Exercise class 10

Introduction to Programming and Numerical Analysis

Class 3 and 6

Annasofie Marckstrøm Olesen

Spring 2024

UNIVERSITY OF COPENHAGEN

Solvng equations

Problem set 6

Linear equations

"Linear equations" refer to anything, that can be written in matrix form:

$$Ax = b \iff \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \iff x = A^{-1}b$$

There are routines for inverting matrices - but matrix inversion may be costly, so sometimes we use other algorithms:

- **Gauss-Jordan** elimination (probably not more efficient)
- **Gauss-Seidel** iterations are faster (but may not converge)
- **LU-factorization** has no matrix inversion, so very fast!

Non-linear equations

With non-linear equation systems, we have to be more general in our approach. Often, we can use a root finder:

$$f(x) = b \iff \hat{f}(x) = 0, \quad \hat{f}(x) = f(x) - b$$

Root-finding algorithms include among others:

- **Bisection** no gradient, but may be slow.
- **Newton** or **Halley** use gradient (and Hessian) to update guesses efficiently.
- **Brent** finds an efficient combination.

Algorithms work well under different conditions - you are welcome to experiment!

Symbolic Python

Symbolic Python (`sympy`) can do math and solve equations *analytically* - everything is very much like what we know from micro and macro courses! This is nice because:

- More familiar when we are used to solving equations analytically.
- Can (sometimes) find all solutions to problems with more than one solution.
- Can solve exactly, not just down to a numerical approximation.

Not so nice because:

- Can only do what we could also have done with pen and paper.
- Does not improve your understanding of numerical methods.
- Some (many) models have no analytical solution.

Problem set 6

Tasks: Solving equations

- Linear equations using LU-factorization and Gauss-Jordan elimination
- Non-linear equations using SymPy

Problem: Solving the Solow model

- With Cobb-Douglas Production using SymPy and root finding
- With CES-production using root finding

Problem set 6

Plan for today

- Now-16.00: Work on tasks
- 16.00-16.15: Break
- 16.15-16.45: Work on problem
- 16.45-17: Talk about the solution to the problem (Solving the Solow model)

Next time...

Video lectures:

- Unconstrained and constrained optimization
- Dynamic optimization

Exercises

- Problem set 7: The household problem with income risk
- **Note:** The problem set for next time is very long and covers some difficult concepts, so it may be a good idea to have a look at it before class.