The Journal of Open Source Software

MacroModelling.jl: A Julia package for developing and solving dynamic stochastic general equilibrium models

Thore Kockerols [1]

1 Norges Bank, Norway

DOI: 10.21105/joss.05598

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @gdalle
- @jmejia8

Submitted: 01 June 2023
Published: 25 September 2023

Summary

`MacroModelling.jl` is a Julia (Bezanson et al., 2017) package for developing and solving dynamic stochastic general equilibrium (DSGE) models. These kinds of models describe the behavior of a macroeconomy and are particularly suited for counterfactual analysis (economic policy evaluation) and exploring / quantifying specific mechanisms (academic research).

The goal of this package is to reduce coding time and speed up model development by providing functions for working with discrete-time DSGE models. The user-friendly syntax, automatic variable declaration, and effective steady state solver facilitate fast prototyping of models. The package includes several pre-defined models from prominent economic papers, providing an immediate starting point for users. The target audience for the package includes central bankers, regulators, graduate students, and others working in academia with an interest in DSGE modelling.

The package supports programmatic model definition. Once the model is defined, the package finds the solution for the model dynamics knowing only the model equations and parameter values. The model dynamics can be solved for using first, (pruned) second, and (pruned) third-order perturbation solutions (Andreasen et al., 2017; O. Levintal, 2017; Villemot, 2011), leveraging symbolic and automatic differentiation. Furthermore, the package can be used to calibrate parameters, match model moments, and estimate the model on data using the Kalman filter (Durbin & Koopman, 2012). The package is designed to be user-friendly and efficient. Once the functions are compiled and the non-stochastic steady state (NSSS) has been found, the users benefit from fast and convenient functions to generate outputs or change parameters.

Statement of Need

Due to the complexity of DSGE models, efficient numerical tools are required, as analytical solutions are often unavailable. `MacroModelling.jl` serves as a tool for handling the complexities involved, such as forward-looking expectations, nonlinearity, and high dimensionality.

`MacroModelling.jl` differentiates itself among macroeconomic modelling packages by offering a unique blend of capabilities and conveniences, such as automatic declaration of variables and parameters, automatic differentiation with respect to parameters, and support for perturbation solutions up to 3rd order. While it operates within the Julia environment, it presents an alternative to the MATLAB-dominated field, which includes dynare (Adjemian et al., 2022), RISE (Maih, 2015), Taylor Projection (Oren Levintal, 2018), NBTOOLBOX, and IRIS, the latter two being capable of providing only 1st order perturbation solutions.

Other Julia-based packages such as DSGE.jl, StateSpaceEcon.jl, SolveDSGE.jl, and DifferentiableStateSpaceModels.jl (Childers et al., 2022) have functionalities similar to those of `MacroModelling.jl`. However, the former are not as general and convenience-focused as the

MATLAB packages and `MacroModelling.jl`. The Julia-based packages are missing convenience functionalities such as automatic creation of auxiliary variables for variables in lead and lags larger than 1, or programmatic model definition. These functionalities are convenient to the user but require significant effort to implement in the parser. Furthermore, the other Julia-based packages do not possess the unique feature set of `MacroModelling.jl` regarding variable declaration and automatic differentiation. Notably, the Python-based dolo.py offers global solutions, but does not include estimation features which are available in `MacroModelling.jl`.

`MacroModelling.jl` stands out as one of the few packages that can solve non-stochastic steady states symbolically, a feature shared only with gEcon (Klima et al., 2015), an R-based package. When, as in most cases, symbolic solution is not possible `MacroModelling.jl` uses symbolic simplification, search space transformation, automatic domain restrictions, restarts with different initial values, warm starts using previous solutions, and a Levenberg-Marquardt-type optimizer. The combination of these elements makes it possible to solve all 16 models currently implemented in the examples out-of-the box. This is remarkable because all other packages rely either on analytical NSSS derivation by hand, or on a smaller subset of the features outlined above. In general this makes the other packages far less reliable in finding the NSSS without further information (e.g. a close enough initial guess). Furthermore, unlike many of its competitors, the domain-specific model language of `MacroModelling.jl` is integrated into the Julia language, which makes for convenient reading and coding, with the help of Julia macros.

Example

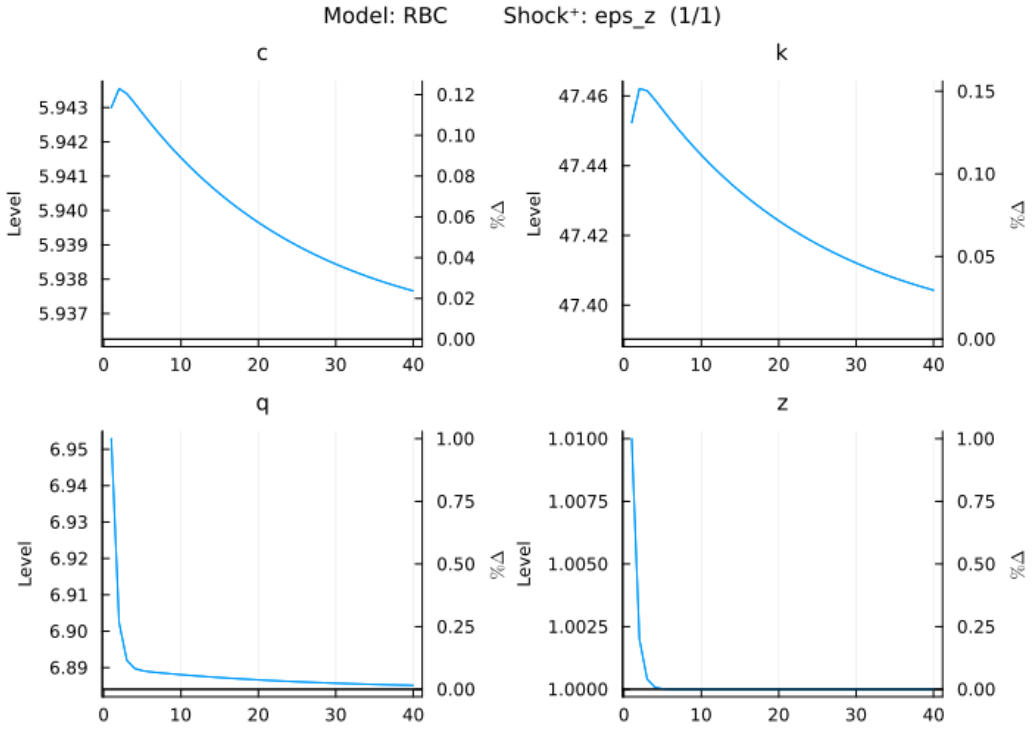
One relatively simple example to study with the package is a real business cycle model (see e.g. (Kydland & Prescott, 1982)). The model describes the dynamics of an economy with households consuming a consumption good `c`, produced by competitive firms using the capital stock `k`, and an exogenous technology process `z`. The households maximize their utility (`log(c)`) and decide whether to invest in the capital stock or consume. The firms decide the amount of production inputs and the quantity of output with the goal to maximize their profits while minimizing their costs. The capital stock depreciates by the factor `δ` and the production technology takes the form: `k^α`. All agents discount the future with `β` and the exogenous AR(1) technology process is governed by parameters `ρ` and `σ_z`. Given the optimization problems of the households and firms one can write down the first-order optimality conditions as follows:

```
using MacroModelling
import StatsPlots

@model RBC begin
    1 / c[0] = (β / c[1]) * (α * z[1] * k[0]^(α - 1) + (1 - δ))
    c[0] + k[0] = (1 - δ) * k[-1] + q[0]
    q[0] = z[0] * k[-1]^α
    z[0] = (1 - ρ) + ρ * z[-1] + σ_z * eps_z[x]
end

@parameters RBC begin
    σ_z = 0.01
    ρ   = 0.2
    δ   = 0.02
    α   = 0.5
    β   = 0.95
end

plot_irf(RBC)
```



The plot shows both the level, percentage deviation from the NSSS as well as the NSSS itself. Note that the code to generate the impulse response function (IRF) plot contains only the equations, parameter values, and the command to plot. Solving the model using first-order perturbation happens automatically in the background.

Acknowledgements

The author thanks everybody who opened issues, reported bugs, contributed ideas, and was supportive in driving `MacroModelling.jl` forward.

References

Adjemian, S., Bastani, H., Juillard, M., Karamé, F., Mihoubi, F., Mutschler, W., Pfeifer, J., Ratto, M., Rion, N., & Villemot, S. (2022). *Dynare: Reference manual version 5* (Dynare Working Papers No. 72). CEPREMAP.

Andreasen, M. M., Fernández-Villaverde, J., & Rubio-Ramírez, J. F. (2017). The Pruned State-Space System for Non-Linear DSGE Models: Theory and Empirical Applications. *The Review of Economic Studies*, *85*(1), 1–49. https://doi.org/10.1093/restud/rdx037

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Childers, D., Fernández-Villaverde, J., Perla, J., Rackauckas, C., & Wu, P. (2022). *Differentiable State-Space Models and Hamiltonian Monte Carlo Estimation* (NBER Working Papers No. 30573). National Bureau of Economic Research, Inc. https://doi.org/10.3386/w30573

Durbin, J., & Koopman, S. J. (2012). *Time series analysis by state space methods, 2nd edn*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199641178.001.0001

Klima, G., Podemski, K., Retkiewicz-Wijtiwiak, K., & Sowińska, A. E. (2015). *Smets-Wouters '03 model revisited - an implementation in gEcon* (MPRA Paper No. 64440). University Library of Munich, Germany.

Kydland, F. E., & Prescott, E. C. (1982). Time to build and aggregate fluctuations. *Econometrica*, *50*(6), 1345–1370. https://doi.org/10.2307/1913386

Levintal, O. (2017). Fifth-order perturbation solution to DSGE models. *Journal of Economic Dynamics and Control*, *80*, 1–16. https://doi.org/10.1016/j.jedc.2017.04.007

Levintal, Oren. (2018). Taylor Projection: A New Solution Method For Dynamic General Equilibrium Models. *International Economic Review*, *59*(3), 1345–1373. https://doi.org/10.1111/iere.12306

Maih, J. (2015). *Efficient perturbation methods for solving regime-switching DSGE models* (Working Paper No. 2015/01). Norges Bank. https://doi.org/10.2139/ssrn.2602453

Villemot, S. (2011). *Solving rational expectations models at first order: What dynare does*. Dynare Working Papers 2, CEPREMAP.