Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

RJDemetra: A R Interface To JDemetra+ Seasonal Adjustment Software

Anna Michalek
European Central Bank

Alain Quartier-La-Tente
Insee

Abstract

The abstract of the article.

Keywords: R, seasonal adjustment, time series.

1. Introduction

The package **RJDemetra** provides a R interface to the seasonal adjustment software JDemetra+. Note that, JDemetra+ being implemented in Java, **RJDemetra** relies on the **rJava** package and Java SE 8 or later version is required. The two leading seasonal adjustment methods TRAMO/SEATS+ and X-12ARIMA/X-13ARIMA-SEATS can be used with all the specifications defined in JDemetra+.

This article is structured as following. In the first section the .. is presented.

1.1. Seasonal adjustment in brief

The **first step** of seasonal adjustment, both in X-12ARIMA/X-13ARIMA-SEATS and TRAMO-SEATS+, consists of pre-adjusting the time series by removing from it the deterministic effects and estimating missing observations. Among deterministic effects, we distinguish outliers, calendar and regression effects. In this step, also forecasts and backcasts of the pre-adjusted series are estimated which allows applying linear filters at both ends of the series in the second step of the seasonal adjustment. The pre-adjustment, linearization, of the input series is achieved with a **RegARIMA** model (model with ARIMA errors) as specified below.

$$z_t = y_t\beta + x_t$$

where

- z_t - is the original series;
- $\beta = (\beta_1, ..., \beta_n)$ - a vector of regression coefficients;
- $y_t = (y_{1t}, ..., y_{nt})$ - n regression variables (outliers, calendar effects, user-defined variables);
- x_t - a disturbance that follows the general ARIMA process:
- $\phi(B)\delta(B)x_t = \theta(B)a_t$; $\phi(B), \delta(B)$ and $\theta(B)$ are the finite polynomials in B; a_t is a white-noise variable with zero mean and a constant variance.

The polynomial $\phi(B)$ is a stationary autoregressive (AR) polynomial in B, which is a product of the stationary regular AR polynomial in B and the stationary seasonal polynomial in B^s:

$$\phi(B) = \phi_p(B)\Phi_{bp}(B^s) = (1 + \phi_1 B + ... + \phi_p B^p)(1 + \Phi_1 B^s + ... + \Phi_{bp} B^{bps}$$

where:

- p - number of regular AR terms (in the package and in JDemetra+ $p \leq 3$);
- bp - number of seasonal AR terms (in the package and in JDemetra+ $bp \leq 1$);
- s - number of observations per year (frequency of the time series).

The polynomial $\theta(B)$ is an invertible moving average (MA) polynomial in B, which is a product of the invertible regular MA polynomial in B and the invertible seasonal MA polynomial in B^s:

$$\theta(B) = \theta_q(B)\Theta_{bq}(B^s) = (1 + \theta_1 B + ... + \theta_q B^q)(1 + \Theta_1 B^s + ... + \Theta_{bq} B^{bqs})$$

where:

- q - number of regular MA terms (in the package and in JDemetra+ $q \leq 3$);
- bq - number of seasonal MA terms (in the package and in JDemetra+ $bq \leq 1$);

The polynomial $\delta(B)$ is the non-stationary AR polynomial in B (unit roots):

$$\delta(B) = (1 - B)^d (1 - B^s)^{d_s}$$

where:

- d - regular differencing order (in the package and in JDemetra+ $d \leq 1$);
- d_s - seasonal differencing order (in the package and in JDemetra+ $d_s \leq 1$);

In the **second part** of seasonal adjustment, called the **decomposition**, the pre-adjusted series is decomposed into the following components: trend-cycle (`t`), seasonal component (`s`) and irregular component (`i`). The decomposition can be:

- additive (`y = t + s + i`)
- multiplicative (`y = t * s * i`)
- log-additive (`log(y) = log(t)+log(s)+log(i)`) or

- pseudo-additive (`y = t*(s+i-1)`)

The last two decompositions are available only under X13.

The method of decomposing the pre-adjusted series differs between TRAMO-SEATS+ and X-12ARIMA/X-13ARIMA. In TRAMO-SEATS+, SEATS ("Signal Extraction in ARIMA Time Series") decomposes the observed series with a ARIMA-model based method. Whereas in X-12ARIMA/X-13ARIMA, the X11 algorithm decomposes the time series by means of linear filters. More information on the TRAMO-SEATS+ method can be found on the Bank of Spain website (link) and on X-12ARIMA/X-13ARIMA, on the U.S. Census Bureau website.

As a result of seasonal adjustment, the final seasonally adjusted series (`sa`) shall be free of seasonal and calendar-related movements.

More details on the methodlgy used in JDemetra+ can be found in the JDemetra+ manuals and user guides available at link.

2. RJDemetra basics

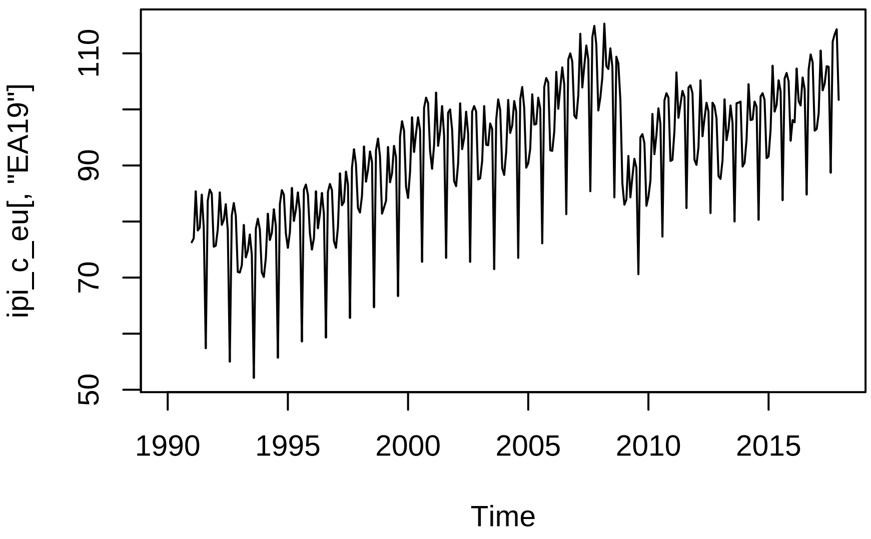
The **RJDemetra** package alows to:

- create and modify model specifications
- create and modify models
- import/export JDemetra+ workspaces

2.1. Dataset

In this package the sts_inpr_m database of Eurostat is included, which contains monthly industrial production indices in manufacturing in the European Union. It contains 37 time series from january 1990 to december 2017 which are considered to be affected by seasonal and working day effects. The data is a `ts` object and can be accessed using the `ipi_c_eu` object. The following snippet of code plots the industrial production index of the euro aera (EA19):

```
R> library(RJDemetra)
R> plot(ipi_c_eu[, "EA19"])
```



3. Estimate a pre-defined RegARIMA and SA model

As in JDemetra+, the **RJDemetra** package allows to perform seasonal adjustment using pre-defined model specifications. The specifications are separately defined for TRAMO-SEATS and X-13ARIMA-SEATS estimation methods. It is also possible to perform only the first step of seasonal adjustment; the RegARIMA estimation. The pre-defined model specifications are described in tables 1 and 2. They are identical for pre-adjustment (column 1) and for seasonal adjustment (column 2). The pre-defined specifications correspond to most commonly used specifications and users are recommended to start their analysis with one of them. In section 5 it is presented how to modify model specifications, including the possibility to incorprate user-defined regressors.

The below code presents how to perform an estimation, with pre-defined specifications, of:

- RegARIMA
  - X-13ARIMA method: `regarima_def_x13(series, spec = c("RG5c", "RG0", "RG1", "RG2c", "RG3","RG4c"))`
  - TRAMO-SEATS method: `regarima_def_tramoseats(series, spec = c("TRfull", "TR0","TR1", "TR2","TR3", "TR4","TR5"))`
- Seasonal adjustment
  - X-13ARIMA method: `x13_def(series, spec = c("RSA5c", "RSA0", "RSA1", "RSA2c", "RSA3","RSA4c"), userdefined = NULL)`
  - TRAMO-SEATS method: `tramoseats_def(series, spec = c("RSAfull", "RSA0", "RSA1", "RSA2", "RSA", "RSA4", "RSA5"), userdefined = NULL)`

Table 1: Pre-defined specification for TRAMO and TRAMO-SEATS

Specification								
TRAMO	TRAMO-SEATS	Trans-formation	Pre-adjust-ment for leap-year	Working days	Trading days	Easter effect	Outliers	ARIMA model
TR0	RSA0	no	no	no	no	no	no	(0,1,1)(0,1,1)
TR1	RSA1	test	no	no	no	no	test	(0,1,1)(0,1,1)
TR2	RSA2	test	no	test	no	test	test	(0,1,1)(0,1,1)
TR3	RSA3	test	no	no	no	no	test	AMI
TR4	RSA4	test	no	test	no	test	test	AMI
TR5	RSA5	test	no	no	yes	test (Standard)	test	AMI
TRfull (default)	RSAfull (de-fault)	test	yes	no	test	test (Include Easter)	test	AMI

Table 2: Pre-defined specification for RegARIMA and X-13ARIMA-SEATS

Specification								
RegARIMA	X-13ARIMA-SEATS	Trans-formation	Pre-adjust-ment for leap-year	Working days	Trading days	Easter effect	Outliers	ARIMA model
RG0	X11	no	no	no	no	no	no	(0,1,1)(0,1,1)
RG1	RSA1	test	no	no	no	no	test	(0,1,1)(0,1,1)
RG2c	RSA2c	test	test	test	no	test	test	(0,1,1)(0,1,1)
RG3	RSA3	test	no	no	no	no	test	AMI
RG4c	RSA4c	test	test	test	no	test	test	AMI
RG5c (default)	RSA5 (default)	test	test	no	test	test	test	AMI

```
R> library(RJDemetra)
R> myseries <- ipi_c_eu[, "EA19"]
R>
R> regx13 <- regarima_def_x13(myseries, spec = "RG5c")
R> regts <- regarima_def_tramoseats(myseries, spec = "TRfull")
R> sax13 <- x13_def(myseries, spec = "RSA5c", userdefined = NULL)
R> sats <- tramoseats_def(myseries, spec = "RSAfull", userdefined = NULL)
```

4. SA object structure

In the previous section it was presented how to run a RegARIMA and complete seasonal adjustment estimation with pre-defined model specifications. In this section the outcome will be described in detail.

As a result of seasonal adjustment estimation (e.g. function `x13_def` or `tramoseats_def`) a S3 class object (`sa_object`) is created. It has a class `c("SA","X13")` or `c("SA","TRAMO_SEATS")` depending on the used estimation method. The `sa_object` consits of lists of S3 class sub-objects. For each of the class `print, plot` methods are defined. The complete structure of the `sa_object` is presented in table 3. The first column gives the name of `sa_object` sub-components, the second the level of the sub-components, the third their type, and the fourth and fifth the name of the new created S3 classe (if any). Where the forth column corresponds to the case when the estimation is done with X-12ARIMA/X-13ARIMA and fifth when estimated with TRAMO-SEATS+. In general, the `sa_object` contains the following five objects: **regarima**, **decomposition**, **final**, **diagnostics** and **user_defined**. Independently which of the two methods is used the regarima, final and diagnostics objects contain the same components, though with different classes (see column 4 and 5). Whereas, the object decomposition differs for the two methods. The object user_defined is empty unless additional output was requested by the user (see next sub-sections). Finally, when estimating RegARIMA only the regarima object is created.

Table 3: SA object structure

			When adjusted with:	
			x13/x13_def	*tramoseats/tramoseats_def*
Object	Level	Type	Class	Class
sa_object	0	list	SA, X13	SA, TRAMO_SEATS
regarima	**1**	**list**	**regarima, X13**	**regarima, TRAMO_SEATS**
specification	2	list		
estimate	3	data.frame		
transform	3	data.frame		
regression	3	list		
userdef	4	list		
specification	5	data.frame		
outliers	5	data.frame or NA(empty)		
variables	5	list		
series	6	mts, ts, matrix or NA(empty)		
description	6	data.frame or NA(empty)		
trading.days	4	data.frame		
easter	4	data.frame		
outliers	3	data.frame		
arima	3	list		
specification	4	data.frame		
coefficients	4	data.frame or NA(empty)		
forecast	3	data.frame		
span	3	data.frame		
arma	2	vector - numeric		
arima.coefficients	2	matrix		
regression.coefficients	2	matrix		
loglik	2	matrix		
model	2	list		
spec_rslt	3	data.frame		
effects	3	mts, ts, matrix		
residuals	2	ts		
residuals.stat	2	list		
st.error	3	numeric		
tests	3	data.frame	regarima_rtests, data.frame	
forecast	2	mts, ts, matrix		
decomposition	**1**	**list**	**decomposition_X11**	
specification	2	data.frame	X11_spec, data.frame	
mode	2	character		
mstats	2	matrix		
si_ratio	2	mts, ts, matrix		
s_filter	2	vector - character		
t_filter	2	character		
decomposition	**1**	**list**		**decomposition_SEATS**
specification	2	data.frame	seats_spec, data.frame	
mode	2	character		
model	2	list		
model	3	matrix or empty list		
sa	3	matrix or empty list		

trend	3	matrix or empty list	
seasonal	3	matrix or empty list	
transitory	3	matrix or empty list	
irregular	3	matrix or empty list	
linearized	2	mts, ts, matrix	
components	2	mts, ts, matrix	
final	**1**	**list**	**final**
series	2	mts, ts, matrix	
forecasts	2	mts, ts, matrix	
diagnostics	**1**	**list**	**diagnostics**
variance_decomposition	2	data.frame	
combined_test	2	list	combined_test
tests_for_stable_seasonality	3	data.frame	
combined_seasonality_test	3	character	
residuals_test	2	data.frame	
user_defined	**1**	**list**	**user_defined**

4.1. Regarima

Here we can also present the output: print and graphs.

```
R> library(RJDemetra)
R> myseries <- ipi_c_eu[, "EA19"]
R> sax13 <- x13_def(myseries, spec = "RSA5c", userdefined = NULL)
R> sats <- tramoseats_def(myseries, spec = "RSAfull", userdefined = NULL)
R> ## PRINT THE RESULTS:
R> sax13$regarima

y = regression model + arima (1, 1, 2, 0, 1, 1)
Log-transformation: no
Coefficients:
          Estimate Std. Error
Phi(1)     -0.7695      0.117
Theta(1)   -1.0644      0.119
Theta(2)    0.3331      0.056
BTheta(1)  -0.5263      0.051

             Estimate Std. Error
Monday       -0.27760      0.103
Tuesday       0.01418      0.102
Wednesday     0.29139      0.103
Thursday     -0.36725      0.102
Friday        0.12606      0.102
Saturday      0.36548      0.103
Leap year     0.24961      0.316
AO (1-2016)   3.58591      0.837
TC (9-2008)  26.20114      3.037
LS (9-2008) -19.99432      2.470
AO (9-2008)  -6.10726      1.458
```

```
Residual standard error: 1.125 on 311 degrees of freedom
Log likelihood = -479.9, aic = 991.8 aicc = 993.7, bic(corrected for length) = 0.5122

R> ## PLOT THE RESULTS:
R> #plot(sax13$regarima)
```

4.2. Decomposition

4.3. Final

4.4. Diagnostics

4.5. user defined

5. Model specification: creation and modification

5.1. X13

5.2. TRAMOSEATS

5.3. Regarima

5.4. Wrong specifications corrections

Parler des corrections automatiques ?

6. Manipulate JDemetra+ workspaces

RJDemetra allows to interact with JDemetra+ workspace that can be openned by the software. A workspace includes :

- The XML file that enables the user to import the workspace to JDemetra+ and to display it content;

- A folder containing several sub-folfders that correspond to the different types of items created by the user.

Each workspace can contain several multi-processings and each multi-processing stores the results of the seasonal adjustment procedure performed with the TRAMO/SEATS or X-13ARIMA-SEATS methods.

Export models to workspace allows to store easily the seasonal adjustment models, to change the specifications with the JDemetra+ graphical interface and to give models to non R users (à reformuler).

6.1. Export a workspace

Four functions have to be used to export models:

- `new_workspace()` to create a workspace;
- `new_multiprocessing()` to create a multi-processing in a workspace;
- `add_sa_item()` to add a seasonal adjustment model to a multi-processing;
- `save_workspace()` to export the workspace.

The following command export the seasonal adjustment models compute by TRAMO/SEATS+ and X-13ARIMA-SEATS:

```
R> myseries <- ipi_c_eu[, "EA19"]
R> sa_x13 <- x13_def(myseries)
R> sa_ts <- tramoseats_def(myseries)
```

To create a workspace and a multi-processing names "MP-1":

```
R> wk <- new_workspace()
R> new_multiprocessing(wk, name = "MP-1")
```

The two models will be added in the multiprocessing "MP1": the name of the seasonal adjustment model computed with X-13ARIMA-SEATS will be "SA with X13" and the one with TRAMO/SEATS+ will be SA with TramoSeats".

```
R> add_sa_item(wk, multiprocessing = "MP-1",
R+             sa_obj = sa_x13, name =  "SA with X13")
R> add_sa_item(wk, multiprocessing =  "MP-1",
R+             sa_obj = sa_ts, name = "SA with TramoSeats")
```

The workspace exported is named "workspace.xml":

```
R> save_workspace(wk, file =  "workspace.xml")
```

6.2. Import a workspace

Height functions can be used to import a workspace:

- `load_workspace()` to load a workspace;

- `compute()` to compute the multi-processings: by default a workspace only contains definitions, computation is needed to get the seasonal adjustment model;
- `get_model()` to get the seasonal adjusted models;
- `get_ts()` to get the input raw times series, `get_object()` and `get_all_objects` to navigate inside the workspace (extract a multi-processing or a seasonal adjustment model), `get_name()` to get the names of the multiprocessings or the seasonal adjustment models and `count()` to count the number of multiprocessing or seasonal adjustment models.

```
R> wk <- load_workspace(file =  "workspace.xml")
R> mp1 <- get_object(wk, 1) # to get the first multiprocessing
R> count(mp1) # there are two seasonal adjustment models in the multiprocessing
```

```
[1] 2
```

```
R> sa_item1 <- get_object(mp1, 1) # to get the first seasonal adjustment model
R> get_name(sa_item1) # the name of the seasonal adjustment model in JDemetra+
```

```
[1] "SA with X13"
```

```
R> raw_ts <- get_ts(sa_item1) # to get the input raw time series
R>
R> compute(wk)
R> sa_model1 <- get_model(sa_item1, workspace = wk)
```

`get_ts()` and `get_model()` can also be used directly to the workspace or a multiprocessing to import all the raw times series or all the seasonal adjustment model:

- for a multiprocessing the result is a list which each element contains the information of a seasonal adjustment model;
- for a workspace the result is a list of length the number of multi-processing and which each element contains a list with the information of each seasonal adjustment model.

```
R> # To get all the raw time series of the workspace:
R> all_raw_ts <- get_ts(wk)
R> # To get all the seasonal adjustment models of the multi-processing "mp1"
R> sa_models_of_mp1 <- get_model(mp1, workspace = wk)
```

Affiliation:

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd