Research among Copycats: R&D, Spillovers, and Feedback Strategies – Code Description and Instructions –

Grega Smrkolj* Florian Wagener†

September 28, 2018

1 Overview

In the accompanying folders we provide programme code that can be used to replicate the results of Smrkolj and Wagener (2017). There are two groups of programmes:

- Fortran code to compute the feedback Nash equilibrium value function and the corresponding drift vector field.
- Matlab code to compute all derivative results and plots.

In what follows, we provide brief instructions for running the code.

2 Fortran code

To compile the code, a Fortran 95 compiler is required. We have successfully compiled the code using both the Intel Fortran compiler *ifort* and the GNU Fortran compiler *gfortran*. The latter is freely available on the GNU Fortran project website.

2.1 Compilation

To generate the executable, the following Fortan files have to be compiled:
`main.f90`, `problem_data.f90`, `initialise.f90`, `comp.f90`, `io_methods.f90`, `iterate.f90`, `refine.f90`.

We provide a 'makefile'. If the *make* command is installed on the target system, the executable can be automatically generated and ran using the command `make run`.

*Newcastle University Business School. Corresponding author. *Address:* Newcastle University Business School, 5 Barrack Road, Newcastle upon Tyne, NE1 4SE, UK; grega.smrkolj@newcastle.ac.uk.

†*University of Amsterdam* and *Tinbergen Institute*; f.o.o.wagener@uva.nl.

2.2 Initialization

The problem parameters are set in the files `main.f90` and `problem_data.f90`.

`main.f90`: The main program has two loops. The inner loop uses small time steps to achieve the convergence for the value function. As explained in our paper, the criterion for convergence is that the value of the L^2-norm of $\mathbf{U}_s$ is below 1×10^{-12}. The outer loop starts at some initial value of noise ε (the default value, $\varepsilon = 0.125$, is specified in line 15 of the code) and reduces it by a small factor (specified in line 16) in each next iteration. This allows one to observe the smallest ε for which the convergence in the numerical integration (inner loop) can be achieved. We suggest to use a higher level of noise than the one specified (e.g., $\varepsilon = 0.250$) and achieve the desired lower value through subsequent iterations. This is especially important when the code is run for lower levels of spillovers β, as there a high spike in the policy function may appear on the diagonal of the state space at lower values of noise.

`problem_data.f90`: The parameters `xmin` (the default is -2.5) and `xmax` (the default is 4.5) are specified in line 4. These are the highest and the lowest value of unit cost on the grid (recall the relation $c = e^{-x}$).

The main model parameters are specified in line 7: `phi` (the default is $\phi = 8$), `beta` (the default is $\beta = 0.5$), and `rho` (the default is $\rho = 1$).

2.3 Results

The executable produces two outputs: `value_xxx.dat` and `state_vf_xxx.dat`, where `xxx` is substituted by the number of time steps used to achieve convergence. The first file gives the value function of firm 1, while the second one gives velocity vectors for unit costs $(\dot{x}_1, \dot{x}_2)$ at each grid point (x_1, x_2). These define the drift vector field for unit costs based on equilibrium policy functions. Both files are used as inputs in MATLAB code.

3 MATLAB code

To run the code, MATLAB 9.0 or later is required. The MATLAB files use `value_xxx.dat` and/or `state_vf_xxx.dat` as inputs. The location of these Fortran files must be specified at the beginning of each MATLAB file. For convenience, we provide two sample files `value05.dat` and `state_vf05.dat`. These are the value function and the vector field, respectively, for the default values of parameters, $(\beta, \phi, \rho, \varepsilon) = (0.5, 8, 1, 0.125)$.

The script `response.m` calculates equilibrium R&D efforts (policy function) and creates plots for R&D efforts and the value function.

The script `VectorField.m` calculates and plots the steady states of the drift vector field, their stable and unstable manifolds, and entry/exit curves for the product market. It calls the function `getAllVertices.m`.

The function `getAllVertices.m` supports the calculation of steady states in the previous file by providing the $dx_i/dt = 0$ isoclines.

The script `det_path_RK.m` calculates the time paths, based on the drift vector field, of unit costs, R&D efforts, and quantity produced, using a 3rd order Runge-Kutta method.

The script `stoch_path.m` simulates stochastic time paths for units costs and R&D efforts for a given initial value of unit costs, specified in lines 42–43, using the Euler-Maruyama method.

The script `stoch_path_surplus.m` simulates stochastic time paths for initial unit costs along the iso-sum loci. In the next step, it calculates the expected net present value of consumer surplus, producer surplus, and total surplus. This is the slowest part of the code. To shorten the execution time, the number of simulated time paths, specified in line 35, can be reduced. The default value is 1000.

References

Smrkolj, G. and Wagener, F. (2017): "Research among Copycats: R&D, Spillovers, and Feedback Strategies," *Tinbergen Institute Discussion Paper*, TI 2014-112/II.