Workshop 4: Random numbers & plotting

FIE463: Numerical Methods in Macroeconomics and Finance using Python

Richard Foltyn
NHH Norwegian School of Economics

February 12, 2026

See GitHub repository for notebooks and data:

https://github.com/richardfoltyn/FIE463-V26

Exercise 1: Static portfolio choice problem

Consider a portfolio choice problem where an investor chooses the fraction α to invest in a risky asset in order to maximize expected utility,

$$\max_{\alpha \in [0,1]} \mathbb{E}_t \left[u(W_{t+1}) \right]$$

Assume that the investor consumes all of next period's wealth W_{t+1} which is given by

$$W_{t+1} = \underbrace{R_{t+1} \alpha W_t}_{\text{Return on risky asset}} + \underbrace{R_f (1-\alpha) W_t}_{\text{Return on risk-free asset}}$$

where W_t is the initial investable wealth in period t, R_{t+1} is the gross return on the risky investment and R_f is the risk-free return on the fraction of the portfolio which is invested in a risk-free asset (e.g., a bank deposit). The utility function $u(\bullet)$ has a constant relative risk aversion (CRRA) form and is given by

$$u(W) = \begin{cases} \frac{W^{1-\gamma}}{1-\gamma} & \text{if } \gamma \neq 1 \\ \log(W) & \text{if } \gamma = 1 \end{cases}$$

where γ is a parameter governing the investor's risk aversion.

For simplicity, let the gross risk-free return be $R_f = 1$. Finally, assume that the risky return can take on two realizations, high and low, with equal probability,

$$R_{t+1} = \begin{cases} 1 + \mu + \epsilon & \text{with probability } \frac{1}{2} \\ 1 + \mu - \epsilon & \text{with probability } \frac{1}{2} \end{cases}$$

where $\mu > 0$ is the risk premium and $\epsilon > 0$ parameterizes the volatility of risky returns.

In this exercise, you are asked to compute the optimal risky share and investigate how it depends on initial wealth. Unless stated otherwise, assume that the problem is parameterized with $W = 1$, $\gamma = 2$, $\mu = 0.04$, and $\epsilon = 0.2$.

1. Compute the expected risky return, the risk premium $\mathbb{E}[R_{t+1}] - R_f$, and the Sharpe ratio for the risky asset.

   Hint: The Sharpe ratio is defined as $(\mathbb{E}[R_{t+1}] - R_f)/\sigma_R$ where σ_R is the standard deviation of risky returns.

2. Write a Python function that takes as arguments the risky share α, the initial wealth W_t, and the parameters μ, ϵ and γ, and returns the expected utility associated with the given values. Your function signature should look like this:

```python
def expected_util(alpha, W, mu, epsilon, gamma):
    """
    Return the expected utility of investing a share alpha into
    the risky asset for given parameters.

    Parameters
    ----------
    alpha: float or array
        Risky share
    W : float
        Initial wealth
    mu : float
        Risk premium
    epsilon : float
        Volatility of risky returns
    gamma : float
        Relative risk aversion

    Returns
    -------
    float or array
        Expected utility associated with choice alpha.
    """
```

Make sure that your function works correctly for both $\gamma = 1$ and $\gamma \neq 1$. Moreover, the function should allow for the arguments α and W to be passed as both scalar values and NumPy arrays.

3. Using the function you wrote, evaluate the expected utility on a grid of risky shares α with 1001 points that are uniformly spaced on the interval $[0, 1]$.

   Plot this expected utility against the risky share. Label both axes and add a legend to your plot.

4. In the previous question, you plotted the expected utility against *all* possible risky share choices. Using grid search, find the optimal risky share and the associated expected utility.

   Augment the plot you created earlier and add a vertical dashed line to indicate the optimal risky share.

   Hint: You can use `np.argmax()` to locate the utility-maximizing risky share.

   Hint: Use `axvline()` to add a vertical line.

5. Now consider a set of 100 initial wealth levels W_t that are uniformly spaced over the interval $[1, 10]$.

   - Write a loop that computes the optimal risky share for each of these wealth levels. Use the same values for the remaining parameters as above.
   - Plot the optimal risky share against the grid of initial wealth levels. Set the y-axis limits to $(0, 1)$ to better visualize the result and explain your findings.

Exercise 2: Computing the mean for increasing sample sizes

In this exercise, you are asked to investigate how the mean of a simulated sample depends on the sample size.

Consider a set of random draws with sample sizes of 5, 10, 50, 100, 500, 1000, 5000, 10000, and 100000.

1. For each sample size, draw a normally distributed sample with mean $\mu = 1$ and standard deviation $\sigma = 1$. Use a seed of 678 and make sure to reset the seed for each sample size.

2. For each sample size, compute the sample mean using `np.mean()`. Print a list of sample sizes and sample means.
3. Plot the means you computed against the sample size on the x-axis. You should use a log scale for the x-axis and a marker symbol `'o'` to make the graph easier to read. Add a dashed horizontal line to indicate the true mean, and label both axes.

   Hint: You can set the axis scale using `xscale()`.

   Hint: You can add a horizontal line with `axhline()`.

Exercise 3: Ordinary Least Squares (OLS)

In this exercise, you are asked to draw a sample from a bivariate normal distribution (i.e., a multivariate normal distribution where the number of dimensions is 2) and estimate a linear relationship using Ordinary Least Squares (OLS).

Consider the linear model

$$y_i = \alpha + \beta x_i + \epsilon_i$$

where x is the explanatory (or exogenous, or independent) variable and y is the outcome (or dependent) variable. The relationship is imperfect, so for each observation i there is an additive error term ϵ_i.

1. Assume that (x, y) are drawn from a bivariate normal distribution,

   $$(x, y) \overset{\text{iid}}{\sim} \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

   with

   $$\boldsymbol{\mu} = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \qquad \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

   where $\boldsymbol{\mu}$ is the vector of means, $\boldsymbol{\Sigma}$ is the variance-covariance (VCV) matrix, σ_x and σ_y are the standard deviations for x and y, respectively, and ρ is the correlation between x and y.

   For this exercise, let $\mu_x = 0$, $\mu_y = 1$, $\sigma_x = 0.1$, $\sigma_y = 0.5$, and $\rho = 0.7$.

   Compute and report the VCV matrix implied by these parameters.
2. Use the variance-covariance matrix and the vector of means to draw $N = 200$ observations of (x, y) from a bivariate normal distribution. Use a seed of `123`.

   Create a scatter plot with x on the x-axis to visualize the sample.

   Hint: The bivariate normal is just a special case of the multivariate normal distribution, so you can use multivariate_normal() for this task.
3. Since this linear model has only one explanatory variable, the estimate of β denoted by $\widehat{\beta}$ is given by the formula

   $$\widehat{\beta} = \frac{\widehat{\text{Cov}}(x, y)}{\widehat{\text{Var}}(x)}$$

   where $\widehat{\text{Cov}}(x, y)$ is the sample covariance between x and y, and $\widehat{\text{Var}}(x)$ is the sample variance of x.

   With $\widehat{\beta}$ in hand, the estimate for the intercept α is given by the formula

   $$\widehat{\alpha} = \overline{y} - \widehat{\beta}\overline{x}$$

   where $\overline{y}$ and $\overline{x}$ are the sample means of y and x, respectively.

   Use the above formulas to compute the OLS estimate for $(\widehat{\alpha}, \widehat{\beta})$ for the sample you have drawn.

   Hint: Use the functions np.mean(), np.var(), and np.cov() to compute the sample mean, variance, and covariance.

4. Augment the scatter plot you created earlier and add the regression line using the estimates $(\widehat{\alpha}, \widehat{\beta})$. This regression line passes through the coordinate $(0, \widehat{\alpha})$ and has a slope given by $\widehat{\beta}$.

   Hint: Use the function axline() to add a straight line to a plot.

5. Since you know the parameters of the true model, you can compare the estimate $\widehat{\beta}$ to the true value β.

   For this model, the true β is given by the formula

   $$\beta = \frac{\rho \sigma_x \sigma_y}{\sigma_x^2}$$

   Compute this true β and compare it to your estimate $\widehat{\beta}$.