Installation Instructions Manual for `dynr` - Developer version

Michael D. Hunter, Sy-Miin Chow, Lu Ou, Sukruth N. Reddy,
Meng Chen, Linying Ji, Jungmin Lee, Xiaoyue Xiong, & Sharon Kim

October 26, 2024

The `dynr` software is a package for R written in R and C. It has utilities which allow users to create C code for linear and nonlinear dynamic models, including regime-switching models, without ever having to actually write C code. Various functions in R are called by the user to specify the model desired. These functions in turn write code in C based on the user's input. These functions are then compiled and shared with the rest of the `dynr` C code to estimate the model, obtaining free parameter estimates, standard errors, and latent variable estimates.

1 Instructions for Installing and Using `dynr` on a Windows Computer

Because the `dynr` package compiles C code in response to user input, more setup is required for the `dynr` package than for many others. The general requirements are as follows:

1. R must be installed in a directory without spaces in the path. See Section 1.1.
2. Rtools must be installed so that C code can be compiled on Windows. See Section 1.2.
3. Additional GSL libraries must be installed so that the C code can use GSL for matrix multiplication routines. See Section 1.3.
4. The environment variable for the system Path must be modified to include R and Rtools. See Section 1.5.
5. An environment variable for GSL called LIB_GSL must be created. See Section 1.4.

Each of these steps is detailed below. We acknowledge that this additional setup can be bothersome, but we believe the ease of use for the rest of the

package and the wide variety of models it is possible to fit with it will compensate for this initial burden. Hopefully you will agree!

1.1 Instructions for Installation of R

- Make sure that R is installed and Check the directory in which R is installed.
- If not go to https://www.r-project.org/ and click on download R and select any mirror.
- The critical step is just to make sure that you install R to a directory where there are **no spaces** in the words describing the directory. By default, it will suggest to install R in "C:\Program Files" on your computer. Instead change it to just "C:\" *or any other directory where there are* **no spaces** *in the words describing the directory.* For example as given in the default directory "Program Files" has a space in between the two words so avoid using such directories.

1.2 Instructions for Installation of R-tools

1. If you already have R-tools on your machine, make sure the version of R-tools matches with your R version. R-tools versions newer than 3.0 should work fine with `dynr`.
2. Install R-tools through `https://cran.r-project.org/bin/windows/Rtools/`. Install the latest *release* version of Rtools, not the *devel* version.
3. The following instructions are based on rtools42.
4. Click "Rtools42 installer", then click "rtools42-5355-5357.exe" in the new webpage. You may be asked if you want to save or run a file "rtools42-5355-5357.exe". Choose "Save" and save the file on the Desktop. Then double-click on the icon for the file to run it.
5. If you are asked what language to install, choose English.
6. The next page says "Select Destination Location" at the top. By default, it will suggest to install R in "C:\rtools42" on your computer. You may also install R-tools in *any other directory where there are* **no spaces** *in the words describing the directory.* Click "Next" at the bottom of the R tools Setup wizard window.
7. The next page says "Select additional tasks" at the top. Check the box to edit the system PATH. Click "Next" again.
8. Add the directories for R and Rtools to your *system* PATH. Examples are shown below. Place these directories at or near the top of your list for the

system PATH variable. The order of the directories should be the same as shown in the example below.
`C:\rtools42\usr\bin`
`C:\R\R-4.2.1\bin\x64;`
You may want to first go to the directory where Rtools and R are installed and check that you know the correct pathway.

9. The next page says "Ready to install" at the top. Click "install".
10. Rtools should now be installed. This will take about a minute. Click "Finish".

1.3 Instructions for Installation of GSL

1. If you already have GSL libraries installed on your machine, you may skip this step and go to Section 1.4.
2. To install GSL libraries, we advocate using Rtools. The new way to get GSL is through Rtools.
3. Go to the directory on your computer for Rtools (usually `C:/rtools42` or something similar). There's a file called `msys2.exe`. Run it. It should open a shell command prompt. Type

   ```
   pacman -Sy mingw-w64-x86_64-gsl
   ```

   in that command prompt. This installs GSL. Check by going to `C:/rtools42/mingw64/lib`, and find a file called `libgsl.a`. If it's there, then it installed GSL.
4. If it's not there, then `pacman` probably did not install GSL correctly. Search online for "msys2" and "GSL" to find updated commands for installing GSL with msys2.

1.4 Setting up the GSL Environment Variable

1. Open - Control Panel\System and Security\System
2. Click on Advanced system settings and then click on "Environment Variables"
3. Add a new *system* variable by clicking on New. Note that this should be a *System* environment variable, not a *User* environment variable.
4. Name the new variable as LIB_GSL and set the variable value to the directory one back from `libgsl.a`. This file is usually in `C:/rtools42/mingw64/lib`, so you'll set `LIB_GSL` to `C:/rtools42/mingw64`.
5. Note that the direction of these slashes is important. This slash / will work but *not* this one \. The `LIB_GSL` path should be `C:/rtools42/mingw64` and not `C:\rtools42\mingw64`.

1.5 Setting up R and Rtools Environment Variable

1. Check if R can be run through CMD as follows. Put a simple R script (e.g., save a simple R command: print ("hello") to a .R document as "Hello.R") to a known directory. Open the command prompt window. If you can't find it just do a search from the Start Menu for "cmd" and open "Command Prompt.exe". Then change directory to the location containing "Hello.R" (e.g., cd C:\myfiles). Run the script by typing "Rscript Hello.R". If the file runs correctly, it should print out "hello" in the command prompt console.
2. Close your command prompt window.
3. If the file did not run correctly, follow the next 4 steps.
4. Open - Control Panel\System and Security\System
5. Click on Advanced system settings
6. Check to make sure that your path variable specified during Rtools installation is specified correctly. In Windows 10 do so by clicking on "Environment Variables" and then in the "System variables" panel click on "Path" and then "Edit". Check that the following paths have been specified at the front of your *system* path variable or just move them all to way up to the top in the following order.
   `C:\rtools42\usr\bin`
   `C:\R\R-4.2.1\bin\x64`
   `C:\cygwin\bin (see section 1.6)`
7. The order of the directories on the PATH variable is important. CMD looks for programs starting at the top of the directories in PATH. Make sure you have Rtoos, then R, then cygwin.
8. Repeat (1) and see if you can now run R from CMD. Make sure that you open a new CMD window after you have completed the steps above. Try typing in "PATH" in CMD to make sure that the newly added paths are indeed shown in the output.

1.6 Instructions for Installation of Cygwin

1. Install Cygwin through https://cygwin.com/index.html.
2. When prompted, select to install the following packages in cygwin:
   (a) git (under Devel): "Distributed version control system"
   (b) gcc (under Devel): "gcc-core: GNU Compiler Collection (C, OpenMP)"
   (c) make (under Devel): "The GNU version of the 'make' utility"
   (d) perl: (under Perl): "Perl programming language interpreter"
3. Make sure that `C:\cygwin\bin` is in the path variable, if not, add it

1.7 Wrapping up the Installation Procedure for Windows

1. Open RGui, Rstudio or whatever editor you use to run R. Please type the following code to check whether the gsl commands can be found correctly:
`shell("echo %LIB_GSL%")`
If this returns something like `C:/rtools42/mingw64`, then everything worked fine.

2. If that command returns something like "%LIB_GSL%", then something might be wrong with the GSL installation (Return to Section 1.3) or GSL path (Return to Section 1.4).

3. If the command worked fine, follow the steps in section 3 to finish the installation process in R.

2 Instructions for Installing and Using `dynr` On Mac

2.1 Instructions for Installing Prerequisites on Mac

1. Open your terminal window and type in `xcode-select --install` and follow the instructions for all the subsequent steps. Please pick the correct software version for your operating system.

   - **NOTE:** If you have Big Sur as your Operating System, you do not need to install the command line tools. In fact, you need to remove it by navigating to /Library/Developer and remove Command Line Tools.
   - Then, edit /Library/Frameworks/R.framework/Resources/etc/Makeconf and remove all compilers' explicit link to a particular OS; e.g., change "CC=clang -mmacosx-version-min=10.16" to just "CC=clang." Please see the FAQ section on the dynr GitHub webpage for further updates.

2. Install Macports by going to `https://www.macports.org/install.php`

3. Open the terminal window. In the terminal window install the gsl library by typing: "sudo port install gsl".

4. Follow the steps in section 2.2 to verify that these steps worked properly.

2.2 Checking the Installation for Mac

1. Open RGui, Rstudio or whatever editor you use to run R. Please type the following code to check whether the gsl commands can be found correctly:
`system("gsl-config --cflags", intern=TRUE)`

When the command can not be found, and you know where it is stored (e.g., "/opt/local/bin"), we could then set the PATH variable by typing: `Sys.setenv(PATH=paste0(Sys.getenv("PATH"),":","/opt/local/bin"))` and then check again.

2. If the above failed, then something went wrong with one or several of the steps in Section 2.1. Please go back and try repeating or checking that section.

3. Follow the steps in section 3 to finish the installation process in R.

3 Getting `dynr` from GitHub Repository

1. Create a GitHub account if you don't already have one (https://github.com/). Email Michael D. Hunter (mike.dynr at gmail.com) for permission to dynr's GitHub repository.

2. Once permission is granted, open a Command Prompt window (it can be done by searching "cmd" in the windows search box) on Windows, or a Terminal window on Macs, and change the directory to a desired location for the dynr package folder (e.g. by typing `cd <path to location>`).

3. Clone the repository from GitHub. It should involve typing something like `git clone <https://github.com/mhunter1/dynr.git>` in the window.

4. Install the dependencies of dynr yourself.

5. There is a bug in `roxygen2` such that the latest version will not build our documentation. It is a known issue that will be resolved soon. In the meantime, use an older version of the package. Install it with `devtools::install_version(package = 'roxygen2',version = '5.0.1', repos = c(CRAN = "https://cran.rstudio.com"))`

6. Install `dynr` by running `make clean install` in the Command Prompt/Terminal window.

7. To test if `dynr` is installed correctly, run one of the demo examples in R. For instance, type:
`demo('LinearSDE',package='dynr')`
`Press <Return> to start the demo.`
Note that this specific demo requires the packages *Matrix* and *mvtnorm* so make sure that these have been installed.

8. If you're on Windows and everything worked fine until you tried to run the model, something probably went wrong with installing Rtools (Section 1.2) or installing GSL (Section 1.3). Please refer to those sections for troubleshooting.

4 Advanced Operations When Developing Dynr

4.1 Import Functions in `dynr`

1. Make sure that you have a local `dynr` repository in your computer.
2. Change the `NAMESPACE` file.
   - Find the `NAMESPACE` file in the `dynr` directory. Open it by any code editor (e.g., TextEdit on Mac or Wordpad on Windows).
   - Add one line to import the package with `importFrom` function or `import` function at the end of the file, so that you can use corresponding datasets and functions from that package when using `dynr` package. For example, add `import(Rcpp)` to use `Rcpp` package.
   - More details can be found in <Managing imports and exports> (https://cran.r-project.org/web/packages/roxygen2/vignettes/namespace.html) and more examples can be found in the `NAMESPACE` file.
   - Save and quit the file.
3. Change the `description.in` file.
   - Find the `description.in` file in the `dynr` directory. Open it by any code editor (e.g., TextEdit on Mac or Wordpad on Windows).
   - Locate the line that begins with `Imports:`, and add the name of the package which you want to import at the end of the line.
   - Save and quit the file.

4.2 Export Functions from Other R Packages

1. Make sure that you have a local `dynr` repository in your computer.
2. Find the `NAMESPACE` file in the `dynr` directory. Open it by any code editor (e.g., TextEdit on Mac or Wordpad on Windows).
3. Add one line to export the package with `export, exportS3Method, exportClass, exportMethod` or `exportPattern` function at the end of the file. More details can be found in <Managing imports and exports> (https://cran.r-project.org/web/packages/roxygen2/vignettes/namespace.html) and more examples can be found in the `NAMESPACE` file.
4. Save and quit the file.