% 3.2. Set the number of simulations as the number of bootstrap samples
N_sim = 5000;
```

4. Generate population data

We consider a simple linear regression model with one predictor, drawn from a uniform distribution. The error term is heteroskedastic, with its variance following an exponential pattern governed by a parameter we set to 1.5. This parameter controls the degree of heteroskedasticity: higher values amplify the variance pattern, allowing us to evaluate how effectively the paired bootstrap deals with non-constant error variance. In our setup, the error variance changes multiplicatively with the value of the predictor. The exponential form ensures that the variance is always positive, reflecting the log-linear variance structures often seen in applied research. Given the assumed true value of the coefficient and the observed predictor values, we generate the outcome variable.

Each pair, comprising an outcome and its corresponding predictor, represents a single observation in the population. Resampling these pairs together preserves their joint distribution and any heteroskedasticity linking them. By treating each pair as the fundamental unit and resampling with replacement, the paired bootstrap retains the original relationship between predictor and response, including variance differences across predictor values, leverage effects, and other forms of dependence.

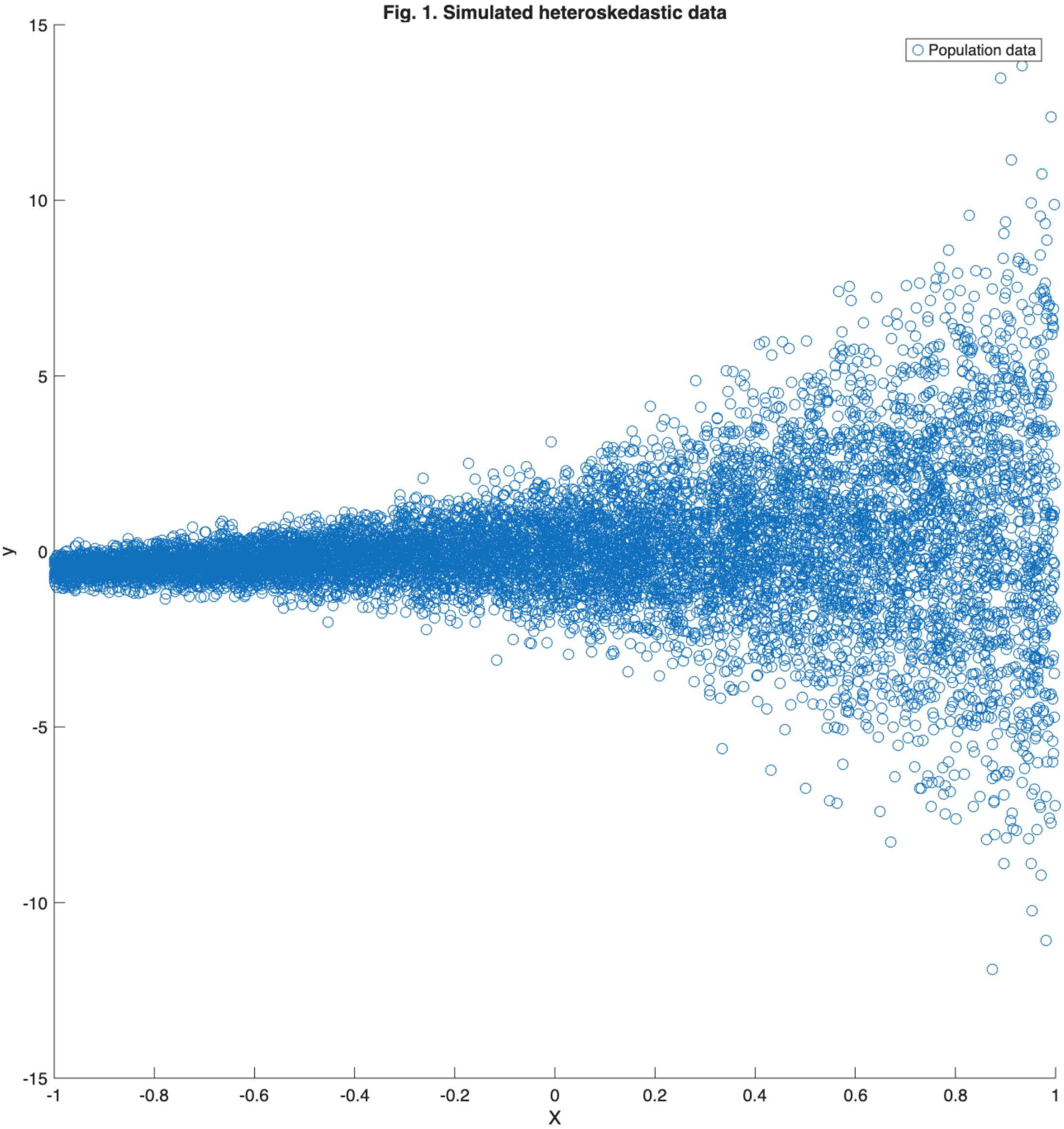
```
%% 4. Generate population data

% 4.1. Set the population size
N_obs_pop = 1000;

% 4.2. Generate data for the independent variable
X_pop = [random('Uniform',-1,1,[N_obs_pop,1])];

% 4.3. Heteroskedasticity parameter
Gamma = 1.5;

% 4.4. Generate error
u_pop = random('Normal',0,exp(X_pop*Gamma),[N_obs_pop,1]);

% 4.5. Set true beta of the model
B_true = 0.5;

% 4.6. Generate y
y_pop = X_pop*B_true+u_pop;

% 4.7. Create the (paired) population data
data_pop = [y_pop X_pop];
```

5. Plot the heteroskedastic true data

Figure 1 plots the generated data. It illustrates the presence of heteroskedasticity: the dispersion of the outcome variable increases with the predictor. That is, the variance is not constant across the values of the predictor.

```
%% 5. Plot the heteroskedastic true data
figure
hold on
```

```
scatter(X_pop,y_pop);
title('Fig. 1. Simulated heteroskedastic data');
xlabel('X');
ylabel('y');
legend('Population data');
hold off
```



6. Draw samples from the population

We conduct a Monte Carlo experiment by repeatedly drawing i.i.d. random samples from the population dataset using MATLAB's `datasample` function, in order to examine the sampling distribution of the regression coefficient estimate. For each of `N_sim` samples, the code selects `N_obs_sample` paired observations (y_i, X_i) without replacement and stores them in the three-dimensional matrix `data_samples_pop` of size $100 \times 2 \times$ `N_sim`. In this structure, rows hold individual observations, columns hold the two variables, and the third dimension indexes the simulation run, so each slice (:,:,i) contains one complete sample. The external OLS estimation function is then applied to each sample, and the resulting coefficient estimate from that run is saved in the vector B_hats_data_samples_pop for later analysis.

```
%% 6. Draw samples from the population

% 6.1. Set the sample size
N_obs_sample = 100;

% 6.2. Preallocate matrix to save (paired) samples
data_samples_pop = NaN(N_obs_sample,2,N_sim);

% 6.3. Preallocate vector to store coeficient estimates
B_hats_data_samples_pop = NaN(N_sim,1);

% 6.4. Monte Carlo sampling
for i = 1:N_sim
    data_samples_pop(:,:,i) = datasample(data_pop,N_obs_sample, ...
        'Replace',false); % Save samples in 3rd dimension
    y = data_samples_pop(:,1,i);
    X = data_samples_pop(:,2,i);
    LSS = exercisefunctionlssrobust(y,X);
    B_hats_data_samples_pop(i) = LSS.B_hat(1,1);
end
```

7. Pick an "initial" sample

Pick a sample among the samples already randomly drawn from the population and use it as the initial sample for bootstrap sampling. Note that the data comprise paired observations of the outcome and predictor, preserving their joint empirical distribution in accordance with paired bootstrap theory.

```
%% 7. Pick an "initial" sample

% 7.1. Randomly pick one sample index
sample_index = randi(N_sim);

% 7.2. Pick one sample from the previously drawn samples
data_sample = data_samples_pop(:,:,sample_index);
```

8. Draw (bootsrap) samples from the initial sample

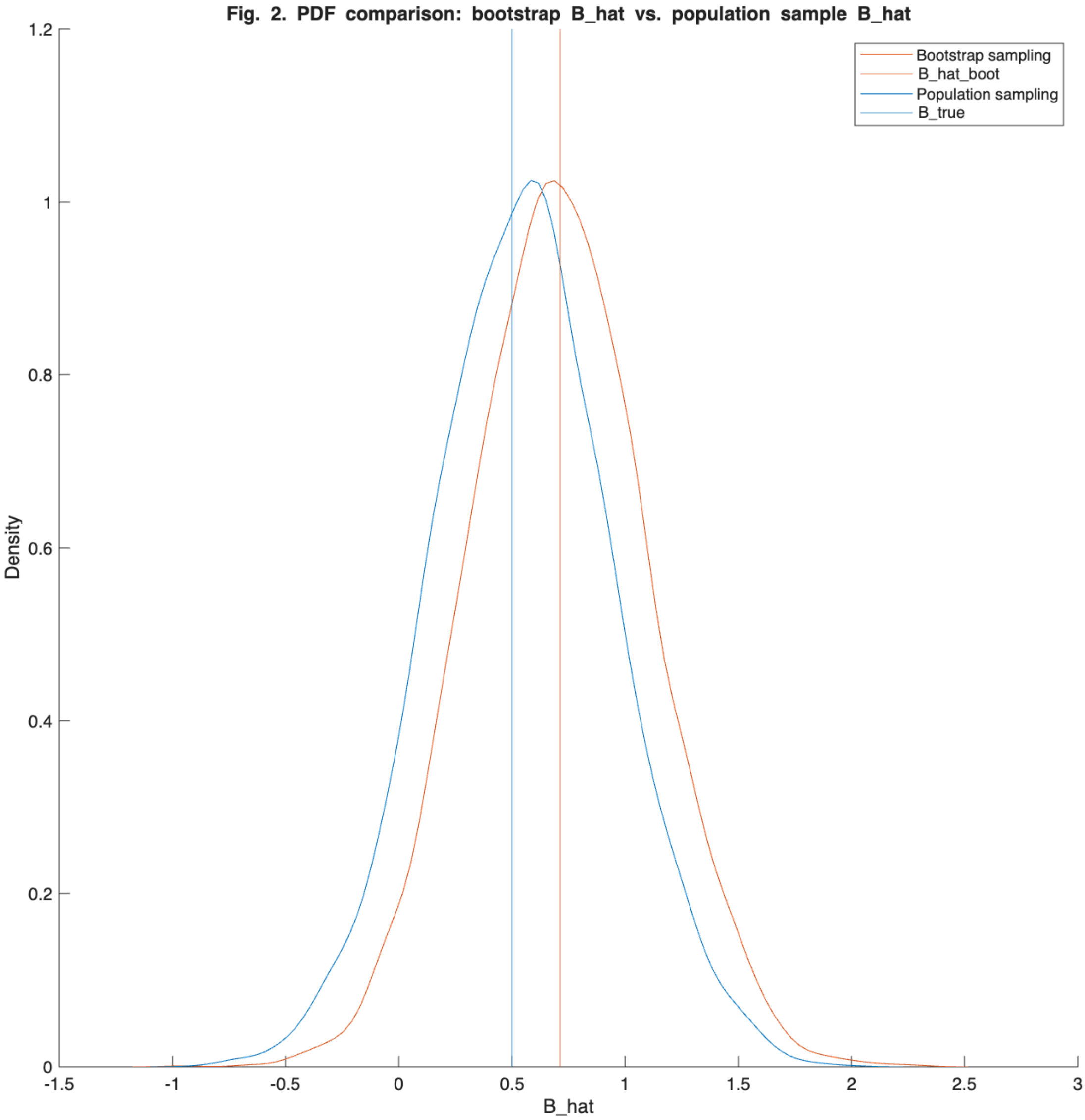
Begin by setting up an empty vector to store coefficient estimates from each simulation. For every run, draw a bootstrap sample with replacement from the initial dataset, then extract the response and predictor variables. Estimate the model using OLS and append the resulting coefficient to the vector. After N_sim repetitions, the collection of coefficients forms a bootstrap-based empirical approximation to the sampling distribution of the estimator under heteroskedastic errors.

```
%% 8. Draw (bootsrap) samples from the initial sample

% 8.1. Preallocate vector to store coefficient estimates
B_hats_data_samples_boot = NaN(N_sim,1);

% 8.6. Resample from initial sample and estimate the coefficient
for i = 1:N_sim
    data_samples_boot = datasample(data_sample,N_obs_sample, ...
        'Replace',true);
    y = data_samples_boot(:,1);
    X = data_samples_boot(:,2);
    LSS = exercisefunctionlss(y,X);
    B_hats_data_samples_boot(i) = LSS.B_hat(1,1);
end
```

9. Plot estimated PDFs

In Figure 2, we compare the sampling distribution of the coefficient estimate obtained from paired bootstrap sampling from the initial sample dataset with that obtained from direct sampling from the population dataset. Each distribution is smoothed into a PDF using the `ksdensity` function. The two PDFs exhibit very similar shapes, indicating that the bootstrap method closely reproduces the form of the true sampling distribution. A small horizontal offset between them arises because the bootstrap is based on the finite initial dataset: if the average coefficient estimate in that dataset deviates slightly from the population's true coefficient, the bootstrap distribution will reflect that shift. This illustrates that the bootstrap preserves any bias present in the initial sample while still accurately capturing sampling variability, even in the presence of heteroskedastic errors.

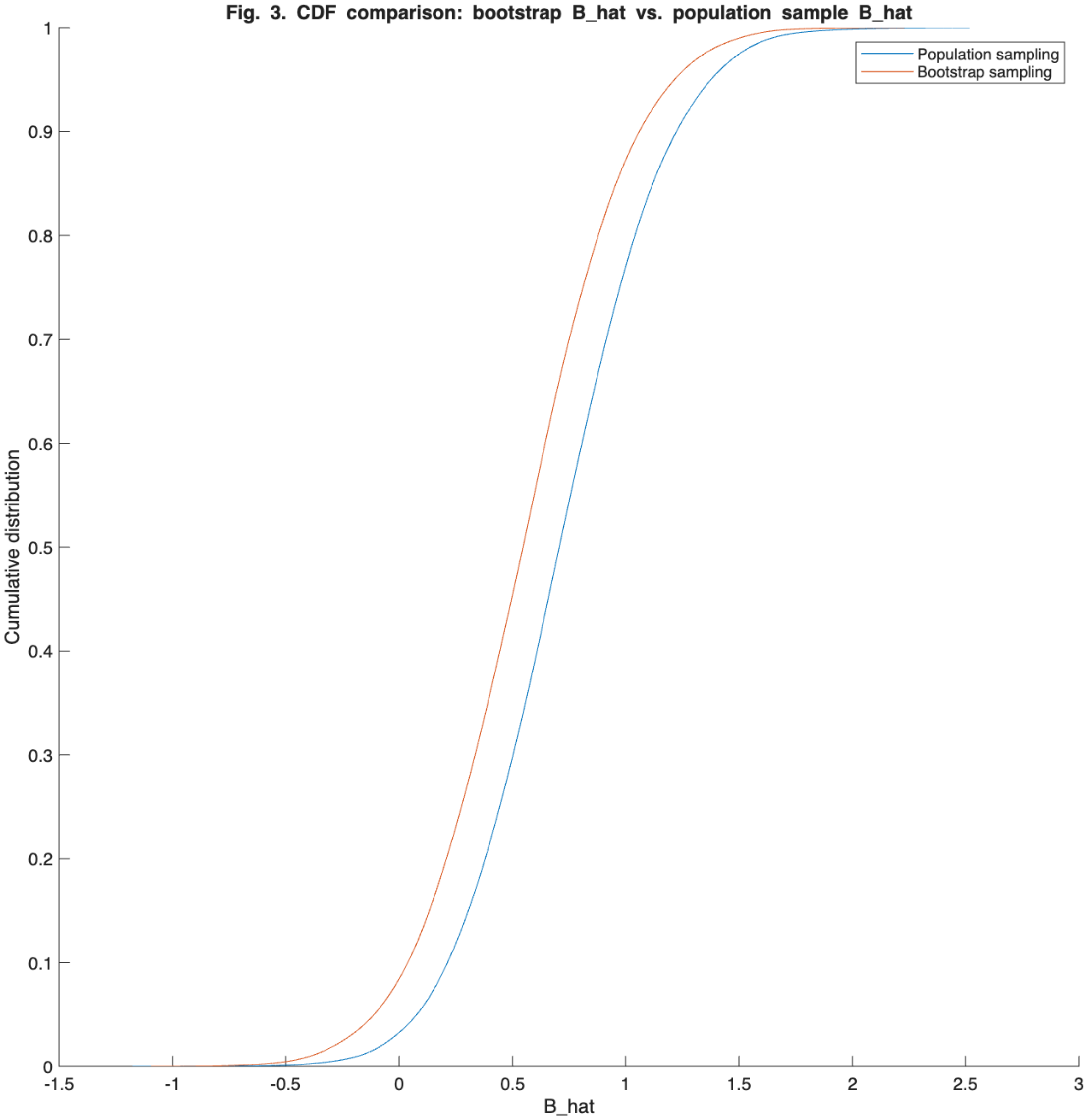
```
%% 9. Plot estimated PDFs
figure
hold on
[f_boot,x_boot] = ksdensity(B_hats_data_samples_boot,'function','pdf');
plot(x_boot,f_boot,'Color',[0.8500,0.3250,0.0980], ...
    'DisplayName','Bootstrap sampling');
xline(mean(B_hats_data_samples_boot),'Color',[0.8500,0.3250,0.0980], ...
    'DisplayName','B\_hat\_boot');
[f_pop,x_pop] = ksdensity(B_hats_data_samples_pop,'function','pdf');
plot(x_pop,f_pop,'Color',[0,0.4470,0.7410], ...
    'DisplayName','Population sampling');
xline(B_true,'Color',[0,0.4470,0.7410], ...
    'DisplayName','B\_true');
ylabel('Density');
xlabel('B\_hat');
legend('show')
title(['Fig. 2. PDF comparison: bootstrap B\_hat vs. ' ...
    'population sample B\_hat']);
hold off
```



10. Plot estimated CDFs

Figure 3 plots the CDFs of coefficient estimates from bootstrap and population sampling, plotted via kernel density estimation.

```
figure
hold on
[f_boot,x_boot] = ksdensity(B_hats_data_samples_boot,'function','cdf');
plot(x_boot,f_boot,'Color',[0,0.4470,0.7410], ...
    'DisplayName','Population sampling');
[f_pop,x_pop] = ksdensity(B_hats_data_samples_pop,'function','cdf');
plot(x_pop,f_pop,'Color',[0.8500,0.3250,0.0980], ...
    'DisplayName','Bootstrap sampling');
ylabel('Cumulative distribution');
xlabel('B\_hat');
legend('show');
title(['Fig. 3. CDF comparison: bootstrap B\_hat vs. ' ...
    'population sample B\_hat']);
hold off
```



11. Comparing the standard error estimators across different sample sizes

We evaluate the performance of the paired bootstrap as an estimator of the standard error of a coefficient estimate, comparing it with the heteroskedasticity consistent estimator and with the Monte Carlo estimator, which here represents the true standard error based on the full population. As described in Section 8, from a given sample drawn from the population, we generated multiple bootstrap resamples and computed the corresponding coefficient estimates, thereby producing an empirical sampling distribution. The standard deviation of this distribution is the bootstrap estimate of the standard error.

From a theoretical standpoint, in small samples the heteroskedasticity consistent estimator often exhibits downward bias, while the bootstrap, although still biased, can more closely approximate the true sampling variability. This is precisely what we observe at a sample size of 10: the bootstrap estimate is 0.8245, the Monte Carlo reference is 1.2883, and the heteroskedasticity consistent estimate is 0.7696, with both methods understating the variability. As the sample size increases, both estimators converge toward the true value. The bootstrap nearly matches it at a sample size of 100, with 0.3879 compared with the Monte Carlo value of 0.3944, while the heteroskedasticity consistent estimator remains slightly lower at 0.3507. This pattern aligns with asymptotic theory, in which larger sample sizes reduce bias in both methods, and the bootstrap's finite sample performance is typically superior.

```
%% 11. Comparing the SE estimators across different sample sizes

% 11.1. True estimate based on Monte Carlo simulation
SE_true = std(B_hats_data_samples_pop);

% 11.2. Heteroskedasticity-consistent estimate
SE_HC = LSS.B_hat_SEE_robust;

% 11.3. Bootstrap estimate
SE_boot = std(B_hats_data_samples_boot);
```

12. Final notes

This file is prepared and copyrighted by Axel Zoons, Quinten Salomons, and Tunga Kantarcı. This file and the accompanying MATLAB file are available on GitHub and can be accessed using this link.