Computational Macroeconomics
Exercises

Week 1

Willi Mutschler
willi@mutschler.eu

Version: April 19, 2022

Contents

1 What is Computational Macroeconomics

Broadly define the scope and research topics of “Computational Macroeconomics”, sometimes referred to as “Numerical Methods in Economics”. What are the challenges and approaches?

Readings

- Fernández-Villaverde, Rubio-Ramírez, and Schorfheide (2016, Part I)
- Judd (1998, Ch. 1)
- L. Maliar and S. Maliar (2014)
- Schmedders and Judd (2014)

2 What are DSGE models?

What are dynamic stochastic general equilibrium (DSGE) models? What are the challenges in solving and simulating DSGE models?

Readings

- Fernández-Villaverde, Rubio-Ramírez, and Schorfheide (2016, Ch. 1)
- Herbst and Schorfheide (2016, Ch. 1)

3 Programming Language

1. Name some popular programming languages in Macroeconomics. Which are general purpose, which are domain specific?

2. What are the differences between compiled and interpreted languages?

3. What is important when choosing a programming language for scientific computing in Macroeconomics? Which programming language(s) would you choose?

Readings

- Aruoba and Fernández-Villaverde (2015) (note the Update available at `https://www.sas.upenn.edu/~jesusfv/Update_March_23_2018.pdf`)
- Aguirre and Danielsson (2020)

4 Quick Tour: MATLAB

Install the most recent version of MATLAB with the following Toolboxes: Econometrics Toolbox, Global Optimization Toolbox, Optimization Toolbox, Parallel Computing Toolbox, Statistics and Machine Learning Toolbox, Symbolic Math Toolbox. Open a new script and do the following:[1]

1. Define the column vectors

$$x = (-1, 0, 1, 4, 9, 2, 1, 4.5, 1.1, -0.9)' \qquad y = (1, 1, 2, 2, 3, 3, 4, 4, 5, \mathrm{nan})'$$

2. Check if both vectors have the same length using either `length()` or `size()`.
3. Perform the following logical operations:

$$x < y \qquad x < 0 \qquad x + 3 \geq 0 \qquad y < 0$$

4. Check if all elements in x satisfy both $x + 3 \geq 0$ and $y > 0$.
5. Check if all elements in x satisfy either $x + 3 \geq 0$ or $y > 0$.
6. Check if at least one element of y is greater than 0.
7. Compute $x + y$, xy, xy', $x'y$, y/x, and x/y.
8. Compute the element-wise product and division of x and y.
9. Compute $\ln(x)$ and e^x.
10. Use `any` to check if the vector x contains elements satisfying $\sqrt{x} \geq 2$.
11. Compute $a = \sum_{i=1}^{10} x_i$ and $b = \sum_{i=1}^{10} y_i^2$. Omit the nan in y when computing the sum.
12. Compute $\sum_{i=1}^{10} x_i y_i^2$. Omit the nan in y when computing the sum.
13. Count the number of elements of $x > 0$.
14. Predict what the following commands will return:
    `x.^y`   `x.^(1/y)`   `log(exp(y))`   `y*[-1,1]`   `x+[-1,0,1]`   `sum(y*[-1,1],1,'omitnan')`
15. Define the matrix $X = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$. Print the transpose, dimensions and determinant of X.
16. Compute the trace of X (i.e. the sum of its diagonal elements).
17. Change the diagonal elements of X to [7,8,9].
18. Compute the eigenvalues of (the new) X. Display a message if X is positive or negative definite.
19. Invert X and compute the eigenvalues of X^{-1}.
20. Define the column vector $a = (1, 3, 2)'$ and compute `a'*X`, `a'.*X`, and `X*a`.
21. Compute the quadratic form $a'Xa$.

[1]If you don't have any previous programming experience, I would highly recommend to go through Brandimarte (2006, Appendix A), Miranda and Fackler (2002, Appendix B), and Pfeifer (2017).

22. Define the matrices $I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $Y = \begin{bmatrix} 1 & 4 & 7 & 1 & 0 & 0 \\ 2 & 5 & 8 & 0 & 1 & 0 \\ 3 & 6 & 9 & 0 & 0 & 1 \end{bmatrix}$ and $Z = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

23. Generate the vectors

$$\begin{aligned} x_1 &= (1, 2, 3, \dots, 9)\,, & x_2 &= (0, 1, 0, 1, 0, 1, 0, 1)\,, & x_3 &= (1, 1, 1, 1, 1, 1, 1, 1) \\ x_4 &= (-1, 1, -1, 1, -1, 1)\,, & x_5 &= (1980, 1985, 1990, \dots, 2010)\,, & x_6 &= (0, 0.01, 0.02, \dots, 0.99, 1) \end{aligned}$$

using sequence operators `:` or `repmat`.

24. Generate a grid of $n = 500$ equidistant points on the interval $[-\pi, \pi]$ using `linspace`.

25. Compare `1:10+1`, `(1:10)+1` and `1:(10+1)`.

26. Define the following column vectors

$$\begin{aligned} x &= \begin{pmatrix} 1 & 1.1 & 9 & 8 & 1 & 4 & 4 & 1 \end{pmatrix}, \quad y = \begin{pmatrix} 1 & 2 & 3 & 4 & 4 & 3 & 2 & \text{NaN} \end{pmatrix}' \\ z &= \begin{pmatrix} true & true & false & false & true & false & false & false \end{pmatrix}' \end{aligned}$$

27. Predict what the following commands will return (and then check if you are right):
`x(2:5)`, `x(4:end-2)`, `x([1 5 8])`, `x(repmat(1:3,1,4))`,
`y(z)`, `y(~z)`, `y(x>2)`, `y(x==1)`,
`x(~isnan(y))`, `y(~isnan(y))`

28. Indexing is not only used to read certain elements of a vector but also to change them. Execute `x2 = x` to make a copy of `x`. Change all elements of `x2` that have the value 4 to the value -4. Print `x2`.

29. Change all elements of `x2` that have the value 1 to a missing value (`nan`). Print `x2`.

30. Execute `x2(z) = []`. Print `x2`.

31. Define the matrix $M = \begin{pmatrix} 1 & 5 & 9 & 12 & 8 & 4 \\ 2 & 6 & 10 & 11 & 7 & 3 \\ 3 & 7 & 11 & 10 & 6 & 2 \\ 4 & 8 & 12 & 9 & 5 & 1 \end{pmatrix}$ using the : operator and the `reshape` command.

32. Predict what the following commands will return (and then check if you are right):
`M(1,3)`, `M(:,5)`, `M(2,:)`, `M(2:3,3:4)`, `M(2:4,4)`, `M(M>5)`, `M(:,M(1,:)<=5)`, `M(M(:,2)>6,:)`, `M(M(:,2)>6,4:6)`

33. Print all rows where column 5 is at least three times larger than column 6.

34. Count the number of elements of M that are larger than 7.

35. Count the number of elements in row 2 that are smaller than their neighbors in row 1.

36. Count the number of elements of M that are larger than their left neighbor.

Readings

- Brandimarte (2006).
- Miranda and Fackler (2002).
- Pfeifer (2017)

5 ARMA(1,1) simulation

Consider the ARMA(1,1) model:

$$x_t - \theta x_{t-1} = \varepsilon_t - \phi\varepsilon_{t-1}$$

where $\varepsilon_t \sim N(0,1)$.

1. Compute the non-stochastic steady-state with pen and paper, i.e. what is the value of x_t if $\varepsilon_t = 0$ for all t?
2. Write a Dynare mod file for this model:
   - x is the endogenous variable, ε the exogenous variabel, and θ and ϕ are the parameters.
   - Set $\theta = \phi = 0.4$.
   - Write either a `steady_state_model` or `initval` block and compute the steady-state.
   - Start at the non-stochastic steady-state and simulate 200 data points using a `shocks` block and the `stoch_simul` command. Drop the first 50 observations and plot both x as well as ε.
   - Try out different values for θ and ϕ. What do you notice?
3. Redo the exercise in MATLAB without using Dynare.

References

Aguirre, Alvaro and Jon Danielsson (2020). *Which Programming Language Is Best for Economic Research: Julia, Matlab, Python or R?* https://voxeu.org/article/which-programming-language-best-economic-research. Blog.

Aruoba, S. Borağan and Jesús Fernández-Villaverde (2015). "A Comparison of Programming Languages in Macroeconomics". In: *Journal of Economic Dynamics and Control* 58, pp. 265–273. ISSN: 01651889. DOI: 10.1016/j.jedc.2015.05.009.

Brandimarte, Paolo (2006). *Numerical Methods in Finance and Economics: A MATLAB-based Introduction*. 2nd ed. Statistics in Practice. Hoboken, N.J: Wiley Interscience. ISBN: 978-0-471-74503-7.

Fernández-Villaverde, Jesús, Juan F. Rubio-Ramírez, and Frank Schorfheide (2016). "Solution and Estimation Methods for DSGE Models". In: *Handbook of Macroeconomics*. Ed. by John B. Taylor and Harald Uhlig. Vol. A. Elsevier North-Holland, pp. 527–724. ISBN: 978-0-444-59469-3.

Herbst, Edward and Frank Schorfheide (2016). *Bayesian Estimation of DSGE Models*. The Econometric and Tinbergen Institutes Lectures. Princeton University Press. ISBN: 978-0-691-16108-2.

Judd, Kenneth L. (1998). *Numerical Methods in Economics*. Vol. 1. The MIT Press.

Maliar, Lilia and Serguei Maliar (2014). "Numerical Methods for Large-Scale Dynamic Economic Models". In: *Handbook of Computational Economics Volume 3*. Ed. by Karl Schmedders and Kenneth L. Judd. Elsevier, pp. 325–477. DOI: 10.1016/B978-0-444-52980-0.00007-4.

Miranda, Mario Javier and Paul L. Fackler (2002). *Applied Computational Economics and Finance*. Cambridge, Mass. London: MIT. ISBN: 978-0-262-63309-3.

Pfeifer, Johannes (2017). *MATLAB Handout*.

Schmedders, Karl and Kenneth L. Judd (2014). "Introduction for Volume 3 of the Handbook of Computational Economics". In: *Handbook of Computational Economics*. Vol. 3. Elsevier, pp. xv–xvii. ISBN: 978-0-444-52980-0. DOI: 10.1016/B978-0-444-52980-0.00020-7.