

Codes and Data for “Dynamic Portfolio Allocation under Market Incompleteness and Wealth Effects” (OPRE-2024-04-0976)

This file summarizes the codes and data for the paper “Dynamic Portfolio Allocation under Market Incompleteness and Wealth Effects”. Section 1 documents the corresponding codes to generate the numerical results (figures and tables) in the paper. Section 2 provides description for other codes/data used in the comparative analysis of wealth effects in the paper. Section 3 provides description of the codes and data used in the parameter calibration.

1. Codes and data for generating the numerical results in the paper

Numerical results in the main text:

1. Figure 1: Optimal policy with respect to wealth level
Code: static_impact_cirsv.m
Data: 'params.mat' (model parameters)
2. Figure 2: Cycle-dependence of HARA/CRRA policy and equivalent risk-aversion level
Code: plot_cycle_dependence_231028.m
Data: 'market_path4.mat' (the representative simulated path of market dynamics);
'params.mat' (model parameters)
3. Figure 3: Regression coefficients for HARA investors with different wealth levels
Code: cycle_dependence_regression.m
Data: 'params.mat' (model parameters)
4. Figure 4: Performance statistics of HARA optimal portfolio by different wealth levels
Code: performance_initial_wealth_plot.m
Data: 'params.mat' (model parameters)

Numerical results in the electronic companion:

1. Figure EC.1 Optimal policy with respect to investment horizon
Code: static_impact_cirsv.m
Data: 'params.mat' (model parameters)
2. Figure EC.2: Simulated bond price and bond weight
Code: plot_cycle_dependence_231028.m
Data: 'market_path4.mat' (the representative simulated path of market dynamics);
'params.mat' (model parameters)
3. Figure EC.3: Optimal stock weights and risk aversion level of high and low wealth HARA investors
Code: plot_cycle_het_investor_231029.m

- Data: 'market_path4.mat' (the representative simulated path of market dynamics);
'params.mat' (model parameters)
4. Figure EC.4: Average and Std of the weighted stock return by initial wealth level
Code: performance_initial_wealth_plot.m
Data: 'params.mat' (model parameters)
 5. Figure EC.5: Representative quantiles of normalized optimal stock weights
Code: cycle_dependence_regression.m
Data: 'params.mat' (model parameters)
 6. Table EC.1: statistical test for difference in performance metrics by wealth level
Code: performance_initial_wealth_plot.m
Data: 'params.mat' (model parameters)

2. Brief description of other functions and data used in the paper

Functions

1. dv.m dr.m: function for solving hedge component in CRRA optimal policy.
2. B.m: calculates the bond price given time to expiration, used in solving HARA optimal function
3. opt_HestonCIR.m: given model parameters and investment horizon, return the optimal weight for bond and stock.
4. generate_stvtrt.m: given model parameters, generates the path of interest rate, bond price, and stock price.
5. simu_pi_paths.m: based on generate_stvtrt.m and opt_HestonCIR.m, simulates an HARA investor's optimal policy and investment performance metrics for each path parallelly
6. generate_bull_bear_path_231029.m: calls generate_stvtrt.m to generate paths parallelly; choose and save a specific path with multiple bull and bear periods
7. bull_bear_classify.m: given the path of stock path and criteria percentages, return a 0-1 vector that represents whether a time point is in bull period.
8. draw_bear_shadow.m: draw shadow on the plot when it is a bear period.
9. record_path_points.m: generates a sample of multiple possible market paths and corresponding wealth paths, sampling data points each season.
10. performance_initial_wealth_simulation.m: iterate through different initial wealth, in each iteration call function simu_pi_paths to get the investment performance for HARA investors with each wealth level

Data

1. params.mat: calibrated model parameters
2. market_path4.mat: a representative market path used in plot
3. performance_CIRSV_23_0928.mat: stores simulated performance metrics of investors of different wealth (this can be generated by performance_initial_wealth_plot.m)

4. `reg_results.mat`: stores regression results (this can be generated by `cycle_dependence_regression.m`)

3. Brief description of functions and data for parameter calibration

The codes and data for calibration are included in the “calibration codes” folder

Functions

1. `MLE.m`: the main function to read in data and perform mle estimation using negative log likelihood; return parameter estimation and se; plot paths of $StVtrt$ and check local minimization
2. `log_likelihood.m`: iterate through periods and sum up log transition density and log likelihood of bond yield observation error to get the total (negative) log likelihood
3. `ltransition_density.m`: calculate log transition density of $(\log(St), VIXt^2, P_{\{t,T\}})$; sum up the log transition density of Z and the Jacobian term
4. `get_interest_rate.m`: given observed bond price and parameters, compute the underlying interest rate and log Jacobian term from bond price to interest rate; apply a lower bound of 0.01% for the interest rate
5. `int_vol.m`: given observed VIX and parameters, compute underlying volatility and log Jacobian term from VIX to volatility; apply a lower bound of 0.1% for the underlying volatility
6. `local_optim_plot`: produce local optimization plot; slightly change one parameter and compute the new log likelihood; this validates our estimation results; see the figure `local_optim_plot_230916.pdf`
7. `MCtest.m`: test codes for MLE with simulated paths to validate the estimation method

Data:

1. `data.csv`: the data used for calibration in `MLE.m`. The columns from left to right are: date, one year bond yield, SP500 index, square of VIX index, two-year bond yield, and five-year bond yield. The data ranges from 2013/4/29 to 2019/12/31.

Supporting package:

1. `DERIVESTsuite`: used to calculate Hessian matrix