Tools installation for seasonnal adjustment

Tanguy BARTHELEMY

2023-01-17

Context

The helping tools for seasonal adjustment are:

- **R** and Rstudio
- JDemetra+

⚠ No assistance in **SAS** language will be given. ⚠

Preliminary

On the computers without administrator rights (*professionnal computer, for example*), it is recommended to create a folder **Software** under C:\Users\...\Software or directly under C:\Users\Software where all software will be installed.

⚠ Warning: when we specify an **absolute** path for a software (JDemetra+, Java, **R**, …) in a program, a shortcut, a variable, …, it must be modified each time any root repository is moved.

1 Installation of JDemetra+

JDemetra+ is a collection of Java programs used for the study of time series and more specifically for seasonnal adjustment. JDemetra+ is delivered in the forme of an GUI (Graphical User Interface) but there are **R** packages developed to be used with **R** as well as a cruncher (executable).

1.1 Version of JDemetra+ and dependence

JDemetra+ is downloadable from the github link of the application: https://github.com/jdemetra/jdemetra-app/releases.

The last release (v2.2.3) dates from July 7, 2020. It is the last **stable** version of JDemetra+. **This version should be downloaded and must be used in production.**

There are two more versions of JDemetra+ which are only at the **test** stage:

- v2.2.4: a v2.2.3 corrected from bugs but without adding some major new features
- v3.0.0: the new JDemetra+ version with new features and a new GUI

JDemetra+ v2.2.3 and v2.2.4 require Java version ≥ 8 while v3.0.0 require Java version ≥ 17:

JDemetra+ version	Java version
v2.2.3	≥ 8
v2.2.4 (RC2)	≥ 8
v3.0.0 (RC1)	≥ 17

RC: Release candidate

For the sequel, the installation processes of this 3 versions are the same. You just have to repeat them for each version you want to install.

1.2 Installation process

There are two possibility for installing:

- **Download and exécuter** the file `.exe` which need administrator rights
- **Download** and **unzip** the compressed folder `.zip` that allows to get a portable version of the software

⚠ Warning: for the second option, you need to **download** the compressed folder `jdemetra+-2.2.3-bin.zip` (for the version 2.2.3 for example) and **not** the folder `Source code (zip)`.

The Software is in the folder \nbdemetra\bin\, these are the file `nbdemetra.exe` (version 32-bit) and `nbdemetra64.exe` (version 64-bit).

ℹ Advise: If you want to use several version of JDemetra+ (v2.2.3, v3.0.0, …), you can rename the unzipped folder in \nbdemetra-2.2.3\, \nbdemetra-2.2.4\ and \nbdemetra-3.0.0\.

ℹ Remark: You can create shortcut fo the executables if you want to launch them from another folder (Desktop, project folder…).

1.3 Installation of the cruncher

The cruncher (**JWSACruncher**) is a tool to update a workspace of JDemetra+ from the console, without opening JDemetra+. The update of a workspace can then be done from another Software (**R** or **SAS** for example).

To use the cruncher, you have to:

- **Download** and **unzip** the file from the **latest stable** version v2.2.3 here https://github.com/jdemetra/jwsacruncher/releases

If you want to install and use a portable Java version (See section Java installation), you have to modified some parameters to use the cruncher:

- In the unziped folder, **open** (for example with Notepad++) the file `jwsacruncher.bat` present in the subfolder \bin\ (that is under jdemetra-cli-2.2.3\bin\ in the version 2.2.3 of the cruncher)
- **Modify** the valeur of the variable `JAVACMD` at the line **71** (currently `JAVACMD=java`) by the adress toward the file `java.exe` of the portable version. Then, if JPortable is installed under C:\Users\Software, the new line is `if "%JAVACMD%"=="" set JAVACMD="C:\\Users\\Software\\Java64\\bin\\java"` (for Java 8).

2 Installation of Java

ℹ On Insee computers, Java is already installed in version 8. Then, there is no need to install a portable version to use JDemetra+ in version 2.2.3 or 2.2.4.

2.1 Java 8

To install Java 8, use the link https://portableapps.com/apps/utilities/java_portable. If you use the version 64-bit of JDemetra+, you should install the version jPortable 64-bit (at the bottom of the page).

2.2 Java 17

To install Java 17, you need to go at the adresse https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html.

- **Download** the version `Compressed Archive` of Windows (https://download.oracle.com/java/17/archive/jdk-17.0.4.1_windows-x64_bin.zip)
- **Unzip** the folder jdk-17.0.6 under C:\Users\Software (for example)

After a Java installation (in version 8, 17 or other), you need to:

- **Modify** the environment variable `PATH` of Rstudio and of Windows (See section Environment variables)
- **Modify** the targets of JDemetra+ to inform of the localisation of the new Java versions.

For example, if you installed Java version 17 to use the JDemetra+ version 3.0.0. You should add the localisation of Java 17 at the shortcut of the executable using the option `--jdkhome`. The target of the shortcut become `C:\Users\Software\nbdemetra-3.0.0\bin\nbdemetra64.exe --jdkhome "C:\Users\Software\Java17\jdk17"`

3 Installation of R and Rstudio

The JDemetra+ features are availables on **R** via **R** packages. To use **R**, it is better to use an IDE so Rstudio. All the executables to download are under https://posit.co/download/rstudio-desktop/#download.

3.1 Installation of R

To install **R**, you should:

- **Download** the binary file `R-4.2.2-win.exe` under https://cran.rstudio.com/bin/windows/base/
- **Execute** the executable to parameter and install **R**.

3.2 Installation of Rstudio

Download the last Rstudio version (under https://posit.co/download/rstudio-desktop/#download) and the **installer**.

If the installation via the file `.exe` fails (because it requires higher rights (administrator, elevation, ...), we will get a portable version of the Software. To do this:

- **Download** and **unzip** the compressed folder `.zip` in a folder named "Rstudio" (under C:\Users\Software)
- **Create a shortcut** of the file `rstudio.exe` on the Desktop.

3.3 Installation of R packages

To install a **R** package, there is several methods:

- Either it is on CRAN and you can install it directly with the function `install.packages()`
- Or it is on Github and you can install it directly with the function `install_github()` from the package **remotes**
- Or you have to retrieve the package at the binary format (`.zip`) and then install with the function `install.packages()` with the arguments `repos = NULL, type = "binary"`.

3.3.1 In version 2

The packages in version 2 are:

Name	Available on CRAN	Github link
RJDemetra	✅	https://github.com/jdemetra/rjdemetra
rjdworkspace	❌	https://github.com/InseeFrLab/rjdworkspace
JDCruncheR	❌	https://github.com/InseeFr/JDCruncheR

The packages installation code is below:

```
# If remotes is not installed
# install.packages("remotes")

install.packages("RJDemetra")
remotes::install_github("InseeFrLab/rjdworkspace")
remotes::install_github("InseeFr/JDCruncheR")
```

3.3.2 In version 3

Currently no package of the version 3 are on CRAN. To install them, you need to go through Github:

```
# If remotes is not installed
# install.packages("remotes")

remotes::install_github("palatej/rjdemetra3")

remotes::install_github("palatej/rjd3toolkit")
remotes::install_github("palatej/rjd3modelling")
remotes::install_github("palatej/rjd3sa")
remotes::install_github("palatej/rjd3arima")
remotes::install_github("palatej/rjd3x13")
remotes::install_github("palatej/rjd3tramoseats")
remotes::install_github("palatej/rjdemetra3")
remotes::install_github("palatej/rjdfilters")
remotes::install_github("palatej/rjd3sts")
remotes::install_github("palatej/rjd3highfreq")
remotes::install_github("palatej/rjd3stl")
remotes::install_github("palatej/rjd3bench")

remotes::install_github("AQLT/ggdemetra3")
```

3.3.3 AUS case

To install a package on AUS, you can't use the function `install_github()`. Then if the package is not on CRAN, it must be downloaded at the binary format (`.zip`)

For the package **JDCruncheR**, the installing code is:

```
install.packages("path/.../JDCruncheR_0.2.4.tar.gz",
                 repos = NULL, type = "binary")
```

4 Environment variables

4.1 In Rstudio

4.1.1 Proxy

ℹ On Insee computer, you need to **configure** the proxy and parameters of Software localisation under Rstudio. You should:

- **Launch** the code `file.edit("~/.Renviron")`
- **Add** the parameters (news lines):

```
http_proxy = http://proxy-rie.http.insee.fr:8080/
https_proxy = http://proxy-rie.http.insee.fr:8080/
```

- **Save** and **close** the file

4.1.2 Java version

If a new Java version has been installed, you should inform Rstudio of the Java installation localisation. For this, as to parameter the proxy:

- **Launch** the code `file.edit("~/.Renviron")` to edit the environment variables
- **Add** the parameters:

```
JAVA_HOME = "C:/Users/Software/Java17/jdk17"
```

- **Save** and **close** the file

4.1.3 PATH

The environment variable `PATH` in **R** is used to indicate to **R** where to find the executable files.

When you install a new software (for example JDemetra+, Rtools, Java…) that Rstudio uses, you have to modify this environment variable:

- **Get** the actual value of the variable `PATH` via the **R** command `Sys.getenv("PATH")` (Rstudio returns a succession of adresses as `C:/WINDOWS/system32;C:/WINDOWS`)
- **Copy paster** this value after `PATH =` and add the paths towards the folder \bin\ (binary) of the software newly installed, by separating them with semicolon (without space before or after). For the Rtools installation, the path is C:\rtools42\mingw64\bin (depending on where Rtools was installed). You have to add `C:\\rtools42\\mingw64\\bin` or `C:/rtools42/mingw64/bin` (In **R**, `\` is a special character, so you have to replace the `\` by `/` or by `\\`). The path become `C:/WINDOWS/system32;C:/WINDOWS;C:/rtools42/mingw64/bin`.
- **Modify** the variable with the function `Sys.setenv()`. For Rtools, the command to launch is:

```
Sys.setenv(PATH = "C:/WINDOWS/system32;C:/WINDOWS;C:/rtools42/mingw64/bin")
```

ℹ NB: Generally a 32 bits version and a 64 bits version are available for downloading and installing a software. You need to check your processor type of your operating system to choose the right folder to download:

Version	Keywords
64 bits	64 x64
32 bits	32 x86

More informations on the variable `PATH` via the page https://java.com/fr/download/help/path.xml.

5 Verifications

To ensure that everything works fine, you can launch some example of **R** code and check that there is no error:

```
library("RJDemetra")

myseries <- ipi_c_eu[, "FR"]
x13_model <- x13(myseries) # X-13ARIMA method
ts_model <- tramoseats(myseries) # TRAMO-SEATS method

# Basic plot with the original series, the trend and the SA series
plot(x13_model, type_chart = "sa-trend")
```

To check the Java version we are using on **R**, you can try to install and use the package **rJava** and launch the command below:

```
# If rJava is not installed
install.packages("rJava")
```

If the installation of **rJava** return an error, it means that Java was incorrectly installed or incorrectly configured on **R**. You need to get back to the section Environment variables.

This block of code tests the Java version with which **R** works:

```
library("rJava")
.jinit()
.jcall("java/lang/System", "S", "getProperty", "java.runtime.version")
```

Finally, you can consult the Java version installed with which Windows works (it doesn't matter to us):

```
system("java -version")
```

6 Optionnal installations

Some supplementary installations are optionnal (that is they are no mandatory but bring external features):

- Miktek to produce PDF document with Latex
- Rtools to develop **R** packages and compile the code

7 Problems you may encounter

7.1 Problems installing R packages

If you get the following error while installing **R** packages:

```
install.packages("RJDemetra")
```

```
## Error in eval(expr, envir, enclos): Erreur: the chargement a échoué
## Exécution arrêtée
## *** arch - x64
```

The problem doesn't come from Java but from the **R** package. By default, the package is installed from a "source" file, it means that the package is recompiled. For some computing reasons, when compiling by default, **R** uses the system (Windows) parameters (which doesn't have necessarily have the correct Java version).

There are two solutions:

- Compile the package by installin from the binary file:

  ```
  install.packages("RJDemetra", type = "binary")
  ```

- Specify that we want to use the local parameters:

```
install.packages("RJDemetra", type = "source", INSTALL_opts = "--no-multiarch")
```

ℹ️ More informations: https://github.com/jdemetra/rjdemetra/wiki/Installation-manual

7.2 The command `library("RJDemetra")` returns an error message

The package **RJDemetra** requires Java version 8 or higher to work. If another package has been loaded before **RJDemetra** via the function `library()` and which doesn't requires an updated Java version, then an old Java version will be used during all the session (**R** is refractory to in-session version change). In case of using **RJDemetra** in a program, you have to specify at the very beginning of the program that **R** must use Java version 8, via the command:

```
Sys.setenv(JAVA_HOME = "C:/Users/Software/Java17/jdk17") # Là où is installé java
```

or load **RJDemetra** first

```
# At the beginning of program
library("RJDemetra")
```

Else you have to restart a new **R** session.

7.3 `Error array index = -1`

The message of the type `Error array index = -1` tells that an auxiliary variable is not found. It can be calendar regressor as other user defined variables (Easter effect, PSO = pure seasonal outlier…).