REPLICATION FILES

"Inflation and Professional Forecast Dynamics: An Evaluation of Stickiness, Persistence, and Volatility"

accepted for publication by Quantitative Economics.

Authors

- Elmar Mertens, (Deutsche Bundesbank, em@elmarmertens.com, www.elmarmertens.com)
- James M. Nason, (NC State University, jmnason@ncsu.edu, www.jamesmnason.net)

Overview

This repository provides replication files for our paper and its online appendices. The replication files contains code as well as raw copes of our input data as obtained from their original sources described further below. The repository is hosted at https://github.com/elmarmertens/MertensNasonQEstickyinformation/ which also provides a working-paper version of our paper and its appendices.

Our code consists of programs written for Matlab, and FORTRAN. The main computations are done in FORTRAN, Matlab code has been used to process results, create charts, and prepare the data inputs. We also provide bash scripts and GNU makefiles to compile and execute parts of the code. Except for the Matlab scripts, our instructions below refer to commands to be issued on a linux

(or macOS) command line with the working directory set to the folder within which the corresponding script has been placed. The Matlab scripts are to be executed from a Matlab command line or the Matlab GUI.

The code has been run in different environments with essentially identical results: Intel FORTRAN (incl. MKL) version 19.0 on macOS Mojave and Catalina as well as ubuntu linux (Version 18.04). (During the development of the project, the code has also been run, with success, on Intel FORTRAN 18, as well as 19.1.). In addition, for pre- and post-processing data and results we used scripts running on Matlab versions 2018a/b as well as 2019a/b and 2020a. Our FORTRAN codes uses OpenMP for parallelizations; typically we used 16 workers. (Note that some of our codes, notably the particle smoothers and MDD computations, employ separate random number streams for each worker. Varying the number of workers thus results in differences in random numbers generated during the computations.)

The repository contains the following directories.

- `main`: particle learning filters and smoothers for the state space models described in the paper and its appendices
- `dataconstruction`: raw data files and Matlab scripts to compile the data set into input files used in `main`
- `ucsv`: particle learning filter applied to the Stock and Watson (2007, JMCB) UCSV model to produce results shown in Section R.6 of our online results appendix
- `montecarlo`: code to simulate data and the application of the particle learning filter as discussed in Section R.8 of our online results appendix
- `matlabbox`: Matlab helper routines, available via github from Elmar Mertens at https://github.com/elmarmertens/em-matlabbox (The Matlab scripts described below modify the Matlab path to include the contents of this toolbox directory.)
- `fortranbox`: a set of auxiliary FORTRAN modules, available via github from Elmar Mertens at https://github.com/elmarmertens/em-fortranbox (The

makefiles employed below automatically refer to this toolbox directory.)

The remainder of this README describes the contents of each directory (except for the toolboxes).

Data

The folder `dataconstruction` contains the raw data, as obtained from the websites of the Federal Reserve Banks of Philadelphia and St. Louis, and Matlab scripts that transform the raw data into input files for our computational routines. The data set used in our paper includes measures of realized inflation and SPF forecasts based on the GDP/GNP deflator; the data set is denoted "GDPD". Our online results appendix provides robustness checks based on CPI inflation. The folder `dataconstruction` contains material to reproduce both data sets.

For the GDPD data set, we obtained the following Excel files from the Federal Reserve Bank of Philadelphia:

- Average ("mean") SPF expectations for the GDP/GNP deflator available at https://www.phil.frb.org/research-and-data/real-time-center/survey-of-professional-forecasters/data-files/files/Mean_PGDP_Level.xls
- Second-revision data on quarterly growth rates in the GNP/GDP deflator available at https://www.philadelphiafed.org/-/media/research-and-data/real-time-center/real-time-data/data-files/files/xlsx/p_first_second_third.xlsx?la=en

The GDPD data was downloaded on February 16, 2019. The Matlab script `cambridgeDataGDPD.m` transforms the data contained in the above-mentioned Excel files as needed for our paper, and prepares input files for the computational routines in the `main` folder described further below. The input files are:

- `cambridge2018GDPD.yData.txt` containing a matrix of quarterly observations on realized inflation, the SPF nowcast and forecasts for 1,2,3,4 quarters ahead

(including the nowcast, these are `H=5` SPF predictions). The matrix provided by the data file has `T=201` rows and `H+1=6` columns. Missing values are denoted as zeros. (Following the timing assumption described in the paper, SPF predictions collected in the middle of a given quarter are treated as forecasts made at the end of the previous quarter.)

- `cambridge2018GDPD.yNaN.txt` provides a `T x (H+1)` matrix of zero and ones; an entry of 1 indicates a missing observation in the `yData` file described above and zero otherwise.
- `cambridge2018GDPD.dates.txt` provides a date vector in Matlab format used for the creation of charts when post-processing the estimation results.

For the CPI data set, we obtained the following files from the Federal Reserve Banks of Philadelphia and St. Louis:

- Average ("mean") SPF expectations for CPI inflation available as Excel file from teh Federal Reserve Bank of Philadelphia at https://www.philadelphiafed.org/-/media/research-and-data/real-time-center/survey-of-professional-forecasters/data-files/files/mean_cpi_level.xlsx?la=en
- A text file in `csv` format of final-vintage headline CPI level data available from the FRED databse at the Federal Reserve Bank of St. Louis at https://fred.stlouisfed.org/series/CPIAUCSL

The CPI data was downloaded on February 17, 2019. The Matlab script `cambridgeDataCPI.m` transforms the data as needed and prepares input files for the computational routines in the `main` folder. The input files are named analogously to the case of the GDPD data set with file names referring to "CPI" in lieu of "GDPD."

Particle learning filters and smoothers

The folder `main` contains FORTRAN code to apply particle-learning filters and smoothers to estimate the four state space models considered in our paper. The folder contains copies of the ASCII data files that were created by the code contained in `dataconstruction` as described above. In addition, the `main` folder provides Matlab scripts for post-processing the results, as well as Bash scripts and GNU makefiles to compile and launch the FORTRAN drivers.

The four state space models differ in whether the parameters `theta` (persistence of the inflation gap) and `lambda` (sticky-information weight) are considered to be constant or time-varying; see also Table 1 of the paper. For each model, there is a separate FORTRAN driver file called `cambridgeSIthetaXXXlambdaYYY.f90` with the expressions `CONST` or `TVP` used in place of `XXX` and `YYY` depending on the model. For example, for model M0, where `theta` is constant and `lambda` time-varying, the driver file is called `cambridgeSIthetaCONSTlambdaTVP.f90` and so on.

To compile and execute each driver, use `make run THIS=XXX` with `XXX` replaced by the name of the driver file (omitting the `.f90` suffix). Compilation and execution of the code will be based on default values encoded in the `makefile` that can be changed there (or via the command line when invoking `make`). To replicate all model estimates as reported in the paper for the GDPD dataset, use the bash script `processmodels.sh`, which also illustrates the use of command line arguments when calling `make`. The script `processmodelsCPI.sh` performs the corresponding computations for the CPI data set. Both scripts, `processmodels.sh` and `processmodelsCPI.sh`, also produce output for versions of the state space models that omit a noise component from the inflation process, as described in Section R.2.2 of the online results appendix.

For the computation of marginal data densities (MDD), as reported in Table 5 of the paper, and Tables R.3, R.4 and R.5 of the online results appendix, we used 250

repetitions of each particle-learnign filter (further details described in the paper). For better computational efficiency, these estimates are performed by separate driver files, that parallelize the computations across the repeated runs of the particle filter, rather than across the particles of each filter. These driver files are named `cambridgeSIthetaXXXlambdaYYYstderr.f90` with the expressions `CONST` or `TVP` used in place of `XXX` and `YYY` depending on the model. To compile and execute the drivers for each state space model, use the Bash scripts `processstderrs.sh` (for the GDPD data set) and `processstderrsCPI.sh`.

The FORTRAN drivers produce output in the form of various ASCII files, named with the suffix `.dat`. The content of these output files is converted into charts by the following Matlab scripts:

- `chartsPanelsCambridgeSIbaselinefigures.m` creates all figures shown in the paper, except for the panels of Figure 7, which are created by `chartsLambdaSmootherSIMSE.m`
- `chartsCambridgeSImodelfigures.m` creates figures for each model and each data set as reported in the online results appendix.
- `chartsMDD.m` collects MDD estimates for use in tables of posterior estimates (as described in the next bullet) and generates charts shown in Section R.2.3 of the online results appendix. The scripts stores MDD estimates in temporary Matlab data files in `.mat` format for further processes by `tabulatePosteriorParams.m` described below.
- `chartMDDtimeseries.m` creates time series plots of the log MDDs shown in Section R.2.3 of the online results appendix as well.
- To tabulate posterior moments of parameter and MDD estimates, as shown in Table 5 of the paper (as well as Tables R.3, R.4 and R.5 of the online results appendix), use `collectParametersCambridgeSI.m`, which collects the posterior moments from the FORTRAN output files and stores those in Matlab `.mat` files, and then `tabulatePosteriorParams.m`.
- `showTermstructuresForecast.m` creates charts of the term structures of

inflation forecasts as discussed in Section R.1.1 of the online results appendix.

- `tabulateRMSE.m` creates forecast comparison tables shown in Section R.1.2 of the online results appendix.

Section R.3 of the online results appendix also considers an alternative specifcation for the process of a time-varying sticky information weight `lambda` in the M2 model. The driver file for estimating this model variant is called `cambridgeSIthetaCONSTlambdaTVPnormcdf.f90` and can be compiled and executed via the Bash script `processM2normcdf.sh`. To generate charts from the estimation output, please use the Matlab script `chartsCambridgeSIm2normcdf.m`.

Monte Carlo simulations

The folder `montecarlo` contains code to perform the Monte Carlo simulations of our estimators that are described in Section R.8 of the online results appendix. We perform two kinds of simulations, each with a separate FORTRAN driver file:

- `cambridgesimSIthetaTVPlambdaTVP.f90` applies the particle learning filter of the "M2" model (with time-varying parameters theta and lambda) to simulated samples of data to measure the bias in estimates of latent states and parameters as described in Section R.8.1. Instructions for compiling and running the driver file are encoded in the GNU makefile `sim.makefile`. To perform the simulations as conducted for the paper use `make -f sim.makefile run`. The Matlab script `showCambridgeSIsimgridBias.m` generates plots of the results, as shown in the appendix. The script assumes that results generated by `cambridgesimSIthetaTVPlambdaTVP.f90` are stored in the the current directory; otherwise please adapt line 43 of `showCambridgeSIsimgridBias.m` so that the string variable `datadir` points to the appropriate directory. The script `showSimdata.m` can be used to plot details of the simulated data.

- `cambridgesimMDD.f90` applies the particle learning filter of the "M2" model (with time-varying parameters theta and lambda) to simulated samples of data to measure the MDD detection rates as described in Section R.8.2. Instructions for compiling and running the driver file are encoded in the GNU makefile `mdd.makefile`. To perform the simulations as conducted for the paper use the bash script `doMDD.sh`. (The script sets the environment variable OMP_NUM_THREADS to equal the number of parallel workers available; please adapt this value as needed.) The Matlab script `showCambridgeSIsimgridMDD.m` generates plots of the results, as shown in the appendix. The script assumes that results generated by `cambridgesimMDD.f90` are stored in the the current directory; otherwise please adapt line 18 of `showCambridgeSIsimgridMDD.m` so that the string variable `datadir` points to the appropriate directory.

Section R.8.2 also describes simulations from an environment where measurement is small. To reproduce these results, change the value of the variable `noisevol0` in `cambridgesimMDD.f90` by commenting out the baseline setting in line 125 of the code and commenting in line 126. Afterwards, recompile and rerun the simulations via `doMDD.sh`.

Particle-learning filter for Stock-Watson UCSV model

The folder `ucsv` contains code for a particle-learning filter estimation of the Sotck-Watson (2007, JMCB) UCSV model. The model uses only realized inflation data as inputs. The code thus operates just on the first column of the data input files constructed for our paper (and ignores the SPF data in subsequent columns). The particle-learning filter of the UCSV model is encoded in the FORTRAN file `ucsvPL.f90`. To compile and execute the FORTRAN code use `make run`. To inspect the results, use the Matlab script `showUCSV.m`.