econo-ml: Reinforce Learning Policy Recommendation for Interbank Network Stability

1 Auxiliary files

- `requirements.txt`: list of the necessary python packages

2 Interbank model

- `interbank.py`: use to execute standalone the Interbank simulation.

  - It accepts command line options. For instance:
    ```
    interbank.py --log DEBUG --n 150 --t 2000
    interbank.py --save results.gdt --p 0.5 eta=0.35 param=X
    ```

  - When it is used as a package, the sequence should be:
    ```
    import interbank
    model = interbank.Model()
    model.config.configure(param=x)
    model.forward()
    eta = model.get_current_fitness()
    model.set_policy_recommendation(eta=0.5)
    ```

- Basic options:
    ```
    # To list all options:
    interbank.py --help

    # Using lender's change mechanism ShockedMarket3
    # with probability of attachment 0.3:
    interbank.py --lc ShockedMarket3 --p 0.3

    # Same for Preferential with m nodes:
    interbank.py --lc Preferential --m 0.3

    # To use a fastest algorithm to run in big simulations:
    interbank.py --fast

    # To run a simulation based on exp_runner:
    python -m experiments.exp_shockedmarket --do
    ```

- `colab_interbank.ipynb`: Notebook version of the standalone `interbank.py` with the same results but plotted using Bokeh.

- `interbank_lenderchange.py`: It contains the different algorithms that control the change of lender in the model.

- `exp_runner.py`: A prototype for executing experiments with different parameters and using MonteCarlo (using concurrent.futures to allow multiple threads).

- `exp_runner_distributed.py`: A sub-prototype that uses ray library to execute in a cluster.

- `exp_runner_no_concurrent.py`: Another sub-prototype that avoids the use of parallelism.

- `exp_runner_no_concurrent.py`: Another sub-prototype that avoids the use of parallelism.

- `exp_runner_comparer.py`: A derivation of the former prototype though to compare the evolution with `p` (probability of attachment in an Erdos-Renyi graph) in the `x` axis and other parameters accross the `y` axis.

- `exp_runner_surviving.py`: A derivation of the former prototype using ray library to execute in a cluster.

- `experiments/`: directory with all the experiments conducted. The results of that executions are stored in a folder determined inside each experiment.

- `utils/plot_psi.py`: Generate a table of axis_x x axis_y plots.

- `utils/labplot2_interbank.lml`: LabPlot2 file to plot the results of the `interbank.py`. By the way the best way is to use Gretl as an export format.

- `algorithm.drawio` and `algorithm.drawio.pf`: the draw.io and PDF schema of the algorithm used in the model to propagate shocks and to balance sheets.

3 RL with Stable Baselines3

- `interbank_agent.py`: agent to test using PPO

- `run_ppo.py`: run and simulate with PPO agent

- `run_td3.py`: run and simulate with TD3 algorithm

- `models/XXXX.zip`: instances of Gymnasium.env trained to use with `run_XXXX.py`

- `utils/plot_ppo.py`: auxiliary creator of plots to play the results of PPO

- Usage:

```
# train first and save the model env:
run_ppo.py --train ppo_10000 --t 10000 --verbose

# use the trained env and generate a simulation of T=1000
# with Interbank model
run_ppo.py --load ppo_10000 --save results_ppo.txt
```

4 Basic usage of the model

- `interbank.py --seed 1234 --t 500 --p 0.2`: Execute the model with $T = 500$ and *LenderChange* algorithm of *ShockedMarket*3 with an Erdös-Réni with probability of attachment of 0.2 and using a seed for generating random values of 1234 (same results if you generate again with other equal parameters and repeat this integer number for seed).

- `interbank.py --save file.gdt --log DEBUG --logfile file.txt`: Save the results in `file.gdt` in CSV and the detailed log in `file.txt`.

- `interbank.py --save file.gdt --stats_market --detail_banks 5,7`: Save the results in `file.gdt`, a second file `fileb.gdt` with the results for only banks and times participating really in the loans market is generated, and also a third file `file_detailed.gdt` with the concrete statistics for banks 5 and 7. With `--detail_times 10,12` all specific details for all banks in times 10 and 12 are present in this third file.

5 Statistics

Different statistics can be obtained after running the model, either in **csv** output, or in **gdt** (Gretl format). This statistics collect data in each time for the average or individually, depending on the usage. Possible statistics obtained from the model are:

- **active_borrowers**: Number of banks that are involved in a loan as borrowers. Both values in global and **stats_market** will be the same.

- **active_lenders**: Number of banks that are involved in a loan as borrowers. Both values in global and **stats_market** will be the same.

- **asset_i**: Assets of the lender of this bank $(D + E)$

- **asset_j**: Assets of the borrowers of this bank $(D + E)$

- **bad_debt**: Sum of the bad debt

- **bankruptcies**: Number of banks that failed in this step

- **bankrupcty_rationed**: Number of banks that failed in this step due to rationing

- **best_lender**: ID of the bank which more connections in the graph

- **best_lender_clients**: Number of banks connected with the best lender

- **c**: Lender capacity $(1 - \frac{E}{maxE})$ of the bank

- **communities**: Subsets of nodes with higher internal edge density than connections to the rest of the graph

- **communities_not_alone**: Number of **communities** that are not formed by only one node

- **deposits**: Deposits D in the balance $L + C + R = D + E$

- **equity**: Equity E of the bank: $L + C + R = D + E$

- **fitness**: Fitness (μ) of the bank

- **gcs**: When we use an Erdös–Rényi graph, the Giant Component Size is the largest number of nodes that are interconnected.

- **grade_avg**: Average number of edges (connections) for the total banks

- **incrementD**: Amount of (ΔD) for the bank

- **interest_rate**: Interest rate r of the bank

- **l_equity**: Log of equity $(log(E))$

- **leverage**: Financial leverage (l/E) of the bank considering only the banks that are inside a loan, named **leverage_** in Gretl due to name restrictions of the environment.

- **liquidity**: Total liquidity L of the Banks $L + C + R = D + E$

- **loans**: Amount borrowed by the bank

- **num_banks**: Number of banks currently surviving in the model (interesting when **allow_replacement_of_bankrupted=False**)

- **num_loans**: Num of loans in this step. Both global and **stats values _market** will be the same

- **num_of_rationed**: Number of banks that were rationed in this step (needed money and were without any possible lender)

- **policy**: Policy recommendation η of the system in the range [0..1]. As η is a global value, the same number applies for all banks.

- **potential_credit_channels**: Considering there is a graph of connections between banks, then **number_of_edges()** in the graph

- **potential_lenders**: Number of banks in the first shock having a possitive shock (ΔD)

- **prob_bankruptcy**: Probability of bankruptcy $p = \frac{E}{E_{max}}$, between $[0..1]$

- **profits**: Profits obtained in that step

- **psi**: Power market (psi) value $[0..1]$

- **rationing**: Total amounf of the loans l of the banks

- **real_t**: Times in which are no loans are removed in the extra statistics generated when we use –**stats_market**. Real t instants of time are stored in this variable to track when were really those values are obtained in the original statistics.

- **reserves**: Reserves R in the balance $L + C + R = D + E$

- **systemic_leverage**: Financial leverage but considering in the mean the total banks of the model N

The different statistics of information obtained in table **??** are classified as:

- Global: using –**save filename**: each data column in **filename.gdt** will be obtained for all the N banks in the model for all instants time T (rows)

- With –**stats_market** what we obtain will be a file as **filenameb.gdt** for the subsets of banks that in each time are engaged in a real loan. So if in the time t there are no loans, it is removed from this statistics. The special value `real_t` indicates which was the original time.

- Individual is data obtained when we use –**detail_times** or –**detail_banks** and it stores **filename_detailed.gdt** of those moments for all the banks individually or specific banks.

- Graphs are data obtained also in **filename.gdt** but only we have a **Lender-Change** algorithm with a random graph.
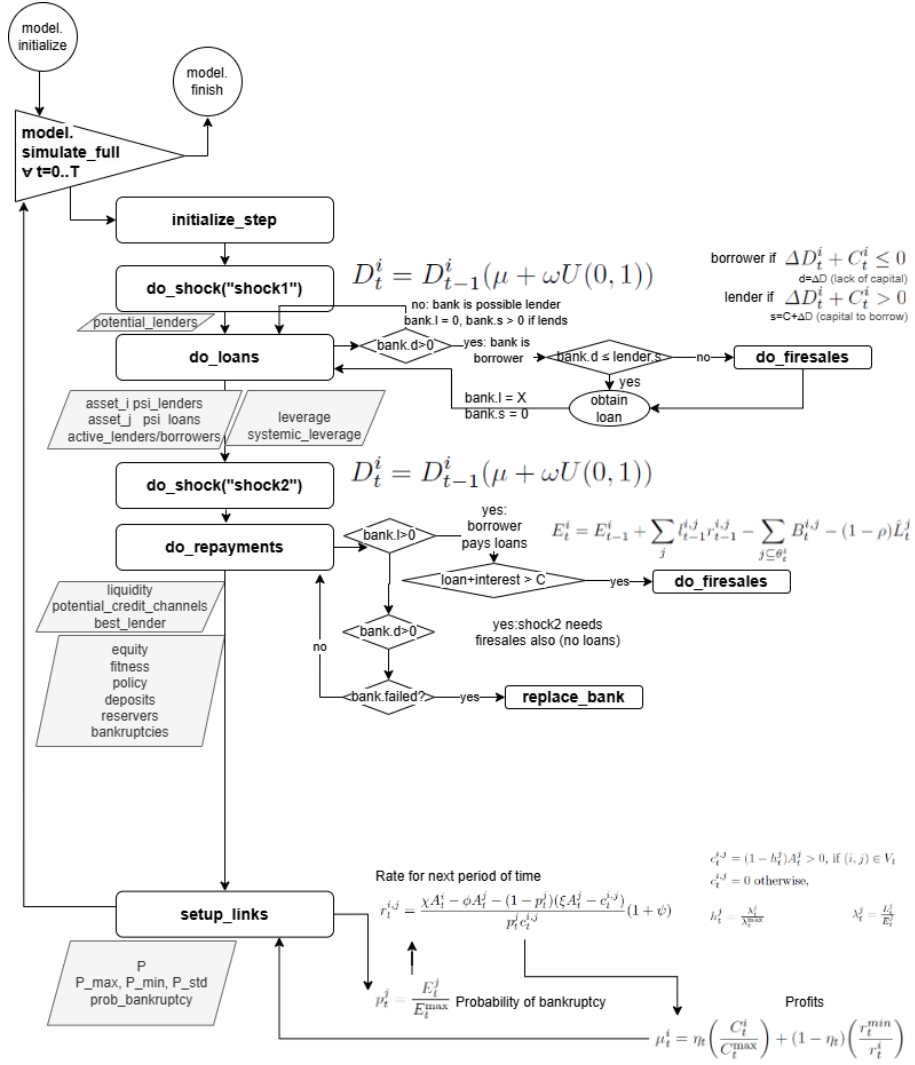
Figure 1: Sequence of steps: grey boxes indicates moments in which that statistic is obtained

Name	Type	Global	**stats_market**	Individual	Graphs
active_borrowers	integer	✓	✓		
active_lenders	integer	✓	✓		
asset_i	float	$\overline{x}/0$	$\overline{x}/0$	✓	
asset_j	float	$\overline{x}/0$	$\overline{x}/0$	✓	
bad_debt	float	$\sum$	$\sum$	✓	
bankruptcies	integer	$\sum$	$\sum$	✓	
bankrupcty_rationed	integer	$\sum$	$\sum$		
best_lender	integer	✓	✓		
best_lender_clients	integer	✓	✓		
c	float	$\overline{x}/nan$	$\overline{x}/nan$		
communities	integer				✓
communities_not_alone	integer				✓
deposits	float	$\sum$	$\sum$	✓	
equity	float	$\sum$	$\sum$		
fitness	float	$\overline{x}$	$\overline{x}/nan$	✓	
gcs	integer				✓
grade_avg	integer				✓
incrementD	float	$\sum$	$\sum$		✓
interest_rate	float	$\overline{x}/0$	$\overline{x}/nan$	✓	
l_equity	float	$\sum$	$\sum$		
leverage / leverage_	float	$\overline{x}$	$\overline{x}/nan$	✓	
liquidity	float	$\sum$	$\sum$	✓	
loans	float	$\sum$	$\sum$	✓	
num_banks	integer	✓	✓		
num_loans	integer	✓	✓	✓	
num_of_rationed	integer	✓	✓		
policy	float	✓	✓		
potential_credit_channels	integer	✓	✓		
potential_lenders	integer	✓	✓		
prob_bankruptcy	float	✓	✓	✓	
profits	float	$\sum$	$\sum$		✓
psi	float	✓/0	✓/nan		✓
rationing	float	$\sum$	$\sum$		✓
real_t	integer		✓		
reserves	float	$\sum$	$\sum$		✓
systemic_leverage	float	$\overline{x}$	$\overline{x}$		

Table 1: Legend for the different columns are: ✓=value without any modification. $\sum$=sum of the value for all banks. $\overline{x}$ = average of the value for all banks. 0 = No banks in this statistic. nan=Instead of zero, the value of "not a number" is used