DynareR Manual: Version 1

Sagiru Mati

2020-06-25

About the Author

The author of this package, **Sagiru Mati**, obtained his PhD in Economics from the Near East University, North Cyprus. He works at the Department of Economics, Yusuf Maitama Sule (Northwest) University, Kano, Nigeria. Please visit his website for more details.

Please follow his publications on **ORCID: 0000-0003-1413-3974**

1 About DynareR

DynareR is an R package that can run `Dynare` program from R, R Markdown and Quarto.

2 Requirements

Users need the following in order to knit this document:

1. Dynare 4.6.1 or above
2. Octave 5.2.0 or above
3. Dynare is installed in the standard location as follows:

- `/usr/lib/dynare/matlab` for `Linux`
- `/usr/lib/dynare/matlab` for `macOS`
- `c:/dynare/x.y/matlab` for `Windows`, where `x.y` is `Dynare` version number.

If `dynare` and `Octave` are installed in standard location, `DynareR` package will take care of the configurations, which include adding `matlab` directory to path, using the latest installed `dynare` and so on. Otherwise, users have to specify the `matlab` folder using `add_path` function, set the `Octave` path using the `set_octave_path` function, or set `dynare` version using the `set_dynare_version` function.

3 Installation

DynareR can be installed using the following commands in R.

```
install.packages("DynareR")

          OR

devtools::install_github('sagirumati/DynareR')
```

4 Usage

Please load the DynareR package as follows:

~~~
```{r DynareR}
library(DynareR)
```
~~~

Then create a chunk for `dynare` (adopted from Dynare example file `example1`) as shown below:

~~~
```{dynare example1}
/*
 * Example 1 from F. Collard (2001): "Stochastic simulations with DYNARE:
 * A practical guide" (see "guide.pdf" in the documentation directory).
 */

/*
 * Copyright (C) 2001-2010 Dynare Team
 *
 * This file is part of Dynare.
 *
 * Dynare is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * Dynare is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with Dynare.  If not, see <http://www.gnu.org/licenses/>.
 */


var y, c, k, a, h, b;
varexo e, u;

parameters beta, rho, alpha, delta, theta, psi, tau;

alpha = 0.36;
rho   = 0.95;
tau   = 0.025;
beta  = 0.99;
delta = 0.025;
psi   = 0;
theta = 2.95;

phi   = 0.1;

model;
c*theta*h^(1+psi)=(1-alpha)*y;
k = beta*(((exp(b)*c)/(exp(b(+1))*c(+1)))
    *(exp(b(+1))*alpha*y(+1)+(1-delta)*k));
y = exp(a)*(k(-1)^alpha)*(h^(1-alpha));
k = exp(b)*(y-c)+(1-delta)*k(-1);
a = rho*a(-1)+tau*b(-1) + e;
b = tau*a(-1)+rho*b(-1) + u;
end;

initval;
y = 1.08068253095672;
c = 0.80359242014163;
h = 0.29175631001732;
k = 11.08360443260358;
a = 0;
b = 0;
e = 0;
u = 0;
end;

shocks;
var e; stderr 0.009;
var u; stderr 0.009;
var e, u = phi*0.009*0.009;
end;

stoch_simul(order=1, hp_filter=1600,graph_format=pdf);
```
~~~

The above chunk creates a Dynare program with the chunk's content, then automatically run Dynare, which will save Dynare outputs in the current directory.

Please note that DynareR uses the chunk name as the model name. So, the outpus of Dynare are saved in a folder with its respective chunk name. Thus a new folder `example1/` will be created in your current working directory.

By default, `dynare` chunk imports log output as a list of dataframes, which can be accessed via `dynare$modelName`. Therefore to access the outputs of the `example1` model produced by the `dynare` chunk, use `dynare$example1`.

Use inline code to access the value of second row and third column of the `moments`, which is 0.0024.

5 Plotting the IRF

The Impulse Response Function (IRF) is saved by default in `example1/example1/graphs/` folder with the IRF's name `example1_IRF_u.pdf`, where `example1` is the Dynare model's name. Therefore, you need to add `stoch_simul(graph_format = (pdf))` to change the default saving behaviour of `Dynare` from `eps` to `pdf`.

6 DynareR functions for base R

The DynareR package is also designed to work with base R. The following functions show how to work with DynareR outside the R Markdown or Quarto documents.

6.1 The include_IRF function

Use this function to embed the graphs Impulse Response Function (IRF) in R Markdown or Quarto document.

The Impulse Response Function (IRF) of the `example1` model can be fetched using the following R chunk. Note that only the last part of the IRF's name (`u`) is needed, that is `example1_IRF_` is excluded.

```
include_IRF(model="example1",IRF = "u")

# Alternatively, use the path argument

include_IRF(path="example1/example1/graphs/example1_IRF_u.pdf")
```

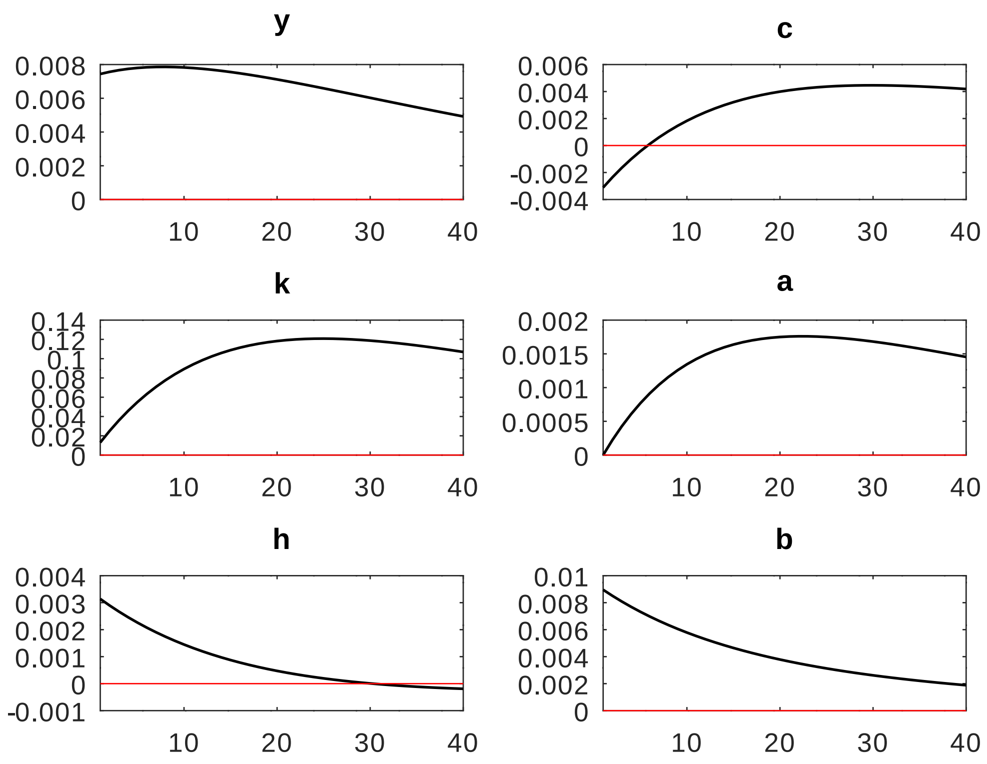

Figure 1: A figure generated from Dynare software

However, Dynare figure can only be dynamically included if the output format is pdf as Dynare produces pdf and eps graphs only.

6.2 The write_dyn function

This function writes a new `dyn` file.

Use `write_dyn(code="code",model="someModel")` if you want the `Dynare` file to live in the current working directory. if you want the Dynare file to live in the path different from the current working directory:

`write_dyn(code="code",model="path/to/someDirectory/someModel")`

```
dynareCodes='var y, c, k, a, h, b;
varexo e, u;
parameters beta, rho, alpha, delta, theta, psi, tau;
alpha = 0.36;
rho   = 0.95;
tau   = 0.025;
beta  = 0.99;
```

```
delta = 0.025;
psi   = 0;
theta = 2.95;
phi   = 0.1;
model;
c*theta*h^(1+psi)=(1-alpha)*y;
k = beta*(((exp(b)*c)/(exp(b(+1))*c(+1)))
          *(exp(b(+1))*alpha*y(+1)+(1-delta)*k));
y = exp(a)*(k(-1)^alpha)*(h^(1-alpha));
k = exp(b)*(y-c)+(1-delta)*k(-1);
a = rho*a(-1)+tau*b(-1) + e;
b = tau*a(-1)+rho*b(-1) + u;
end;
initval;
y = 1.08068253095672;
c = 0.80359242014163;
h = 0.29175631001732;
k = 11.08360443260358;
a = 0;
b = 0;
e = 0;
u = 0;
end;

shocks;
var e; stderr 0.009;
var u; stderr 0.009;
var e, u = phi*0.009*0.009;
end;

stoch_simul;'


write_dyn(code=dynareCodes, model="example1")

write_dyn(code=dynareCodes,model="DynareR/write_dyn/example1")
```

6.3 The write_mod function

This function writes a new `mod` file.

Use `write_mod(code="code",model="someModel")` if you want the `Dynare` file to live in the current working directory. if you want the Dynare file to live in the path different from the current working directory:

`write_mod(code="code",model="path/to/someDirectory/someModel")`

```
DynareCodes='var y, c, k, a, h, b;
varexo e, u;
parameters beta, rho, alpha, delta, theta, psi, tau;
alpha = 0.36;
rho   = 0.95;
tau   = 0.025;
beta  = 0.99;
delta = 0.025;
psi   = 0;
theta = 2.95;
phi   = 0.1;
model;
c*theta*h^(1+psi)=(1-alpha)*y;
k = beta*(((exp(b)*c)/(exp(b(+1))*c(+1)))
          *(exp(b(+1))*alpha*y(+1)+(1-delta)*k));
y = exp(a)*(k(-1)^alpha)*(h^(1-alpha));
k = exp(b)*(y-c)+(1-delta)*k(-1);
a = rho*a(-1)+tau*b(-1) + e;
b = tau*a(-1)+rho*b(-1) + u;
end;
initval;
y = 1.08068253095672;
c = 0.80359242014163;
h = 0.29175631001732;
k = 11.08360443260358;
a = 0;
b = 0;
e = 0;
u = 0;
end;

shocks;
var e; stderr 0.009;
var u; stderr 0.009;
var e, u = phi*0.009*0.009;
end;
```

```
stoch_simul;'


write_mod(model="example1",code=dynareCodes)

write_mod(code=dynareCodes,model="DynareR/write_mod/example1")
```

6.4 The run_dynare function

Create and run Dynare `mod` file

Use this function to create and run Dynare mod file. Use `run_dynare(code="code",model="someMode` if you want the Dynare files to live in the current working directory. if you want the Dynare files to live in the path different from the current working directory:

`run_dynare(code="code",model="path/to/someDirectory/someModel")`

Use `import_log=T` argument to return the `dynare` log file as list of dataframes in an environment `dynare`, which can be accessed via `dynare$modelName`.

```
DynareCodes='var y, c, k, a, h, b;
varexo e, u;
parameters beta, rho, alpha, delta, theta, psi, tau;
alpha = 0.36;
rho   = 0.95;
tau   = 0.025;
beta  = 0.99;
delta = 0.025;
psi   = 0;
theta = 2.95;
phi   = 0.1;
model;
c*theta*h^(1+psi)=(1-alpha)*y;
k = beta*(((exp(b)*c)/(exp(b(+1))*c(+1)))
          *(exp(b(+1))*alpha*y(+1)+(1-delta)*k));
y = exp(a)*(k(-1)^alpha)*(h^(1-alpha));
k = exp(b)*(y-c)+(1-delta)*k(-1);
a = rho*a(-1)+tau*b(-1) + e;
b = tau*a(-1)+rho*b(-1) + u;
end;
initval;
y = 1.08068253095672;
```

```
c = 0.80359242014163;
h = 0.29175631001732;
k = 11.08360443260358;
a = 0;
b = 0;
e = 0;
u = 0;
end;

shocks;
var e; stderr 0.009;
var u; stderr 0.009;
var e, u = phi*0.009*0.009;
end;

stoch_simul;'

run_dynare(code=DynareCodes,model="example1",import_log = T)
run_dynare(code=DynareCodes,model="DynareR/run_dynare/example1")
```

6.5 The run_models function

Run multiple existing `mod` or `dyn` files.

Use this function to execute multiple existing Dynare files. Use `run_models(model="someModel")` if the Dynare files live in the current working directory. If the Dynare files live in the path different from the current working directory:

`run_models(model="path/to/someDirectory/someModel")`

Use `run_models()` to exectute all the `dynare` models in the current working directory. Use `run_models("path/to/someDirectory*)` to run all the `dynare` models in `path/to/someDirectory`.

Where `agtrend.mod`, `example1.mod` and `example1.mod` are the Dynare model files (with `mod` or `dyn` extension), which live in the current working directory.

```
demo(agtrend)
demo(bkk)
demo(example1)

# Provide the list of the `Dynare` files in a vector
```

```r
# Ensure that "agtrend.mod", "bkk.mod" and "example1.mod"
# live in the current working directory

# Copy the dynare files to the current working directory

lapply(c("agtrend","bkk","example1"),\(x) file.copy(paste0(x,"/",x,".mod"),"."))

run_models(c("agtrend","bkk","example1")) # Run the models in the vector.
```

To run all Dynare models that live in the current working directory, use the following:

```r
run_models() # Run all models in Current Working Directory.
```

To run all Dynare models that live in particular path (for example 'DynareR/run_dynare/' folder), use the following:

```r
# Copy the dynare files to the 'DynareR/run_dynare' directory

lapply(c("agtrend","bkk","example1"),\(x) file.copy(paste0(x,".mod"),"DynareR/run

run_models(model = 'DynareR/run_dynare*') # notice the * at the end
```

7 import_log function

This function returns the dynare log output as a list of dataframes, which include summary, shocks, policy, moments, decomposition, correlation and autocorrelation. The list is accessible via dynare$modelName. if the model name is example1, the policy variables can be obtained via dynare$example1$policy as a dataframe.

```r
import_log(model="example1")

import_log(path="example1/example1.log")

knitr::kable(dynare$example1$autocorrelation)
```

8 set_dynare_version function

On Windows, you can set the version of dynare you want to use. By default, `DynareR` package does this for you if the dynare version ranges from 4.6.1 to 9.9. However, if you are using the development version of `dynare`, for example version `6-unstable-2022-04-03-0800-700a0e3a`, you can override the default as follows

```
set_dynare_version("6-unstable-2022-04-03-0800-700a0e3a")
```

9 set_octave_path function

You can use this function if `Octave` is not installed in the standard location.

```
set_octave_path('C:/Program Files/GNU Octave/Octave-6.4.0/mingw64/bin/octave20.exe
```

10 add_path function

This function is a wrapper of `addpath` in `Octave`. If `dynare` is not installed in the standard location, use this function to add the `matlab` subdirectory. By default, `DynareR` does this for if `dynare` is installed in the standard location.

```
add_path('/usr/lib/dynare/matlab')#  Default for Linux

add_path('c:/dynare/5.1/matlab') # Default for Windows, but 5.1 can change if late
# `Dynare` is installed.

add_path('/usr/lib/dynare/matlab') # Default for macOS
```

11 Demo

The demo files are included and can be accessed via demo(package="DynareR")

```
demo(run_dynare)
demo(run_models)
demo(import_log)
```

12 Template

Template for R Markdown is created. Go to `file->New File->R Markdown-> From Template->DynareR`.

Similar packages

Similar packages include EviewsR (Mati, 2020b, 2022b; Mati et al., 2023, 2024), gretlR (Mati, 2020c, 2022c), and URooTab (Mati, 2023b, 2023a)

For further details, consult Mati (2020a) and Mati (2022a).

Please download the example files from Github.

References

Mati, S. (2020a). DynareR: Bringing the power of dynare to R, R Markdown, and Quarto. *CRAN*. https://CRAN.R-project.org/package=DynareR

Mati, S. (2020b). *EviewsR: A seamless integration of EViews and R*. https://CRAN.R-project.org/package=EviewsR

Mati, S. (2020c). *gretlR: A seamless integration of Gretl and R*. https://CRAN.R-project.org/package=gretlR

Mati, S. (2022a). *Package "DynareR"*. https://cran.r-project.org/web/packages/DynareR/DynareR.pdf

Mati, S. (2022b). *Package "EviewsR"*. https://cran.r-project.org/web/packages/EviewsR/EviewsR.pdf

Mati, S. (2022c). *Package "gretlR"*. https://cran.r-project.org/web/packages/gretlR/gretlR.pdf

Mati, S. (2023a). *Package "URooTab"*. https://cran.r-project.org/web/packages/URooTab/URooTab.pdf

Mati, S. (2023b). *URooTab: Tabular reporting of EViews unit root tests*. https://github.com/sagirumati/URooTab

Mati, S., Civcir, I., & Abba, S. I. (2023). EviewsR: An r package for dynamic and reproducible research using EViews, r, r markdown and quarto. *The R Journal*, *15*(2), 169–205. https://doi.org/10.32614/rj-2023-045

Mati, S., Ismael, G. Y., Masoud, S., Hamad, K. Q., Mohammed, A. A., & Hussaini, M. (2024). Revisiting ECOWAS-eurozone exports in the light of asymmetry. *Cogent Economics & Finance*, *12*(1), 2309812. https://doi.org/10.1080/23322039.2024.2309812