REPLICATION FILES

"A Time Series Model of Interest Rates With the Effective Lower Bound"

This readme file describes the set of replication files (“the replication set“) for “A Time Series Model of Interest RatesWith the Effective Lower Bound“ forthcoming in the Journal of Money, Credit, and Banking. The replication set contains code as well as all of our input data in raw form as obtained from their original sources described further below.

Authors

- Benjamin K. Johannsen, (Federal Reserve Board) [1]
- Elmar Mertens, (Deutsche Bundesbank)[2]

Overview

The replication set comes in the form of this *readme* as well as the zip file *JohannsenMertensJMCBtimeseriesELB.zip* (henceforth "the zip file"). The zip file comprises the contents of several directories with code and data for this project; use of its contents is described further below after a description of our data sources.

Our code consists of programs written for R, Matlab, and FORTRAN. The main computations are done in FORTRAN, matlab code has been used to process results, create charts, and an R script has been used to prepare the data inputs. We also provide bash scripts and GNU makefiles to compile and execute parts of the code. In addition, we used the matlab codes from Wu and Xia (2016, JMCB) to generate out-of-sample forecasts from their model.

The code has been run in different environments with (nearly) identical results: Intel FORTRAN (incl. MKL) versions 18 and 19 on ubuntu linux (Version 18.04) and macOS Mojave. Matlab versions 2018a/b as well as 2019a/b. Our FORTRAN codes uses OpenMP for parallelizations; typically we used four workers per MCMC run and 16 for the IRF computations. Further gain from parallelization can be achieved by running different instances of our code simultaneously (for example, estimates using different data samples and/or model specifications).

Some of the code needed to do the comparison between our model and the model of Wu and Xia (2016, JMCB) requires Matlab on Windows.

Our main programs and data are contained in the main folder of the zip file. In addition, the zip file contains the following folders:

- toolbox: fortran toolboxes
- matlab toolbox: subdirectories that contain various matlab helper functions
- wuxiaCode: replication codes for the Wu and Xia (2016, JMCB) model as well as data and driver files adapted to our application

Data

Our main data source is the St. Louis Fed's FRED database available at https://fred.stlouisfed.org. As described in a separate section below, our forecast comparison against output from the Wu-Xia model and the SPF relies also on yield-curve data obtained from the Board of Governors and SPF responses collected by the Philadelphia Fed.

The R script `updatedata.R` downloads data files from FRED and produces ASCII data files that serve as inputs for our FORTRAN code. As described in footnote 22 of our paper, the data files used are PCECTPI (headline PCE inflation), TB3MS, GS2, GS5, and GS10 (yields on 3m Tbill and longer-term Treasuries), GDPC1 and GDPPOT (real GDP and the CBO potential), as well as UNRATE and NROU (unemployment rate and CBO's long-term NAIRU).

MCMC estimation of main model

Full-sample estimates

MCMC estimates of our baseline model and its variants are generated by the following FOTRAN driver files:

- `spectre2018.f90` for the baseline model with SV in all shocks, except the real-rate-trend shock
- `spectre2018rbarSV.f90` for the model its SV in *all* shocks
- `spectre2018constvar.f90` for the model *without* any SV
- `nomas2018.f90` for the model *with* SV in all shocks (except for the real-rate trend) but without use of longer-term yields

All of theses driver files can compiled and executed by a GNU make file `makefile`. The necessary steps to compile and execute the various models are listed in the bash script `mcmcpak.sh`. (To obtain merely estimates from our baseline model, the bash script `mcmcbaseline.sh` can be used as well.)

As shown in our bash scripts, runtime behavior of these driver files can be controlled by the use of command line arguments. For example, for our baseline driver, `spectre2018.f90` the command line arguments are: `spectre2018 ZLBrejection doPDF doPInoise doZLBasZeroes T T0 gapOrderKey datalabel Nyield p Nsim burnin Nstreams` where

- ZLBrejection: set to 1 to perform shadow-rate sampling, set to 0 to treat policy rate as missing data at ELB
- doPDF: set to 1 to computes predictive densities (at end of estimation sample)
- doPInoise: adds a noise component (with SV) to the inflation process (deprecated)
- doZLBasZeroes: set to 1 to treat missing values for policy rate at ELB as data values equal to the ELB
- T: integer value for last observation in sample (default: last observation in data file, for example, 236 in case of 2018Q4 data set)
- T0: jump-off for estimates (default: set to 0 to use first observation in data file)

- gapOrderKey: numeric key for ordering of gap variables, where the values 1,2,3,4 designate the following variables: 1 inflation, 2 activity gap, 3 policy rate, 4 block of long term yields. For example, 1234 orders inflation, activity gap, policy rate and long-term yields whereas 1324 uses inflation, policy rate, activity gap and long-term yields
- datalabel: Label of input data files (XXX.yData.txt for data values and XXX.yNaN.txt with 0/1 values marking location of missing data where XXX stands for the datalabel)
- Nyield: Number of long-term yields in data set
- p: Number of lags in Gap VAR
- Nsim: Number of MCMC draws post burnin
- burnin: Number of MCMC draws for burnin
- Nstreams: Number of separate MCMC streams used for assessing convergence (Default: environment variable `OMP_NUM_THREADS`)

All driver files output progress reports and a summary of convergence tests to standard output. Our bash scripts collect these outputs into log files.

To produce various outputs based on the full-sample estimates, please execute the following scripts in matlab:

- `figuresTRENDS.m` to plot trends estimates for inflation, the real-rate as well as stochastic volatility in the real-rate gap (Figures 3 of the paper as well as Figure S.1 of the supplementary appendix.)
- `figuresSV.m` to plot stochastic volatility estimates for trends and gaps (Figure S.2 of the supplementary appendix)
- `figuresandtablesPosteriormoments.m` generates tables and figures of priors and posteriors for various parameters as shown in Section III.2 of the supplementary appendix.

MDD computation

MDD estimates of our baseline model and its variants are generated by the following FORTRAN driver files:

- `spectre2018mdd.f90` for the baseline model with SV in all shocks, except the

real-rate-trend shock
- `spectre2018rbarSVmdd.f90` for the model its SV in *all* shocks
- `spectre2018constvarmdd.f90` for the model *without* any SV

All of these driver files can be compiled and executed by a GNU make file `makefile`. The necessary steps to compile and execute the various models are listed in the bash script `mddpak.sh`. Note that for each model we estimate the MDD using 10 different seeds of the random number generator. We report the average and the standard deviation across each estimation. The values are taken from the log files from each estimation.

Quasi-real-time estimates

To produce quasi-real-time estimates from our model, we estimate the model using data up to each date. Our FORTRAN driver files that are used to produce the full sample estimates accept an argument that indicates the final date of data to use in estimation. The necessary steps to compile and execute the various models are listed in the bash script `mcmcqrtpak.sh`. Figures that are produced from the quasi-real-time and full sample data for the real rate trend and the shadow rate are produced using `realratetrend_charts.m`, `realratetrend_charts_alt_models.m`, `shadowrate_charts.m`, and `shadowrate_charts_alt_models.m`. The figures including using the interquartile range or forecasts are produced by `iqr.m`. Before running any of these scripts, the function defined by `datadirfn.m` should be modified to return the location of the output files produced when computing the quasi-real-time estimates.

IRF estimates

The main fortran file used to compute impulse-response functions (IRF) is `particlefilterRollingstoneELBIRFchol.f90` (henceforth "the FORTRAN driver") and it can be compiled with `irf.makefile`. Execution of this code assumes that parameter estimates of our baseline model are stored in a subdirectory called `datMCMC`. To change this setting, please adapt the value of `mcmcdir` on line 29 in the FORTRAN driver.

Compilation and execution of the FORTRAN driver for our baseline model (using either output or unemployment gap) is performed by the bash script `irfpak.sh`. When the IRF computations are done, charts can be produced with the matlab script `figuresIRF.m` (Figures 6, 7 and 8 of the paper).

Comparison of forecasts against SPF and the Wu-Xia model

To replicate the forecast comparison of our model against Wu-Xia, the SPF and a random walk, please perform the following steps.

1. Produce the quasi-real-time estimates from our model as explained above.
2. Downloaded feds200628.xls from https://www.federalreserve.gov/pubs/feds/2006/200628/200628abs.html. (or unzip the version included with our replication files).
3. Downloaded WX.zip from https://sites.google.com/view/jingcynthiawu/.
4. unzip WX.zip.
5. copy feds200628.xls to ./WX/data (this overwrites an older version that is included in WX.zip).
6. Change working directory to ./WX.
7. Run the following matlab lines (Windows).

```
>> addpath(genpath('./functions'));
>> load ./parameters/maturities.mat;
>> copyfile ./parameters/maturities.mat ./
>> copyfile ./parameters/parameters.mat ./
>> [time,forwardrates] =
genmonthlyforwardrate('./data/feds200628',maturities);
>> save data_forwardrates.mat time forwardrates;
```

1. Copy wrapper_srtsm.m to ./WX
2. Call 'wrapper_srtsm.m'. The argument to the function is the ending month of the sample. It is passed as a string that can be parsed into YYMM format, where '1' is January 2000 and '1206' is June 2012. You cannot call the function for an earlier period than January 2000 without modification. The call to wrapper_srtsm save the output of the function call to a .mat file with a name of

the form ['RealTimeEstimates',enddate,'.mat']. To call all of the estimations, do the following:

```
>> for yy = 9:18
>>     for mm = 1:12
>>         yymm = num2str(yy*100 + mm);
>>         wrapper_srtsm(yymm);
>>     end
>> end
```

1. Change working directory to wuxiaCode.
2. Download the spacial econometrics toolbox from https://www.spatial-econometrics.com/html/download.html
3. Unzip the file (named jplv7.zip) in the wuxiaCode directory.
4. Run xlsdata.m in matlab (Windows).
5. Download the "Mean Forecast Data for Levels" from the Survey of professional forecasters from https://www.philadelphiafed.org/research-and-data/real-time-center/survey-of-professional-forecasters/historical-data/mean-forecasts.
6. Save the sheet named `TBILL` as a csv file named `meanLevel_TBILL.csv` in the wuxiaCode directory.
7. Save the sheet named `TBOND` as a csv file named `meanLevel_TBOND.csv` in the wuxiaCode directory.
8. Set 'rootdir' in jm_vs_rw.m, jm_vs_spf.m, and jm_vs_wx.m to the location where the output from the model is stored.
9. Run jm_vs_rw.m, jm_vs_spf.m, and jm_vs_wx.m to create the tables.

---

1. benjamin.k.johannsen@frb.gov
2. em@elmarmertens.com