Empirical Asset Pricing
Assignment 01

Morteza Aghajanzadeh* Ge Song†

January 28, 2024

Question 1

(a) Let's define the variables that we need to use in the estimation.

$$f(v_t,\theta) = \begin{bmatrix} R_{t1}-\mu_1 \\ R_{t2}-\mu_2 \\ (R_{t1}-\mu_1)^2-\sigma_1^2 \\ (R_{t2}-\mu_2)^2-\sigma_2^2 \end{bmatrix} \quad , \quad \theta = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \sigma_1^2 \\ \sigma_2^2 \end{bmatrix}$$

$$g_T(\theta) = \frac{1}{T}\sum_{t=1}^{T} f(v_t,\theta)$$

We know from the lecture that we need to calculate the $\frac{\partial f}{\partial \theta'}$ to get the $\hat{D_T}$:

$$\frac{\partial f(v_t,\theta)}{\partial \theta'} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ -2(R_{t1}-\mu_1) & 0 & -1 & 0 \\ 0 & -2(R_{t2}-\mu_2) & 0 & -1 \end{bmatrix}$$

$$\Rightarrow \hat{D_T} = \frac{1}{T}\sum_{t=1}^{T}\frac{\partial f(v_t,\theta)}{\partial \theta'} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} = -I$$

We also know that $A_T = I$ and $A_T g_T(\theta) = 0$. Therefore, we can calculate the $\hat{\theta}$:

$$A_T g_T(\theta) = 0 \Rightarrow \quad g_T(\theta) = 0$$

$$g_T(\theta) = \begin{bmatrix} \frac{\sum R_{t1}}{T} - \mu_1 \\ \frac{\sum R_{t2}}{T} - \mu_2 \\ \frac{\sum (R_{t1}-\mu_1)^2}{T} - \sigma_1^2 \\ \frac{\sum (R_{t2}-\mu_2)^2}{T} - \sigma_2^2 \end{bmatrix} = 0 \Rightarrow \begin{bmatrix} \frac{\sum_{t=1}^{T} R_{t1}}{T} \\ \frac{\sum_{t=1}^{T} R_{t2}}{T} \\ \frac{\sum_{t=1}^{T} (R_{t1}-\hat{\mu_1})^2}{T} \\ \frac{\sum_{t=1}^{T} (R_{t2}-\hat{\mu_2})^2}{T} \end{bmatrix} = \begin{bmatrix} \hat{\mu_1} \\ \hat{\mu_2} \\ \hat{\sigma_1^2} \\ \hat{\sigma_2^2} \end{bmatrix} = \hat{\theta}$$

Our calculated $\hat{\theta}$ based on the given data is:

$$\hat{\theta} = \begin{bmatrix} 0.0162 \\ 0.0045 \\ 0.0212 \\ 0.0167 \end{bmatrix}$$

*Department of Finance, Stockholm School of Economics. Email: morteza.aghajanzadeh@phdstudent.hhs.se
†Department of Finance, Stockholm School of Economics. Email: ge.song@phdstudent.hhs.se

```
    mu_1 = sum(df['Stock1'])/len(df['Stock1'])
    mu_2 = sum(df['Stock2'])/len(df['Stock2'])
    sigma_1 = sum((df.Stock1 - mu_1)**2)/(len(df.Stock1))
    sigma_2 = sum((df.Stock2 - mu_2)**2)/(len(df.Stock2))
```

Listing 1: Python code for calculating $\hat{\theta}$

(b) Still we assume that there is no serial correlation in the moments. Therefore, we can calculate the $\hat{S_T}$ as follows:

$$\hat{S_T} = \frac{1}{T}\sum_{t=1}^{T} f(v_t,\theta)f(v_t,\theta)'$$

$$= \frac{1}{T}\sum_{t=1}^{T} \begin{bmatrix} (R_{t1}-\mu_1)^2 & (R_{t1}-\mu_1)(R_{t2}-\mu_2) & (R_{t1}-\mu_1)^3-\sigma_1^2(R_{t1}-\mu_1) & (R_{t1}-\mu_1)(R_{t2}-\mu_2)^2-\sigma_2^2(R_{t1}-\mu_1) \\ (R_{t1}-\mu_1)(R_{t2}-\mu_2) & (R_{t2}-\mu_2)^2 & (R_{t1}-\mu_1)^2(R_{t2}-\mu_2)-\sigma_1^2(R_{t2}-\mu_2) & (R_{t2}-\mu_2)^3-\sigma_2^2(R_{t2}-\mu_2) \\ (R_{t1}-\mu_1)^3-\sigma_1^2(R_{t1}-\mu_1) & (R_{t1}-\mu_1)^2(R_{t2}-\mu_2)-\sigma_1^2(R_{t2}-\mu_2) & (R_{t1}-\mu_1)^4-2\sigma_1^2(R_{t1}-\mu_1)^2+\sigma_1^4 & (R_{t1}-\mu_1)^2(R_{t2}-\mu_2)^2-\sigma_1^2(R_{t1}-\mu_1)\sigma_2^2(R_{t2}-\mu_2) \\ (R_{t1}-\mu_1)(R_{t2}-\mu_2)^2-\sigma_2^2(R_{t1}-\mu_1) & (R_{t2}-\mu_2)^3-\sigma_2^2(R_{t2}-\mu_2) & (R_{t1}-\mu_1)^2(R_{t2}-\mu_2)^2-\sigma_1^2(R_{t1}-\mu_1)\sigma_2^2(R_{t2}-\mu_2) & (R_{t2}-\mu_2)^4-2\sigma_2^2(R_{t2}-\mu_2)^2+\sigma_2^4 \end{bmatrix}$$

$$= \begin{bmatrix} \hat{\sigma_1}^2 & Cov(\hat{R}_1,\hat{R}_2) & \hat{m}_1^{(3)} & \hat{k}_{12}^{(2)} \\ Cov(\hat{R}_1,\hat{R}_2) & \hat{\sigma_2}^2 & \hat{k}_{21}^{(2)} & \hat{m}_2^{(3)} \\ \hat{m}_1^{(3)} & \hat{k}_{12}^{(2)} & \hat{m}_1^{(4)} - \hat{\sigma}_1^4 & \hat{k}_{12}^{(3)} \\ \hat{k}_{21}^{(2)} & \hat{m}_2^{(3)} & \hat{k}_{12}^{(3)} & \hat{m}_2^{(4)} - \hat{\sigma}_2^4 \end{bmatrix}$$

Here the definition of $\hat{m}_1^{(j)}$ is the one in the lecture notes. For the $\hat{k}_{mn}^{(j)}$, I do not have a good way to write it in the matrix form. Therefore, I just write it as a way to sum up the terms in the matrix.

As we know that two process are independent, and normally distributed, we can calculate the $\hat{S_T}$ as follows:

$$\hat{S_T} = \begin{bmatrix} \hat{\sigma_1}^2 & 0 & 0 & 0 \\ 0 & \hat{\sigma_2}^2 & 0 & 0 \\ 0 & 0 & 2\hat{\sigma_1}^4 & 0 \\ 0 & 0 & 0 & 2\hat{\sigma_2}^4 \end{bmatrix}$$

Our calculated $\hat{S_T}$ based on the given data is:

$$\hat{S_T} = \begin{bmatrix} 0.0212 & 0 & 0 & 0 \\ 0 & 0.0167 & 0 & 0 \\ 0 & 0 & 0.0347 & 0 \\ 0 & 0 & 0 & 0.0011 \end{bmatrix}$$

```
theta = np.array([mu_1,mu_2, sigma_1, sigma_2])
def f_v(theta,x):
    mu_1 = theta[0]
    mu_2 = theta[1]
    sigma_1 = theta[2]
    sigma_2 = theta[3]
    x_1 = x[0]
    x_2 = x[1]
    f = np.array([x_1-mu_1,x_2-mu_2,(x_1-mu_1)**2-sigma_1,(x_2-mu_2)**2-
        sigma_2]).reshape(len(theta),1)
    return f
def s(theta,x):
    f = f_v(theta,x)
    return f @ f.T
x = np.array([df[['Stock1','Stock2']]])[0]
s_hat = sum([s(theta,i) for i in x])/len(x)
# set non-diagonal elements to zero
```

```
s_hat = s_hat * np.eye(4)
```

Listing 2: Python function for calculating standard error of estimation

(c) Now we want to adjust the standard errors by Newey-West estimator. Therefore, we need to calculate the $\hat{\Gamma_1}$ by using the fact that two distributions are independent:

$$\hat{\Gamma_1} = \begin{bmatrix} \frac{\sum_2^T (R_t^1-\mu_1)(R_{t-1}^1-\mu_1)}{T-1} & 0 & \frac{\sum_2^T (R_t^1-\mu_1)(R_{t-1}^1-\mu_1)^2}{T-1} & 0 \\ 0 & \frac{\sum_2^T (R_t^2-\mu_2)(R_{t-1}^2-\mu_2)}{T-1} & 0 & \frac{\sum_2^T (R_t^2-\mu_2)(R_{t-1}^2-\mu_2)^2}{T-1} \\ \frac{\sum_2^T (R_{t-1}^1-\mu_1)(R_{t-1}^1-\mu_1)^2}{T-1} & 0 & \frac{\sum_2^T (R_{t-1}^1-\mu_1)^2(R_{t-1}^1-\mu_1)^2}{T-1} + \sigma_1^4 & 0 \\ 0 & \frac{\sum_2^T (R_{t-1}^2-\mu_2)(R_{t-1}^2-\mu_2)^2}{T-1} & 0 & \frac{\sum_2^T (R_{t-1}^2-\mu_2)^2(R_{t-1}^2-\mu_2)^2}{T-1} + \sigma_2^4 \end{bmatrix}$$

and then we can calculate the $\hat{S_T}$ as follows:

$$\hat{S_T} = \begin{bmatrix} \hat{\sigma_1}^2 & 0 & 0 & 0 \\ 0 & \hat{\sigma_2}^2 & 0 & 0 \\ 0 & 0 & 2\hat{\sigma_1}^4 & 0 \\ 0 & 0 & 0 & 2\hat{\sigma_2}^4 \end{bmatrix} + \frac{1}{2}(\hat{\Gamma}_1 + \hat{\Gamma}_1')$$

Our calculated $\hat{S_T}$ based on the given data is:

$$\hat{S_T} = \begin{bmatrix} 0.0162 & 0 & 0 & 0 \\ 0 & 0.0160 & 0 & 0 \\ 0 & 0 & 0.04 & 0 \\ 0 & 0 & 0 & 0.0013 \end{bmatrix}$$

```
theta = np.array([mu_1,mu_2, sigma_1, sigma_2])
lag = 1
def gamma(theta,x,lag):
    gamma = {}
    for i in range(1,lag +1):
        lag = np.array(df[['Stock1','Stock2']].shift(i).dropna())
        tempt = []
        for num,j in enumerate(x[i:]):
            tempt.append(f_v(theta,j) @ f_v(theta,lag[num]).T)
        gamma[i] = sum(tempt)/len(tempt)
    gamma = [gamma[i] for i in gamma]
    return sum(gamma)

def s_newywest(theta,x):
    gamma_hat = gamma(theta,x,lag)
    return sum([s(theta,i) for i in x])/len(x) + 0.5 * (gamma_hat +
        gamma_hat.T)
s_hat_newywest = s_newywest(theta,x)
s_hat_newywest * np.eye(4)
```

Listing 3: Python function for calculating Newy-West standard error

(d) Now we want to compare the Sharpe ratio of two stocks. Therefore, we need to test the hypothesis that:

$$\begin{cases} H_0 : \frac{\mu_1}{\sigma_1} = \frac{\mu_2}{\sigma_2} \\ H_1 : \frac{\mu_1}{\sigma_1} \neq \frac{\mu_2}{\sigma_2} \end{cases} \Rightarrow \begin{cases} H_0 : \mu_1\sigma_2 - \mu_2\sigma_1 = 0 \\ H_1 : \mu_1\sigma_2 - \mu_2\sigma_1 \neq 0 \end{cases}$$

now we can define the $R(\theta)$ as follows:

$$R(\theta) = \mu_1\sigma_2 - \mu_2\sigma_1$$

and then rewrite the hypothesis as follows:

$$H_0 : R(\theta) = 0$$
$$H_1 : R(\theta) \neq 0$$

Now we can use the Delta method to find the distribution of $R(\hat{\theta})$:

$$\sqrt{T}(R(\hat{\theta}) - R(\theta)) \xrightarrow{d} N(0, \frac{\partial R(\theta)}{\partial \theta'} V_\theta \frac{\partial R(\theta)}{\partial \theta'})$$
$$\sqrt{T}(R(\hat{\theta})) \xrightarrow{d} N(0, \frac{\partial R(\theta)}{\partial \theta'} V_\theta \frac{\partial R(\theta)}{\partial \theta'})$$

where V_θ is the variance of $\hat{\theta}$. Now we can calculate the $\frac{\partial R(\theta)}{\partial \theta'}$ as follows:

$$\frac{\partial R(\theta)}{\partial \theta'} = \begin{bmatrix} \sigma_2 & -\sigma_1 & -\mu_2 & \mu_1 \end{bmatrix}$$

and we know that $V_\theta = \hat{S_T}$. Therefore, we can find the distribution of $R(\hat{\theta})$ as follows:

$$\frac{\partial R(\theta)}{\partial \theta'} V_\theta \frac{\partial R(\theta)}{\partial \theta'} = \begin{bmatrix} \sigma_2 & -\sigma_1 & -\mu_2 & \mu_1 \end{bmatrix} \hat{S_T} \begin{bmatrix} \sigma_2 \\ -\sigma_1 \\ -\mu_2 \\ \mu_1 \end{bmatrix} = \hat{V}_T$$

Now we can calculate the test statistic as follows:

$$T R(\hat{\theta})' \hat{V}_T^{-1} R(\hat{\theta}) \xrightarrow{d} \chi_1^2$$
$$\frac{T(\mu_1\sigma_2 - \mu_2\sigma_1)^2}{\hat{S}_T} \xrightarrow{d} \chi_1^2$$

Let's calculate the test statistic:

$$\hat{V}_T = \begin{bmatrix} 0.0167 & -0.0212 & -0.0045 & 0.0162 \end{bmatrix} \begin{bmatrix} 0.0212 & 0 & 0 & 0 \\ 0 & 0.0167 & 0 & 0 \\ 0 & 0 & 0.0347 & 0 \\ 0 & 0 & 0 & 0.0011 \end{bmatrix} \begin{bmatrix} 0.0167 \\ -0.0212 \\ -0.0045 \\ 0.0162 \end{bmatrix} = 0.001449$$

therefore, the test statistic is 1.2602 and the p-value is 0.2612. Therefore, we cannot reject the null hypothesis under 5% significance level. Therefore, we can conclude that there is no significant difference between the Sharpe ratios of two stocks.

```
R_theta = mu_1*sigma_2 - mu_2*sigma_1
R_prime = np.array([sigma_2, -sigma_1, -mu_2, mu_1])
V_T = R_prime @ s_hat @ R_prime.T
test_stat = len(df) * (R_theta)**2 / V_T
```

Listing 4: Python code for calculating the test statistics

(e) Now we need to recalculate the standard error with the results in part (c). Our $\hat{V}_T$ is equal to 0.000005 and test statistics is 3.1692. The test statistics has been increased but it is still under the critical value of 3.841, which means that we cannot reject the null hypothesis.

(f) Now we use the bootstrap method to find a distribution of the Sharpe ratios. As we conducted the sample, we realized that our 95% confidence interval contains zero, which means that still we cannot reject the null hypothesis and two stocks' Sharpe ratio are statistically indifferent.

```
  # Function to calculate Sharpe ratio
def calculate_sharpe_ratio(returns):
    return np.mean(returns) / np.std(returns)

# Parameters
n_iterations = 10000
confidence_level = 0.95

# Calculate observed Sharpe ratio difference
```

```
sharpe_ratio_diff_observed = calculate_sharpe_ratio(R1) -
    calculate_sharpe_ratio(R2)

# Bootstrap procedure

sharpe_ratio_diff_bootstrap = []
np.random.seed(123)
for _ in range(n_iterations):
    # Generate bootstrapped samples
    bootstrap_sample_stock1 = np.random.choice(R1, size=len(R1), replace=
        True)
    bootstrap_sample_stock2 = np.random.choice(R2, size=len(R2), replace=
        True)

    # Calculate Sharpe ratios for bootstrapped samples
    sharpe_ratio_stock1 = calculate_sharpe_ratio(bootstrap_sample_stock1)
    sharpe_ratio_stock2 = calculate_sharpe_ratio(bootstrap_sample_stock2)

    # Store the Sharpe ratio difference
    sharpe_ratio_diff_bootstrap.append(sharpe_ratio_stock1 -
        sharpe_ratio_stock2)

# Calculate 95% confidence interval
lower_bound, upper_bound = np.percentile(sharpe_ratio_diff_bootstrap, [(1
    - confidence_level) * 100 / 2, confidence_level * 100 - (1 -
    confidence_level) * 100 / 2])

print(f'Observed Sharpe Ratio Difference: {sharpe_ratio_diff_observed:.4f
    }')
print(f'95% Confidence Interval: [{lower_bound:.4f}, {upper_bound:.4f}]')
# The sharp ratio difference is not significant at 5% level.
```

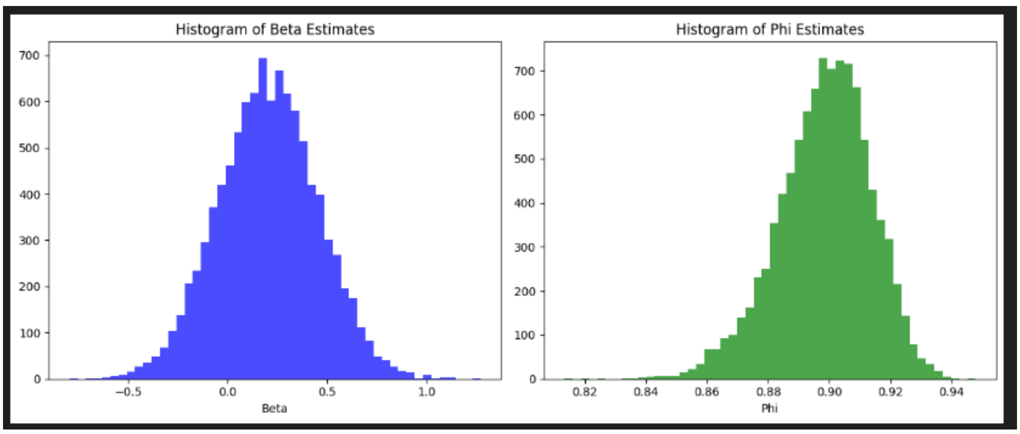
Listing 5: Python code for Bootstrapping

Question 2

(a) Optional

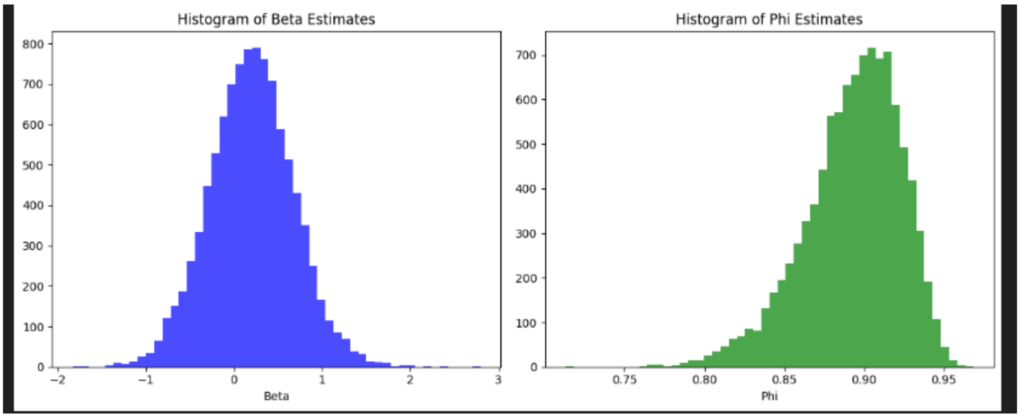
Question 3

(a) N=840



(b) N=240

Looks more skewed to the right side for $\hat{\phi}$



(c) A rejection rate of 475 out of 10,000 replications corresponds to a 4.75% rejection rate. In hypothesis testing, this would be close to the commonly chosen significance level of 5%. The reason for this difference might be that the number of replications is not big enough and we need to conduct more simulations to see a more precise result; however, 4.75% is quite close to the ideal 5% level.

(d) The null hypothesis (H0) is rejected 512 times out of 10,000 when using Newey-West standard errors, suggesting that there might be autocorrelation present in the errors, and the standard errors are adjusted to account for this.

Comparing this result to the rejection rate from problem 3b (475 out of 10,000) when standard errors assuming IID errors were used, the increased number of rejections and being closer to the 5% ideal level with Newey-West standard errors indicate that autocorrelation might be influencing the precision of the coefficient estimates.