R tools for SA in production in V3

with JDemetra R packages

Anna Smyk and Tanguy Barthelemy - Insee

TSACE Webinar - 28th February, 2024

Table of Contents I

Introduction

This presentation will focus on the V3 production tools which are now replacing totally the tools in V2. The aim is to present new effective tools for seasonnally adjusted statistics.

Lexicon

Word	Abbreviation
Workspace	WS
SA-Processing (or Multiprocessing)	SAP
Java pointer to a SAP	jsap
SA-Item	SAI
Java pointer to a SAI	jsai
Version 2	V2
Version 3	V3

Context

In a production chain, with one or several workspaces, there are benefits in using workspaces and R packages in conjunction.

So in this presentation we will see how to:

- Create a workspace from scratch
- Load an existing workspace
- Save a workspace to a new location
- Update a workspace with **new SA-Item** created in R
- Merge two workspaces, according to the name of the series, as you would merge two data tables
- Update the link between **raw data** and a **workspace**

The workspace

What are we working on?

[Reminder] the workspace is the **essential** data structure to be able to use **the graphical interface (GUI)** and **the cruncher**.

- The workspace is stored with a XML file and a folder.
- This imposes constraints on modifying the data, the metadata and estimation parameters stored there.
- Limits of working in R –> creation of new Java routines to access Java core in R

2 interesting properties:

- reading by GUI (1)
- refreshing (with new raw data for example) by GUI or cruncher (2)

R

By using R packages to modify workspaces, this allows us:

- to **automate** potential manual operations for large datasets (modify and customize JD+ objects more massively)
- to **immerse** JD+ results in the R world:
  - to integrate the seasonal adjustment steps into a production chain in R (which is not necessarily dependent)
  - to display the JD+ objects in different forms (plots, tables...)
  - to combine with the wide variety of existing R packages (functions...)

Version 3

Goal :

- Update the tools from version 2 to version 3
- Present the new packages **rjdemetra3** and **rjd3providers**

Warning

A WS created in V3 cannot be read in V2. So the switch to V3 is definitive !

Presentaion of the packages

We will use the packages **rjdemetra3** and **rjd3providers**.

The packages are available here:

- **rjdemetra3** on https://github.com/rjdemetra/rjdemetra3
- **rjd3providers** on https://github.com/rjdemetra/rjd3providers.

Installation

To install the packages use the following code:

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

remotes::install_github("https://github.com/rjdemetra/rjdemetra3")
remotes::install_github("https://github.com/rjdemetra/rjd3providers")
```

To load the packages use the following code:

```
library("rjdemetra3")
library("rjd3providers")
```

Application

Automating operations on workspaces

Goals:

- produce and reproduce the workspace structure without the JDemetra+ GUI but with R
- keep the reading, writing and refreshing properties both by the GUI and by the cruncher
- enable dynamic updates of physical workspaces with R

Instant reading: loading workspaces

With **rjdemetra3** it's possible to:

- import in R physical workspaces
- get any JD+ object (SAP, SAI, spec, output...) automatic and chain reading –> many series and quickly
- make faster comparisons between WS

2 classes of objects

In R there are 2 sort of objects created by the RJD3 packages:

- **java pointers** (unreadable by humans but can be manipulated by machine)
- R object with a classical **list structure** (readable by humans but not efficient in computing)

Coding with **rjdemetra3**

To get the Java pointer towards your ws, use the function `.jws_open`. To make it readable, you have yo add the function `read_workspace`.

For example, the following code open the workspace and then read the content and write it in an R object:

```
# Get the Java WS
jws_object <- .jws_open(file = "ws_example.xml")
# Get the readable WS
ws_r <- read_workspace(jws = jws_object, compute = TRUE)
```

Other JD+ object

To manipulate SAP (and respectively SAI), the equivalent functions are `.jws_sap` and `read_sap` (respectively `.jsap_sa` and `.jsa_read`)

To continue the previous example:

```
# Get the Java SAP
jsap_object <- .jws_sap(jws = jws_object, idx = 1L)
# Get the readable SAP
sap_r <- read_sap(jsap = jsap_object)
# Get the Java SAI
jsai_object <- .jsap_sa(jsap_object, idx = 1L)
# Get the readable SAI
sai_r <- .jsa_read(jsai_object)
```

Creation and saving of workspaces

- **Creation**

rjdemetra3 has a collection of functions to create a workspace and multiprocessing:

- create a *virtual* workspace: `.jws_new`
- create a *virtual* multiprocessing: `.jws_sap_new`

virtual object = object only existing in the R session and not written in our folder

- **Saving**

For the export part, the function `save_workspace` exports in a new location and creates a *real* workspace (with folder and XML files).

Filling workspaces

To aliment a WS with series, the function `add_sa_item` is very useful.

You can add a new SA-Item in a SAP with different possibilities:

- With a R-created SA-Item (using `rjd3x13::x13` and `rjd3tramoseats::tramoseats` functions)

```
# add a SA-Item created with R
sa_x13 <- rjd3x13::x13(rjd3toolkit::ABS[, 1])
sa_ts <- rjd3tramoseats::tramoseats(rjd3toolkit::ABS[, 2])

add_sa_item(jsap = new_jsap, name = "ABS_1", x = sa_x13)
add_sa_item(jsap = new_jsap, name = "ABS_2", x = sa_ts)
```

- With a specification and a raw series

```
# add a raw series with a spec created in R
add_sa_item(
    jsap = new_jsap,
    name = "ABS_3",
    x = rjd3toolkit::ABS[, 3],
    spec = rjd3x13::x13_spec(name = "RSA5c")
)
add_sa_item(
    jsap = new_jsap,
    name = "ABS_4",
    x = rjd3toolkit::ABS[, 4],
    spec = rjd3tramoseats::tramoseats_spec(name = "rsafull")
)
```

- With another SAI (from the same or another SAP)

```
# add a SA_Item from another workspace
add_sa_item(jsap = new_jsap, name = "ABS_4", x = jsai_from)
```

The function `replace_sa_item` allows the user to replace an existing SA-Item with a new one.

```
# replace the first SAI from jsap_object with jsai_from
replace_sa_item(
    jsap = jsap_object,
    jsa = jsai_from,
    idx = 1L
)
```

You can also remove SAI from WS with the functions `remove_sa_item` and `remove_all_sa_item`

```
remove_sa_item(jsap = jsap_object, idx = 5L)
remove_sa_item(jsap = jsap_object, idx = 6L)

remove_all_sa_item(jsap = jsap_object)
```

Wrangling several workspaces

When we are working with different workspaces it's important to know we how to make them communicate.

- During an annual campaign, we want to merge a reference workspace (current) with an automatic workspace (for example according to a score created using **JDCruncheR**) and be able to **crunch the result**.
- Use an empty WS template (with specific settings) and fill it with existing SAI from other WS

Wrangling workspaces with rjdemetra3

So the goal is to **merge WS**. We will use the useful function `transfer_series`.

This function takes two jsap object as argument and transfers the selected_series argument from one to the other.

- The series (SAI) are identified by **their name**
- Origin and destination SAP are identified by **their java pointer**

Merging 2 WS with **rjd3providers** - Example

```
transfer_series(
    jsap_from = jsap_object,
    jsap_to = new_jsap,
    selected_series = c("RF0899", "RF1039", "RF1041")
)
```

Initialising production chain: update raw data path

Description:

My data has changed, how can I ensure that my workspaces are still functional?

- New path?
- New name?
- New file structure?

The raw data path is stored in the metadata of the SAI.

The metadata consists (*mainly*) of :

- Date and time of last modification
- **Path and source** of the raw data file
- Comments

Update raw data path in V3

The functions `XXX_update_path()` (from package **rjd3providers**) updates the data path according to the file extension (XXX for spreadsheet or txt).

Arguments:

- `jws`: the Java pointer to our WS
- `new_path`: the new path to the raw data.
- `idx_sap`: the SAP index containing the series to update. (Optional)
- `idx_sai`: the index of the SAI element of the series to update. (Optional)

Update raw data path in V3

A code example:

```
txt_update_path(
    jws = jws_object,
    new_path = path_csv,
    idx_sap = 1L
)
spreadsheet_update_path(
    jws = jws_object,
    new_path = path_xlsx,
    idx_sap = 2L
)
```

rjd3providers: a data explorer

rjd3providers also provides functions to explore your data files:

Function	Description
`XXX_content`	Get a description of a file (colnames, rownames, sheet names...)
`XXX_data`	Get all the information of a file
`XXX_series`	Get all the information of a file of a specified sheet and series

For example:

```
txt_content(file = path_csv, delimiter = "SEMICOLON", fmt.date = "dd/MM/yyyy")
txt_data(file = path_csv, delimiter = "SEMICOLON", fmt.date = "dd/MM/yyyy")
spreadsheet_data(file = path_xlsx, sheet = 1L, cleanMissings = TRUE)
spreadsheet_series(file = path_xlsx, sheet = 1L, series = 3L)
```

Summary of available functionalities

With the combination of **rjdemetra3** and **rjd3providers**, there are a lot of features to handle workspaces with R.

In a production process, it is now easier in version 3 to:

- Create (100% with R) functional WS (usable with the GUI)
- Copy, merge, update and save workspaces
- Update one or more WS with new raw data (or with data that has been moved)

than doing it by hand with the GUI.

Useful links

Thank you for your attention!

Documentation of the used packages:

- Packages :
  - **rjdemetra3** : https://github.com/rjdemetra/rjdemetra3
  - **rjd3providers** : https://github.com/rjdemetra/rjd3providers

- GitHub content of the webinar:
  https://github.com/annasmyk/TSACE_rjd3_webinar_feb2024
- On-line documentation of JDemetra+:
  https://jdemetra-new-documentation.netlify.app/
- Our blog: https://jdemetra-universe-blog.netlify.app/
- YouTube channel: https://www.youtube.com/@TSwithJDemetraandR

Our GitHubs :

- Anna SMYK https://github.com/annasmyk
- Tanguy BARTHELEMY https://github.com/TanguyBarthelemy