Using the cruncher with JDCruncheR pdf

2022-11-08

Contents

About the cruncher **1**

Generating the settings file with `create_param_file()` **1**

Launching the cruncher **3**

With the current version of the JDCruncheR package (0.2.4), you can:

- easily launch the JDemetra+ cruncher (JWSACruncher) to update a workspace and export the results;
- generate a *quality report* to synthesise seasonal adjustment diagnostics. It can be used to spot the most problematic series which will require a finer analysis. This is most useful when dealing with a great number of series, which renders impossible the examination of every diagnostic for every series in a reasonable time.

About the cruncher

To prevent the package from being too voluminous, it does not contain the JDemetra+ cruncher. Said cruncher can be downloaded here: https://github.com/jdemetra/jdemetra-app/releases. For more information on the cruncher installation and its set up with a portable version of Java, please refer to the JDCruncheR wiki: https://github.com/InseeFr/JDCruncheR/wiki/Installation-&-configuration.

The JDemetra+ cruncher requires three elements to function:

- a settings file containing the refresh policy parameters to apply to the workspace;
- a valid JDemetra+ workspace;
- the path to the cruncher directory.

In the JDCruncheR package, three functions are cruncher-related:

- `create_param_file()` generates a settings file;
- `cruncher()` crunches a workspace from a pre-generated settings file;
- `cruncher_and_param()` both generates the settings file and crunches the workspace. With this function, some of the cruncher output can also be customised.

Generating the settings file with `create_param_file()`

The arguments of the function `create_param_file()` are described in the JDemetra+ cruncher wiki (https://github.com/jdemetra/jwsacruncher/wiki). The three main arguments are:

1. `policy`, the refresh policy:

Table 1: The refresh policies

Option in JDemetra+	Cruncher option	Description
Current adjustment (AO approach)	current	The ARIMA model, outliers and other regression variables are not re-identified, and the values of all associated coefficients are fixed. All new observations are classified as additive outliers and corresponding coefficients are estimated during the regression phase. The transformation type remains unchanged..
Partial concurrent adjustment		
Fixed model	fixed	The ARIMA model, outliers and other regression variables are not re-identified and the values of all coefficients are fixed. The transformation type remains unchanged.
Estimate regression coefficients	fixedparameters	The ARIMA model, outliers and other regression variables are not re-identified. The coefficients of the ARIMA model are fixed but the regression variables coefficients are re-estimated. The transformation type remains unchanged.
Estimate regression coefficients + Arima parameters	parameters (default policy)	The ARIMA model, outliers and other regression variables are not re-identified. All coefficients of the RegARIMA model are re-estimated, for regression variables and ARIMA parameters. The transformation type remains unchanged.
Estimate regression coefficients + Last outliers	lastoutliers	Outliers in the last year of the sample are re-identified. All coefficients of the RegARIMA model, regression variables and ARIMA parameters, are re-estimated. The transformation type remains unchanged.
Estimate regression coefficients + all outliers	outliers	All outliers are re-identified. All coefficients of the RegARIMA model, regression variables and ARIMA parameters, are re-estimated. The transformation type remains unchanged.
Estimate regression coefficients + Arima model	stochastic	Re-identification of the ARIMA model, outliers and regression variables, except the calendar variables. The transformation type remains unchanged.
Concurrent		
Concurrent	complete or concurrent	Complete re-identification of the whole RegARIMA model, all regression variables and ARIMA model orders.

2. `matrix_item`, a list containing the names of the parameters to export. By default, such names are those contained in the `default_matrix_item` option. Thus, the user can either modify the `default_matrix_item` option or the `matrix_item` option:

```
library("JDCruncheR")
# To see the default parameters:
getOption("default_matrix_item")
# To customise the parameter selection (here, only the information criteria are exported):
options(default_matrix_item = c("likelihood.aic",
                                "likelihood.aicc",
                                "likelihood.bic",
                                "likelihood.bicc"))
```

3. `tsmatrix_series`, a list containing the names of the parameters to export. By default, such names are those contained in the `default_tsmatrix_series` option. Thus, the user can either modify the

`default_tsmatrix_series` option or the `tsmatrix_series` option:

```
# To see the default parameters:
getOption("default_tsmatrix_series")
# To customise the parameter selection (here, only the seasonaly adjusted series and its previsions are
options(default_tsmatrix_series = c("sa", "sa_f"))
```

To visualise all parameters that can be used to customise these options, enter `?create_param_file` (in R).

Here are some use cases for the function `create_param_file()`:

```
# A .param parameters file will be created in D:/, containing the "lastoutliers" refresh policy
# and default values for the other parameters
create_param_file(dir_file_param = "D:/",
                  policy = "lastoutliers")

# To customise the "default_matrix_item" and "default_tsmatrix_series" options
# to only export the information criteria, the adjusted series and its forecast:
create_param_file(dir_file_param = "D:/",
                  policy = "lastoutliers",
                  matrix_item = c("likelihood.aic", "likelihood.aicc",
                                  "likelihood.bic", "likelihood.bicc"),
                  tsmatrix_series = c("sa", "sa_f"))
```

Launching the cruncher

To crunch a workspace with the functions `cruncher()` or `cruncher_and_param()`, the paths to the cruncher and the workspace must be specified via two parameters, `cruncher_bin_directory` and `workspace`.

Once declared, the path to the cruncher (stored in the parameter `cruncher_bin_directory`) is valid for all cruncher executions. The path must refer to the *jwsacruncher.bat* file location, within the "bin" installation folder. For example, if the cruncher is installed in `D:\jdemetra-cli-2.2.3`, the *jwsacruncher.bat* file will be in `D:\jdemetra-cli-2.2.3\bin`. The code to declare this path via the `cruncher_bin_directory` parameter is:

```
options(cruncher_bin_directory = "D:/jdemetra-cli-2.2.3/bin/")
```

If no path to a workspace was specified, a window opens for a manual workspace selection.

The function `cruncher_and_param()` creates a temporary parameters file by calling `create_param_file()`, which is then used to by the function `cruncher()` to crunch the workspace. In addition to the parameters used by these two functions, `cruncher_and_param()` carries the option `rename_multi_documents` with which the output folders can be renamed after the SAProcessings' names as displayed in the JDemetra+ interface (by default, `rename_multi_documents = TRUE`: the output files are renamed). Here are some use cases:

```
# Code to update the "ipi" workspace stored in D:/seasonal_adjustment/, with the refresh policy "lastou
# All other create_param_file() parameters are default ones. In particular, the exported parameters are
# "default_matrix_item" and "default_tsmatrix_series", and the output folder is D:/seasonal_adjustment/
cruncher_and_param(workspace = "D:/seasonal_adjustment/ipi.xml",
                   rename_multi_documents = FALSE,
                   policy = "lastoutliers")

# Example of customisation of the parameter "output":
cruncher_and_param(workspace = "D:/seasonal_adjustment/ipi.xml",
                   output = "D:/cruncher_results/",
                   rename_multi_documents = FALSE,
                   policy = "lastoutliers")
```

```
# Here, we explicitely have "rename_multi_documents = TRUE" (which is also the default value) to rename
# after the SAProcessings as displayed in the JDemetra+ interface.
# With parameter "delete_existing_file = TRUE", all pre-existing versions of such folders are deleted b
cruncher_and_param(workspace = "D:/Campagne_CVS/ipi.xml",
                   rename_multi_documents = TRUE,
                   delete_existing_file = TRUE,
                   policy = "lastoutliers")

# To see all the function parameters:
?cruncher_and_param
```