Computer codes for: "From Dual to Unified Employment Protection: Transition and Steady State" by Juan J. Dolado, Etienne Lalé and Nawid Siassi

October 28, 2020

1 Overview of the content

The replication package contains 3 folders:

- `baseline`: Computations based on the baseline model, with Matlab files to replicate the tables and figures presented in the paper and additional statistics and Monte Carlo simulation exercises not reported in the paper;
- `approx_model_savings`: Computations for the approximate model with savings summarized in Section 4 of the paper and presented in details in the online appendix;
- `model_extensions`: This folder contains three subfolders that correspond to the model extensions presented in Section 6 of the paper:
  - `1_wage_rigidity`: Computations for the model extension featuring wage rigidities;
  - `2_match_heterogeneity`: Computations for the model extension with heterogeneity in initial match productivity;
  - `3_human_capital`: Computations for the model extension with human capital accumulation.

Matlab codes are used to run computations of the steady-state equilibrium of the model. Fortran codes are used to run computations of the transition dynamics of the model. The folder `approx_model_savings` contains only Matlab codes since the transition dynamics of this model is too complex to be computed (see the section of the online appendix devoted to the approximate model with savings for details). The subfolders `2_match_heterogeneity` and `3_human_capital` contain Fortran codes to compute the steady-state equilibrium of the model since, again, we do not compute the transition dynamics of these two model extensions.

2 File dictionary

Matlab files in folder `baseline` and `model_extensions`:

- `main.m` is the program where the computation starts;

- `init.m` defines values of parameters, defines productivity grids, EPL scheme and initializes guess;
- `vfi.m` solves the model by value function iteration;
- `simul.m` computes the stationary distribution;
- `showstats.m` displays some statistics (e.g. for calibration) and compares distributions to the data;
- `reformEPL.m` runs the numerical experiment to find the unified EPL scheme;
- `initx.m` (only in the folder `baseline`) is a reduced version of `init.m`.

Additional Matlab files contained in folder `baseline`:

- `compstat.m` checks alternative calibrations and performs comparative statics exercises for UI replacement rates, etc.;
- `montecarlo.m` runs Monte Carlo simulation exercises to calculate earnings volatility and autocorrelation;
- `modelplot.m` is a script to create figures (EPL scheme, wage function, productivity thresholds, etc.).

Matlab files in folder `approx_model_savings`:

- `mainsavings.m` is the program where the computation starts;
- `labormarket.m` computes the earnings process as implied by the labor market part of the model;
- `init_labormarket.m` defines values of parameters, defines productivity grids, etc.;
- `initx_labormarket.m` is a reduced version of `init_labormarket.m`;
- `vfi_labormarket.m` solves Bellman equations in labor market part of the model;
- `simul_labormarket.m` computes distribution for the labor market part of the model;
- `showstats_labormarket.m` displays statistics for the labor market part of the model;
- `periodreturnssavings.m` computes single-period returns;
- `vfirsavings.m` solves Bellman equation for retired workers, `vfiosavings.m` solves Bellman equation for older workers and `vfiysavings.m` solves Bellman equation for young workers;
- `distsavings.m` computes the stationary distribution with savings;
- `evalresults.m` produces and evaluates allocational results from the model;
- `evalwelfare.m` produces and evaluates welfare results from the model;
- `stataux.m` translates decision rule for savings;
- `popsamp.m` creates an artificial population sample;
- `gini.m` - Computes the Gini coefficient.

Fortran computer codes:

- `main.f90` is the program where the computation starts;
- `globals.f90` defines variables that will be assigned values and sets the size of asset values, decisions rules, wages and distributions;
- `misc_functions.f90` contains various subroutines and functions to:
  - define values of parameters, define productivity grids, and approximate the match productivity process;
  - initialize value functions, distributions and guess the time paths of taxes;
  - compute the EPL scheme during the transition path;
  - compute workers' marginal utility.
- `solve_model.f90` computes the equilibrium transition path;
- `wage_function_old.f90` computes the wage and asset values of older workers;
- `wage_function_young.f90` computes the wage and asset values of young workers;
- `distribution.f90` updates the population distribution one period ahead;
- `results_transition.f90` computes various statistics (separation rates, job-finding rates, etc.) along the transition path;
- `results_welfare.f90` computes the welfare impacts (aggregate and within sub-groups of the population) of the EPL reform.

Additional Fortran codes:

- `rigid_wage.f90` (in `1_wage_rigidity`) computes the rigid wage and asset values of young and older workers;
- `distribution.f90` (in `2_match_heterogeneity` and `3_human_capital`) computes the stationary distribution;

3 Running the programs

Both Matlab and Fortran computer codes in each folder have a `main` wrap-up program that initializes the computation and calls on the files to run all computations. To compute the transition dynamics, it is best to organize the folder as follows:

- Create a subfolder called `results` that contains the specification files `params.txt` and `reform.txt` (see Model specification below). Final results will be printed as .txt files in this subfolder;
- Create a subfolder `t0` that contains: (i) equilibrium quantities for the initial steady state (tax κ, tightness θ, vacancy posting cost k, duration of joblessness Δ, average wage $\widetilde{w}$), (ii) EPL scheme of the initial steady state, (iii) population distribution of the initial steady-state equilibrium;
- Create a subfolder `t1` that contains: (i) equilibrium quantities for the final steady state (tax κ, tightness θ, vacancy posting cost k, duration of joblessness Δ, average wage $\widetilde{w}$), (ii) EPL scheme of the final steady state, (iii) wage and asset values of final steady-state equilibrium;

- Create a subfolder `t0_in_t1` that contains: the wage and asset values of being employed under the initial, t_0 EPL scheme, given the equilibrium quantities of the final steady-state equilibrium.

Prior to running the Fortran codes, the content of the subfolders `t0`, `t1`, and `t0_in_t1` should be pre-computed and stored as .txt files using the Matlab codes.

Fortran programs should be compiled in the following order (here this assumes using the Intel(R) Fortran compiler, but is straightforward to understand when using another compiler; i.e., if using MinGW, then replace the `ifort` command by `gfortran` and `.obj` files by `.o` files):

```
ifort -c globals.f90
ifort -c misc_functions.f90
ifort -c wage_function_old.f90
ifort -c wage_function_young.f90
ifort -c distribution.f90
ifort -c solve_model.f90
ifort -c results_transition.f90
ifort -c results_welfare.f90
ifort globals.obj misc_functions.obj wage_function_old.obj
wage_function_young.obj distribution.obj solve_model.obj
results_transition.obj results_welfare.obj main.f90 -o my_prog.exe
```

Computations reported in the paper were run with the Intel(R) Fortran compiler on a machine with 64 GBytes memory, and running 64-bit Windows 7 64-bit.

4 Model specification

In the specification file `params.txt`, there are a number of additional "free" parameters that allows to check some variants of the model. Specifically, `params.txt` should contain the following:

1. On line 1, `delta_sp` is an indicator that takes the value of 1 if workers are entitled to receive SP after the δ shock (our baseline assumption) and is equal to 0 otherwise;

2. On line 2, `young_sp` is an indicator that takes the value of 1 if young workers are entitled to purchase an annuity (our baseline assumption) and is equal to 0 otherwise;

3. On line 6, `leis` is the leisure utlity of retired workers (set to 0 in our model);

4. On line 13, `R` is the income that retired workers would receive until they die (set to 0 in our model, and we set the value of retiring from the labor market to 0);

5. On line 9, `nu` is the fraction of SP that are transferred to the worker after dismissal (1 is our baseline assumption).

The specification file `reform.txt` contains a single value encoded in `reform_sp`, which is an indicator that takes the value of 1 for the partially non-retroactive reform (our baseline assumption) and is equal to 0 for the statu-quo reform presented in the online appendix.