**JASA ACS Reproducibility Initiative**

## Data

Data consists of 6 macroeconomic time series data from the FRED database. The 6 series are monthly inflation, wage, unemployment, consumption, investment, and interest rate from 1986/1-2015/12. All data are publicly available from the FRED database:

Inflation: <https://fred.stlouisfed.org/series/CPILFENS>

Wage: <https://fred.stlouisfed.org/series/CEU0500000008>

Unemployment: <https://fred.stlouisfed.org/series/UNRATE>

Consumption: <https://fred.stlouisfed.org/series/PCE>

Investment: <https://fred.stlouisfed.org/series/NAPMNOI>

Interest Rate: <https://fred.stlouisfed.org/series/FEDFUNDS>

## Code

The zip file contains the Matlab file and function that runs multivariate BPS, with the agent forecasts included as well for the 1-step ahead forecasts examined in the paper.

The function inputs the agent forecast distribution information and prior specifications, and outputs the forecast coefficients and variances as well as the posterior smoothed coefficients and agent densities. The code follows the synthesis function specification of the paper.

BPSsim.m is a file that loads the agent forecast densities and calls the mBPS.m function (where mBPS(y,a\_j,A\_j,n\_j,delta,m\_0,C\_0,n\_0,s\_0,burn\_in,mcmc\_iter) produces the posterior parameters needed to forecast and analyze) sequentially, outputting the predictive distributions and then computing the performance measures.

## Instructions

Running BPSsim.m computes the predictive performances for the 1-step ahead forecasts. The BPS outputs are computed within the mBPS function (i.e. running mBPS(y,a\_j,A\_j,n\_j,delta,m\_0,C\_0,n\_0,s\_0,burn\_in,mcmc\_iter) produces the posterior parameters needed to forecast and analyze). MSFE and LPDR are then computed outside the function to compare with the results of the paper (Table 1 and Figures 2-3). On-line coefficients and retrospective posteriors for the coefficients and the agents are also saved (which is used to produce Figures 4-8).

Using Matlab on Intel Core i7-7700k CPU @4.20 GHz, each iteration (t) takes on average approximately 5 minutes, taking approximately 30 hours for the whole process to complete. Each iteration (t) can be parallelized for faster computation.

## Notes

Codes is available publicly on the personal website for lead author Ken McAlinn