Code for "Penalized estimation of sparse Markov regime-switching vector auto-regressive models"

Gilberto Chavez-Martinez * Ankush Agarwal † Abbas Khalili *

Syed Ejaz Ahmed ‡

1 Introduction

In this document, we illustrate how to reproduce Figure 1(a) and Figure 2(a) of our paper titled "Penalized estimation of sparse Markov regime-switching vector auto-regressive models".

We also illustrate how to fit an MSVAR model to a simulated or user-provided dataset.

The GNU Scientific Library GSL, version 2.6 or above, needs to be installed, as well as a `C++` compiler, `R` version 4.1 or above, and the `R` package `RcppGSL`. To reproduce Figure 1(a) and Figure 2(a) of the paper, the `R` package `ggplot2` is required. For Windows, the package `Rtools` is also required to run our code. For macOS, the `R` compilation toolchain is required. See instructions below.

The directory `"C"` contains the `C`/`C++` source code files. The directory `"R"` contains the `R` scripts. All files needed by the scripts are located in the directories with prefix `"msvar_data_"`.

Reproducing Figure 1(a) and Figure 2(a)

The `R` script `msvar_generate_figures.R` generates a set of random samples (50 replications considered in our paper) from MSVAR models with the pre-specified parameters given in the subdirectories with prefix `"msvar_data_d40_"`, for dimension $d = 40$. It fits an MSVAR model and obtains the parameter estimates based on each generated random sample. It then generates and saves the plots as in Figure 1(a) and Figure 2(a).

*Department of Mathematics and Statistics, McGill University, Montreal, Quebec H3A 0B9, Canada **e-mail**: gilberto.chavezmartinez@mail.mcgill.ca and abbas.khalili@mcgill.ca

†Adam Smith Business School, University of Glasgow, G12 8QQ Glasgow, United Kingdom **e-mail**: ankush.agarwal@glasgow.ac.uk

‡Faculty of Mathematics and Science, Brock University, St. Catharines, Ontario L2S 3A1, Canada **e-mail**: sahmed5@brocku.ca

Fitting an MSVAR model to a simulated or a user-provided dataset

The `R` script `msvar_usage.R` generates a random sample from an MSVAR model with the pre-specified parameters given in the subdirectory `"msvar_data_d5_n200"`, for dimension $d = 5$ and sample size $n = 200$, as a toy example. It then fits an MSVAR model and obtains the parameter estimates based on the generated data.

The `R` script `msvar_usage.R` can also be used to fit an MSVAR model to a user-provided dataset.

Further details are provided within each `R` script.

The rest of this document is organized as follows. Section 2 describes how to install the compilers and GSL for Linux, Windows, and macOS. Section 3 contains the instructions on installing the required `R` packages. Section 4 provides instructions on setting up our `R` scripts and explains how to use them.

2 Installing compilers and GSL

2.1 Linux

The compilers and GSL can be installed using the package or software manger for some distributions, or compiling it from source, as explained in the following two subsections.

2.1.1 Using the package manager

Several Linux distributions allow the installation through their package or software manager, for example:
For Ubuntu and other Debian-based distributions, run the commands:

```
sudo apt install build-essential
sudo apt install libgsl-dev
```

For Arch Linux and other Arch-based distributions, run the commands:

```
sudo pacman -Syu base-devel
sudo pacman -Syu gsl
```

(the packages `build-essential` and `base-devel` install the GCC compiler suite).

If the package manager of the Linux distribution cannot install GSL, then GSL can be compiled and installed following the steps in Section 2.1.2.

2.1.2 Compiling the library from source

Download the package from https://www.gnu.org/software/gsl/. We assume that the downloaded GSL package has version 2.7.

Place the downloaded package file in the home directory and unpack the file with the following command:

```
tar -zxvf gsl-2.7.tar.gz
```

This will create a directory called "gsl-2.7" in the home directory. Change to the new directory running the command:

```
cd gsl-2.7
```

Configure the installation running the command:

```
./configure
```

If there are no errors, compile the library with the following command; please note that this step may take several minutes:

```
make
```

If there are no errors, install the library with:

```
sudo make install
```

2.2 macOS

To begin, the simplest way to install the R compilation toolchain is by using the macrtools package. This is achieved by running, inside an R session, the commands:

```
install.packages("remotes")
remotes::install_github("coatless-mac/macrtools")
macrtools::macos_rtools_install()
```

The next step is to install GSL. It can be installed using the Homebrew or MacPorts package mangers, or compiling it from source. Additionally, GSL can be installed using a script from the R macOS repository. See below.

2.2.1 Using a package manager

Install GSL using Homebrew with the command:

```
brew install gsl
```

or using MacPorts with the command:

```
sudo port install gsl
```

2.2.2 Using the R repository for macOS

This currently supports Intel Mac with macOS 10.13 and higher, and Apple Silicon (M1) Mac with macOS 11 and higher. First start an R session and run the following:

```
source("https://mac.R-project.org/bin/install.R")
```

This defines the R function install.libs. To install the GSL library, run inside the same R session:

```
install.libs("gsl")
```

Note: if you do not have writing permission in the installation location, you may need to start the R session by running in a terminal the command:

```
sudo R
```

2.2.3 Compiling the library from source

Alternatively, GSL can be compiled and installed following the instructions in Section 2.1.2.

2.3 Windows

The `Rtools` package contains a compiled version of GSL and the compiler tools needed. The installer can be downloaded from https://cran.r-project.org/bin/windows/Rtools/. Please note that the appropriate version must be used, depending on the `R` version installed on your computer.

3 Installing the R packages

Inside an `R` session, run the commands:

```
install.packages("RcppGSL")
install.packages("ggplot2")
```

Up to this point, all the requirements are installed and our `R` scripts can now be used as explained in the next section.

4 Using the R scripts

4.1 Setting the working directory

Inside an `R` session, first set the working directory to the subdirectory named `"R"` that contains our `R` scripts. If the `R` session is started from this directory, this step is not needed. Otherwise, inside the `R` session run

```
setwd("PATH")
```

where `PATH` is the path to our `"R"` subdirectory. If using `RStudio`, open any of our `R` scripts, then go to menu `Session` and click `Set Working Directory` and then `To Source File Location`.

4.2 Reproducing Figure 1(a) and Figure 2(a)

The `R` script `msvar_generate_figures.R` fits MSVAR models to 50 randomly generated samples (see Section 5 of the main paper) and generates Figure 1(a) and Figure 2(a) of the paper. The plots are generated and saved in the working directory.

To save computational time, here we provide pre-selected tuning parameters as input files.

4.3 Fitting an MSVAR model to a simulated or user-provided dataset

The `R` script `msvar_usage.R` illustrates how to use our code to generate one synthetic dataset, select the tuning parameters, and fit an MSVAR model to the dataset. Alternatively, a user-provided dataset can be used.

Further details are provided within each `R` script.