Computational Economics 2020

Assignment 1

(due: Monday, April 6, 2020, 11:59 PM (GMT + 1))

The assignment is split into part A and part B which are weighted equally with 15% each. You are allowed to form groups of 2 (if there is a odd number of people, groups of 3 are allowed too). Each group has to present at least one of their solutions.

Please follow these instructions for handing in your assignment:

1. Submit the assignment via Email to Philipp (philipp.mueller@business.uzh.ch). The Email should contain:
   - A single PDF-file with the names of all group members, and assignment number on page 1. The file should contain all your answers and results.
   - The source code in a separate zip archive. The code should be well documented and readable.
2. Only the students taking the course for credits are getting feedback to their solution; feel free to send your solution anyway. A sample solution will be published after the deadline. If you have any specific question, reach out on GitHub `https://github.com/KennethJudd/CompEcon2020/issues`.

Refrain from sharing complete solutions.
For the content of the PDF we expect the following:

1. Provide a brief introduction/ motivation for the problem.
2. Explain how you solved the exercise and show the most relevant calculations (formulas and essential parts of the code) with brief comments.
3. Concisely interpret the results of the exercise.
4. For each exercise the floating text should not exceed 2 pages (this does not include formulas, codes, graphs, and tables).

Exercise A-1

A consumer has a utility $u(x_1, x_2, x_3, x_4)$ over four goods. She can spend \$1 on buying quantities of these four goods, all of which have a price of \$1. After substituting the budget equation, $x_1 + x_2 + x_3 + x_4 = 1$, into the utility function the consumer wants to maximize $u(x_1, x_2, x_3, 1 - x_1 - x_2 - x_3)$.

(A) Solve the unconstrained optimization problem with $u(x_1, x_2, x_3, x_4) = e^{x_1} + \ln(x_2) + \sqrt{x_3} - \frac{1}{x_4}$.

(B) Now use $u(x_1, x_2, x_3, x_4) = \sum_{i=1}^{4} \left(i \frac{x_i^{1-\gamma}}{1-\gamma} \right)$ for $\gamma \in \{10^{-2}, 10^{-1}, \frac{1}{2}, 2, 10, 50, 100\}$ and use a constrained optimizer to ensure $x_i \in [0, 1]$ and $\sum_i x_i = 1$. Try a variety of different starting points. How sensitive is the optimal solution to the starting point?

Exercise A-2

Consider the endowment economy with m goods and n agents. Assume that agent i's utility function over the m goods is

$$u^i(x) = \left(\sum_{j=1}^{m} a_j^i \, x_j^{\nu_j^i} \right)^{\gamma_i} .$$

Suppose that agent i's endowment of good j is e_j^i. Assume that $a_j^i,\ e_j^i > 0$, and $0 < \gamma_i,\ \nu_j^i < 1$, $i = 1, \cdots, n$, $j = 1, \cdots, m$. The social planner's problem with n consumers is

$$\begin{aligned} \max_{x_j^i,\ i=1,..,n,\ j=1,...,m} \quad & \sum_{i=1}^{n} \lambda_i u^i(x^i) \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_j^i \le \sum_{i=1}^{n} e_j^i,\ j = 1, .., m \end{aligned}$$

Write a program (using any solver you can access) that will read in the ν_j^i, a_j^i, and e_j^i values and the social weights $\lambda_i > 0$, and output the solution to the social planner's problem. Report the results to four significant digits in three tables, one for each social weight vector, like the following:

good	agent 1 consumption	agent 2 consumption	⋮
1	?	?	⋮
⋮	⋮	⋮	⋮

On April 01, we will send you values for the endowments, utility functions, and social weights for your assignment. Your program should be debugged before that time.

Exercise A-3

Apply the stopping rule for linearly convergent sequences where you estimate the rate of convergence to the following sequences:

(A) $x_k = \sum_{n=0}^{k} \frac{5^n}{n!}$

(B) $x_k = \sum_{n=1}^{k} (2n+1)^{-3}$

(C) $x_k = \sum_{n=1}^{k} (3n-2)^{-1.7}$

(D) $x_k = \sum_{n=1}^{k} n^{-0.92}$

(E) $x_0 = 5$, $x_{k+1} = (2/3)x_k + x_k^{-2}$

(F) $x_0 = 5$, $x_{k+1} = (1 + x_k - 4x_k^2)/(4 - 6x_k)$

Use $\epsilon = .01$, .001, and .0001. For each ϵ and each sequence, display the final appoximate answer, the error of the approximation, and how many iterations it took (by the way, a sequence may not converge, so do not do more than 100,000 iterations in any case). Use tables such as

Approximate result

	ϵ :		
	.01	.001	.0001
(a):	?	?	?
(b):	?	?	?
(c):	?	?	?
(d):	?	?	?

Number of terms

	ϵ :		
	.01	.001	.0001
(a):	?	?	?
(b):	?	?	?
(c):	?	?	?
(d):	?	?	?

1 Exercise A-4

In this exercise, you are asked to explore your machine and the achievable machine speed.

(A) List all of the following information: Processor manufacturer, name, and number; Number of logical and physical cores; CPU-core frequency; CPU maximum frequency and whether your CPU supports Turbo Boost.

(B) Write programs to determine the relative and absolute speeds expressed as floating point operations per second of addition, multiplication, division, exponentiation (e^x), logarithm (base 10), sine, tangent, hyperbolic sine, density function for the Normal distribution, and the cumulative Normal distribution function on your computer in the programming language of your choice.

(C) Determine the floating point operations per second for the multiplication of a vector times a square matrix. Implement two versions: (i) element-wise by using for-loops and (ii) use any vectorized operations available in your programming language. Repeat both for increasing number of elements $\{200, 300, \ldots, 4000\}$. Report the floating point operations per second.

Be sure to write the programs so that you get sensible answers. (Ensure that there is not too much overhead in the background and average over many repertitions of the experiment.)