

Mandatory assignment

FIE463: Numerical Methods in Macroeconomics and Finance using Python

About the mandatory assignment

- Deadline: Wednesday, February 18 at 12:00 (noon)
- Your solution needs to be submitted on Canvas.
- The assignment is to be completed *individually*.
- You are allowed to use AI for support, but you are not allowed to hand in output directly produced by AI as your own solution (see [AI Regulations](#) on the Canvas course page).
- The assignment will be graded as pass/fail. A passing grade is required to receive credit in this course.

Requirements

- You must submit a *single* Jupyter notebook which contains your solution.
- Make sure your notebook runs without errors (Restart and Run all).
- Your implementation *must* work with the FIE463 Anaconda environment we created in the course (if you haven't done so, see the Canvas announcements for instructions).
- Your notebook should be well structured and visually appealing, e.g.,:
 - Use markdown cells with headings and other formatting elements (see [here](#) and [here](#)).
 - Each part should be in a clearly distinguishable sub-section.
 - Each function should be defined in its own code cell.
 - Each graph should be in its own cell.
 - Graphs should be visually appealing. They should include legends and labels so they are easy to comprehend.
 - Don't print hundreds of lines of output in your final notebook.
- Your code should satisfy the following criteria:
 - Your code is reusable and avoids duplication.
 - Your code should be efficient. For example, use NumPy arrays and vectorization instead of loops where it makes sense.
 - Your code should be commented to help the reader understand what it is doing.
- Be sure to provide an explanation for your results where applicable.

Hints

- If you are stuck, make sure to consult the past workshops which cover simpler variants of the tasks below.
- There is no need to use fancy NumPy functions we haven't covered to solve this assignment.

Optimal savings and portfolio choice

An investor with initial wealth w lives for two periods and maximizes expected lifetime utility,

$$\max_{c_1, s, \alpha} \left\{ u(c_1) + \beta \mathbb{E} [u(c_2)] \right\}$$

subject to the constraints

$$c_1 + s = w$$

$$c_2 = R\alpha s + R_f(1 - \alpha)s$$

$$c_1 \geq 0, s \geq 0, \alpha \in [0, 1]$$

where (c_1, c_2) are the consumption levels in periods 1 and 2, respectively, s are savings in period 1, and α is the fraction of s to be invested in the risky asset while the remaining fraction $1 - \alpha$ is invested in the risk-free asset. The parameter β denotes the discount factor.

The flow utility function $u(\bullet)$ has a constant absolute risk aversion (CARA) form and is given by

$$u(c) = -\exp(-\gamma c)$$

where γ is a parameter governing the investor's absolute risk aversion and $\exp(\bullet)$ is the exponential function. In finance and asset pricing, CARA utility is frequently used because it often yields closed-form solutions for optimal portfolios and prices, especially when combined with normally distributed returns. A key property is that the optimal dollar amount invested in risky assets is independent of the level of wealth.

The risk-free asset yields a gross return R_f , which is assumed to be $R_f = 1$. The risky return is given by $R = R_f + \mu + \sigma\eta$ where μ is the expected excess return, σ^2 is the variance of the risky return, and η is a shock representing uncertainty. In this assignment, we approximate the distribution of the risky return R using three possible realizations for the shock η :

$$\eta = \begin{cases} -3^{0.5} & \text{with probability } \pi = 1/6 \\ 0 & \text{with probability } \pi = 2/3 \\ 3^{0.5} & \text{with probability } \pi = 1/6 \end{cases}$$

In this assignment, you are asked to compute the optimal savings level s and risky share α and investigate how these depend on parameters. Unless stated otherwise, use the following parameter values throughout the assignment:

Symbol	Description	Value
β	Discount factor	0.9
γ	Absolute risk aversion	4.0
σ^2	Variance of risky return shock	0.16^2
μ	Expected excess return	0.04
w	Initial wealth	1.0
R_f	Risk-free gross return	1.0

Tasks

1. You first want to verify that the discretized risky return approximates the true mean and variance of the underlying normal distribution well.

To this end, compute the expected gross risky return $\mathbb{E}[R]$, the variance of the risky return $\text{Var}(R)$, the expected excess return (or risk premium) $\mathbb{E}[R] - R_f$, and the Sharpe ratio $(\mathbb{E}[R] - R_f)/\sigma$, where $\sigma = \sqrt{\text{Var}(R)}$.

Comment on whether these results are close to the relevant parameters μ and σ^2 in the table above.

2. Write a Python function `util()` to evaluate the flow utility u for a given consumption level and parameters:

```
def util(c, gamma):  
    """  
    Return the flow utility of consumption for a given risk aversion.  
    """
```

The function should allow for the argument c to be a scalar or a NumPy array.

Plot the flow utility function $u(c)$ for $c \in [0, 2]$ for three different values of $\gamma \in \{1, 4, 10\}$.

Hint: Use `np.exp()` to evaluate the exponential function.

3. Write a Python function `expected_util()` that evaluates the expected lifetime utility for given choices (s, α) and parameters:

```
def expected_util(s, alpha, beta, gamma, sigma, mu, w, etas, probs):  
    """  
    Calculate the expected lifetime utility for given parameters.  
    """
```

where `etas` is the array of the three possible realizations of η and `probs` is the array of the corresponding probabilities.

You should use the function `util()` you just wrote to accomplish this task. The function should allow for arguments s and α to be both scalars or arrays.

4. You want to visualize the expected lifetime utility for selected values of possible choices of (s, α) . To this end, consider three values for α from the set $\{0, 0.5, 1\}$.
 - For each α , use `expected_util()` to evaluate the associated expected lifetime utility on a grid of 200 candidate savings levels that are uniformly spaced on the interval $[0.1, w - 0.1]$.
 - Create a single graph that depicts this lifetime utility plotted against the candidate savings grid on the x -axis for each of the alphas (i.e., the plot should contain three distinct lines).

Add a legend and axes labels to clearly show what is being plotted.

Explain the intuition behind the shape of the expected utilities as shown in the graph you created.

5. Write a Python function `find_optimum()` which locates the optimal choices (s, α) for given parameters using grid search:

```
def find_optimum(beta, gamma, sigma, mu, w, etas, probs):  
    """  
    Find the optimal savings level and risky share to maximize expected utility.  
    """
```

This function should return the optimal choices and the maximized expected utility as a tuple (s, α, U_{\max}) .

Inside the function, create the following candidate grids for s and α :

- s : 1000 points uniformly spaced on the interval $[10^{-6}, w - 10^{-6}]$

- α : 1000 points uniformly spaced on the interval $[0, 1]$

Evaluate the expected lifetime utility for all possible combinations of (s, α) on these grids to locate the optimal choice.

Hint: Make sure to call `expected_util()` in vectorized fashion instead of looping over both s and α .

6. Use the function `find_optimum()` to find the optimal choices and report these together with the maximized expected utility.
7. Now consider a set of 101 values for the parameter γ which are uniformly spaced on the interval $[1, 10]$.
 - Compute the optimal risky share for each value of γ . Use the same values for the remaining parameters as above.
 - Plot the optimal risky share against the grid of gammas. How does the optimal risky share depend on the absolute risk aversion? Explain your findings.
8. Finally, consider a set of 101 values for the parameter σ which are uniformly spaced on the interval $[0.01, 0.5]$.
 - Compute the optimal risky share for each value of σ . Use the same values for the remaining parameters as above.
 - Plot the optimal risky share against the grid of sigmas. How does the optimal risky share depend on σ ? Explain your findings.