Problem Set 1

Quantitative Economics, Fall 2025

November 10, 2025

This problem set consists of four problems. Deadline: Nov. 26th 11:59 PM.

Problem 1: Julia basics and Random.jl

Recall the Lindeberg-Levy Central Limit Theorem:[1]

$$\frac{\bar{X}_n - \mu}{\sigma\sqrt{n}} \xrightarrow{d} N(0,1)$$

for $\{X_1, X_2, \ldots\}$ i.i.d. with $E[X_i] = \mu$ and $Var[X_i] = \sigma^2 < \infty$. Here $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$.

Your task is to illustrate it for $X_i \sim \text{Pois}(\lambda)$.[2] You will have to plot four histograms of 1000 realizations of $\bar{X}_n$, for $n = 5, 25, 100, 1000$, (in blue) and superimpose the density of $N(0,1)$ on each of them (in red).

We will set $\lambda = 1$ for simplicity. Your task is to write code that will generate the requested plots. There is more than one way of doing it! For example: you can calculate averages by summation or by using a function that already exists in Julia.[3]

You will have to use the *Random.jl* package to generate the random numbers and *Distributions.jl* to specify the distribution.[4] *StatsPlots.jl* will be useful for plotting the density function.

Your script should return the four plots.

[1] See here for more details.

[2] Recall that for $X \sim \text{Pois}(\lambda)$, $E[X] = \lambda$ and $Var[X] = \lambda$.

[3] We do not ask you to write first a function that works for arbitrary argument values (including even the density function). We however encourage you to think how to make your code well-organized, easy to read and modular. Use this simple exercise as an opportunity to think about these issues.

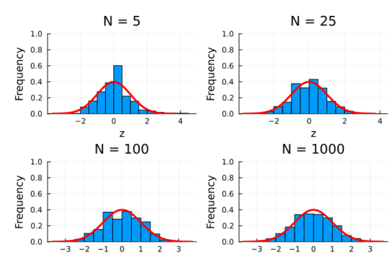

Figure 1: You should produce something like this.

[4] See the *ps1_hint.jl* file for a hint.

Problem 2: Some linear algebra

Let

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & \alpha-\beta & \beta \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \alpha \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

The system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is relatively easy to solve by hand.

Steps to follow:

1. Your first task is to write a function that takes α and β as arguments and returns the exact solution. You will have to do algebra by hand to find it – start from the last equation.

2. Next, write a function that takes α, β as arguments and return the exact solution (using the previous function), the solution obtained by using the backslash operator, the relative residual, and the condition number.

3. Use your functions to create the table that shows x_1 (both exact and obtained by using the backslash operator), the condition number, and the relative residual for $\alpha = 0.1$ and $\beta = 1, 10, 100, \ldots, 10^{12}$. This table has to be generated in your script automatically.

Moreover, answer the following question (it can be answered in comments at the end of your script):

- What patterns do you notice? Summarize your findings.

Problem 3: Finding an equilibrium in a simple exchange economy

Consider a simple endowment economy with two goods, populated by two agents. Both agents have preferences given by

$$u_i(c_{i,1}, c_{i,2}) = \left(\alpha_i c_{i,1}^{\frac{\sigma_i - 1}{\sigma_i}} + (1 - \alpha_i)\, c_{i,2}^{\frac{\sigma_i - 1}{\sigma_i}} \right)^{\frac{\sigma_i}{\sigma_i - 1}},$$

[5] where $c_{i,1}, c_{i,2}$ is consumption of good 1 and good 2 by agent i, and $\alpha_i, \sigma_i \geq 0$. We assume that consumption of neither good can be negative, $c_{i,1}, c_{i,2} \geq 0$.

Agents maximize their utility subject to budget constraints

$$p_1 c_{i,1} + p_2 c_{i,2} = p_1 \omega_{i,1} + p_2 \omega_{i,2},$$

where p_1, p_2 are prices of good 1 and good 2, and where endowments of agent i are $\omega_{i,1}, \omega_{i,2}$.

Basic algebra delivers demand functions:[6]

$$c_{i,1}(p_1, p_2) = \frac{\alpha_i^{\sigma_i} p_1^{-\sigma_i}}{\alpha_i^{\sigma_i} p_1^{1-\sigma_i} + (1-\alpha_i)^{\sigma_i} p_2^{1-\sigma_i}} \cdot [p_1 \omega_{i,1} + p_2 \omega_{i,2}],$$

$$c_{i,2}(p_1, p_2) = \frac{(1-\alpha_i)^{\sigma_i} p_2^{-\sigma_i}}{\alpha_i^{\sigma_i} p_1^{1-\sigma_i} + (1-\alpha_i)^{\sigma_i} p_2^{1-\sigma_i}} \cdot [p_1 \omega_{i,1} + p_2 \omega_{i,2}].$$

Market clear when

$$\sum_{i=1,2} c_{i,1} = \sum_{i=1,2} \omega_{i,1}, \quad \sum_{i=1,2} c_{i,2} = \sum_{i=1,2} \omega_{i,2}.$$

The task is to find all Walrasian equilibria of this economy: prices p_1, p_2 and allocations $(c_{i,1}, c_{i,2})\, \forall i \in 1,2$ that: (i) given prices, allocations solve utility maximization problems of agents; (ii) all markets clear.

[5] This a CES (Constant Elasticity of Substitution) utility function. The elasticity of substitution between the two goods is equal to σ_i.

[6] If you do not why, try to solve the utility maximization problem on paper first!

We can do it as follows: normalize one of the prices to 1, find the price of the other good that clears one of the two markets. If we normalize $p_2 = 1$, this amount to finding values of p_1 that satisfy

$$\sum_{i=1,2} c_{i,1}(p_1, 1) = \sum_{i=1,2} \omega_{i,1}.$$

It is a simple nonlinear equation in p_1. Given the p_1, you can find equilibrium allocations by plugging the price back into demand functions.

You will need to write a function that takes all parameters of the model as inputs and returns p_1 that solves the equation above.[7] Be careful about the stopping criteria of your nonlinear equation solver. Provide a justification why your stopping criterion is reasonable.

Use the above function in the script that does the following:[8]

Let $\sigma_1 = \sigma_2 = 0.2$ and $\alpha_1 = 1 - \alpha_2 = x$. Let endowments of the first agent be $(1, 1)$ and endowments of the second agent be $(0.5, 1.5)$ for all agents be equal to 0.5.

1. Plot equilibrium price p_1 as a function of $x \in (0, 1)$.
2. Plot equilibrium allocations $c_{1,1}(x)$ and $c_{2,1}(x)$ as a function of $x \in (0, 1)$ on the same graph.
3. Repeat steps 1-2 for $\sigma_1 = \sigma_2 = 5$.
4. Compare the equilibrium prices across the two cases ($\sigma = 0.2$ vs $\sigma = 5$).
5. For which values of x (if any) do the two agents consume equal amounts of both goods?

Moreover, answer the following question (it can be answered in comments at the end of your script):

- How does the elasticity of substitution affect equilibrium prices and allocations? When is the equilibrium price more sensitive to x? Summarize your findings.

[7] Think how to do it in a smart way, without passing too many arguments to the function.

[8] You are responsible for checking that your solution is indeed an equilibrium! Be sure to verify that the solver has converged to a solution that satisfies the market clearing condition within your specified tolerance. You will not get credit for incorrect solutions!

Problem 4: A minimum variance portfolio

Consider a portfolio of n risky assets in a single-period mean-variance framework. Each asset i has expected return μ_i and the assets have a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$. An investor wants to find the **minimum variance portfolio** that achieves a target expected return $\bar{\mu}$.

As you will see, this will lead to a system of linear equations.

The investor's problem is:

$$\min_{\mathbf{w}} \frac{1}{2} \mathbf{w}^T \mathbf{\Sigma} \mathbf{w}$$

subject to:

$$\mathbf{w}^T \boldsymbol{\mu} = \bar{\mu}, \quad \mathbf{w}^T \mathbf{1} = 1$$

where $\mathbf{w} \in \mathbb{R}^n$ is the vector of portfolio weights, $\boldsymbol{\mu} \in \mathbb{R}^n$ is the vector of expected returns, and $\mathbf{1}$ is a vector of ones.

The solution to this problem is given by:

$$\begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\mu} & \mathbf{1} \\ \boldsymbol{\mu}^T & 0 & 0 \\ \mathbf{1}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \bar{\mu} \\ 1 \end{bmatrix}.$$

Here we use λ_1 and λ_2 as Lagrange multipliers on the two constraints.

Your tasks:

1. You need to load data on asset returns from the provided `asset_returns.csv` file. Each column corresponds to an asset, and each row corresponds to a time period (daily returns).

2. Compute the sample mean vector $\boldsymbol{\mu}$ and sample covariance matrix $\boldsymbol{\Sigma}$ of asset returns. Your code should work for any number of assets and report the values of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.

3. Form the $(n+2) \times (n+2)$ matrix system $\mathbf{Ax} = \mathbf{x}$ above with target return $\bar{\mu} = 0.10$. Report the condition number of the $\mathbf{A}$ matrix.

4. Solve the system using the following methods:

   (a) Solve the system using Julia's backslash operator.

   (b) We cannot apply the Jacobi or Gauss-Seidel methods directly here – note that these methods require division by the diagonal elements of the matrix, which in this case has zeros on the diagonal. Instead, consider a system

   $$\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}.$$

   Check if $\mathbf{A}^T \mathbf{A}$ has zeros on the diagonal. Calculate the condition number of $\mathbf{A}^T \mathbf{A}$. Is this matrix diagonally dominant? If so, implement the Jacobi or Gauss-Seidel method to solve this system.

   (c) Conjugate Gradient method (CG) cannot be applied directly to the original system – it requires that the matrix is symmetric and positive definite. Verify if $\mathbf{A}^T \mathbf{A}$ satisfies these properties.[9] If so, implement the Conjugate Gradient method to solve the system involving $\mathbf{A}^T \mathbf{A}$.[10]

   (d) Use GMRES (the generalized minimal residual method) to solve the original system directly. [11]

[9] You can use `issym(·)` to check symmetry and `isposdef(·)` to check positive definiteness.

[10] Use the function `cg(·)` from the *IterativeSolvers.jl* package.

[11] Use the function `gmres(·)` from the *IterativeSolvers.jl* package.

(e) Let

$$\mathbf{P} = \begin{bmatrix} \mathrm{diag}(\boldsymbol{\Sigma}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0}^T & 1 & 0 \\ \mathbf{0}^T & 0 & 1 \end{bmatrix}.$$

Solve the preconditioned system $\mathbf{P}^{-1}\mathbf{A}\mathbf{x} = \mathbf{P}^{-1}\mathbf{b}$ using GMRES. You can either implement the preconditioner manually by working with the new system, or use the preconditioning option in the `gmres(·)` function.

5. For each method, report:
   - Number of iterations to converge (if applicable)
   - Total computational time
   - Relative residual norm $\frac{\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2}{\|\mathbf{b}\|_2}$
6. Report the optimal portfolio weights $\mathbf{w}$ obtained from each method. Verify and report that the weights sum to 1 and that the expected return constraint is satisfied. Report the portfolio variance $\sigma_p^2 = \mathbf{w}^T\boldsymbol{\Sigma}\mathbf{w}$.

Asset pricing models often require solving many similar systems (e.g., for different target returns or market conditions).

1. Choose your preferred method and solve the system for 50 different values of $\bar{\mu} \in [0.01, 0.10]$. For each new target return, use the previous solution as the initial guess for the iterative method.
2. Plot the efficient frontier by solving the optimization problem for multiple target returns $\bar{\mu} \in [0.01, 0.10]$. For each point, record portfolio variance σ_p^2 and plot σ_p (standard deviation) against $\bar{\mu}$.