Readme for *Steering the Climate System: Using Inertia to Lower the Cost of Policy*

Derek Lemoine and Ivan Rudik

This code is written in MATLAB. As is, the model replicates results found in the paper but parameters can be changed to explore other settings. Initial guesses for the solver are set near the solution to reduce run time.

Instructions

The main file is `MAIN_SCRIPT.m`. The user selects which models to run and which solver to use by utilizing the `logic` structure. Setting `logic.knitro = 1` will use KNITRO to search over terminal conditions and setting it to 0 will use fmincon. At the top of the file, the user must indicate the location of the KNITRO options file using `options_file` if KNITRO is the chosen solver, otherwise leave it as is. Next, the user can select which type of model to run. The combinations to replicate the three types of runs in the paper are:

1. Main text / base model: `logic.stat_ems = 1` and `logic.ghkt = 0`
2. Appendix / Golosov, Hassler, Krusell, and Tsyvinski (2014) decay structure: `logic.stat_ems = 1` and `logic.ghkt = 1`
3. Appendix / Non-stationary emissions: `logic.stat_ems = 0` and `logic.ghkt = 0`

Setting `logic.stat_ems = 0` will run the model with non-stationary emissions. `logic.solve_nonstat = 1` will solve the non-stationary emissions problems if `logic.stat_ems = 0`. We put this option in since the solve is extremely long. If `logic.solve_nonstat = 0` the code will skip straight to simulation from a pre-stored solution. Setting `logic.ghkt = 1` will alter the carbon decay structure to match that of Golosov, Hassler, Krusell, and Tsyvinski (2014).

`logic.hotelling` determines whether inertia will be accounted for when optimizing policy. Setting `logic.hotelling = 0` accounts for inertia, while setting `logic.hotelling = 1` does not and the program will find the least cost path to satisfy the "equivalent" constraint on CO_2 concentrations. When running the base model, setting `logic.vary_params = 1` will loop and solve several versions of the model: ones using an inertia parameter that is 25% higher and lower, and then for a discount rate of 1.4% from Stern (2007). Note two additional things for model selection:

1. To perform a non-stationary emissions run, you **must** use the base decay structure, and to use the GHKT decay structure, you **must** have stationary emissions.

2. You can only use the code to automatically vary discounting and the level of inertia for the base model in the main text. However, alternative parameterizations for the GHKT or non-stationary emissions models can be explored by hard coding changes to the parameter values.

Finally, the user must determine which temperature targets to solve for. `params.first_temp` and `params.final_temp` determine the lowest and highest temperature target to use, and `params.temp_incr` determines the size of the increment in the temperature target as the code loops from the lowest to the highest target.

Results

`calculate_results_XXXX.m` where `XXXX` is either `ghkt` or `non_ghkt` calculates the final results for the GHKT and base/non-stationary models found in the paper. The results displayed in the paper are saved in a structure called `output`. The script saves the results to model-specific folders with file names that indicate the temperature target (or temperature target equivalent); whether it is a Hotelling run or not; and in the case of the base model, the value of the inertia parameter and the discount rate.

Solver tolerances

There are several tolerances the user can adjust to alter accuracy and speed. Within the `ode_....` files are ode tolerance options. Settings for relative and absolute tolerances below 10^{-11} will replicate the paper. Coarser tolerances will be quicker to solve but at the expense of some small error. Within `initialize_parameters` is a variable `params.grid_res` that defines the resolution of the ODE solution mesh in units of years. A finer mesh (smaller number) will result in a more accurate solution however compute time increases rapidly in the mesh resolution. Generally values of .01 or .005 can be used to obtain an accurate solution in reasonable time.

File list

- Main files
  - MAIN_SCRIPT
- Initialization files called by MAIN_SCRIPT
  - load_initial_base
  - initialize_parameters
  - initialize_solver
- Files called by MAIN_SCRIPT that search over terminal conditions
  - parameter_search_base
  - parameter_search_ghkt
  - parameter_search_nonstat
- ODE solver files called by search files
  - ode_base_to_constraint
  - ode_base_constrained
  - ode_base_to_init
  - ode_ghkt_to_constraint
  - ode_ghkt_constrained
  - ode_ghkt_to_init
  - ode_nonstat_to_constraint
  - ode_nonstat_constrained
  - ode_nonstat_to_init
- System of ODEs files called by ODE solver files
  - state_path_base
  - state_path_base_constr
  - state_path_base_hotel
  - state_path_ghkt
  - state_path_ghkt_constr
  - state_path_ghkt_hotel
  - state_path_ghkt_constr_hotel
  - state_path_nonstat
  - state_path_nonstat_constr

  - state_path_nonstat_hotel

- Event files called by ODE solver files for when hitting constraints or initial time
  - event_fullabate
  - event_M1
  - event_M
  - event_mu0
  - event_T0
  - event_time0

- Files called by MAIN_SCRIPT to simulate state trajectories for final results calculation
  - simulate_trajectories_base
  - simulate_trajectories_ghkt
  - simulate_trajectories_nonstat

- Files called by MAIN_SCRIPT to calculate final results
  - calculate_results_non_ghkt
  - calculate_results_ghkt