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, calendar effects, time series.

1. Introduction

Since the 20th century, more and more infra-annual statistics are produced, especially by national institutes, to analyse the progression and the outlook of an economy. It is for example the case of the gross domestic product (GDP), unemployment rate, household consumption of goods and industrial production indices. However, most of those time series are affected by seasonal and trading day effects. A seasonal effects is an effect that occur in the same calendar month with similar magnitude and direction from year to year. For instance, automobile production is usually lower during summer, due to holidays, and chocolate sales are usually higher in December, due to Christmas. Trading day effect is the fact that a time series can be affected by each calendar month's weekday composition. For example retail sales are usually higher on Saturday, thus they are likely to be higher in months with a surplus of weekend days.

Therefore, seasonal and trading days effects can make it difficult to analyse the infra-annual movements of a time series or to make spatial comparison. That's why time series are often seasonally and working day adjusted and seasonal adjustment is the process of removing the effects of seasonal and trading day fluctuations.

The most popular seasonal adjustment methods are TRAMO-SEATS+[1] (Gómez and Maravall

[1]The program TRAMO-SEATS+ was developed by Gianluca Caporello and Agustin Maravall — with programming support from Domingo Perez and Roberto Lopez — at the Bank of Spain. It is based on the program TRAMO-SEATS, previously developed by Victor Gomez and Agustin Maravall.

1996; Caporello and Maravall 2004), a parametric method based on ARIMA models, and X-13-ARIMA-SEATS[2] (Findley, Monsell, Bell, Otto, and Chen 1998; Ladiray and Quenneville 2001), a non-parametric method based on moving average. Both methods are recommended by Eurostat and the European Central Bank (ECB) to seasonnaly adjust economic indicators. These methods proceed in two steps summarized in figure 1.

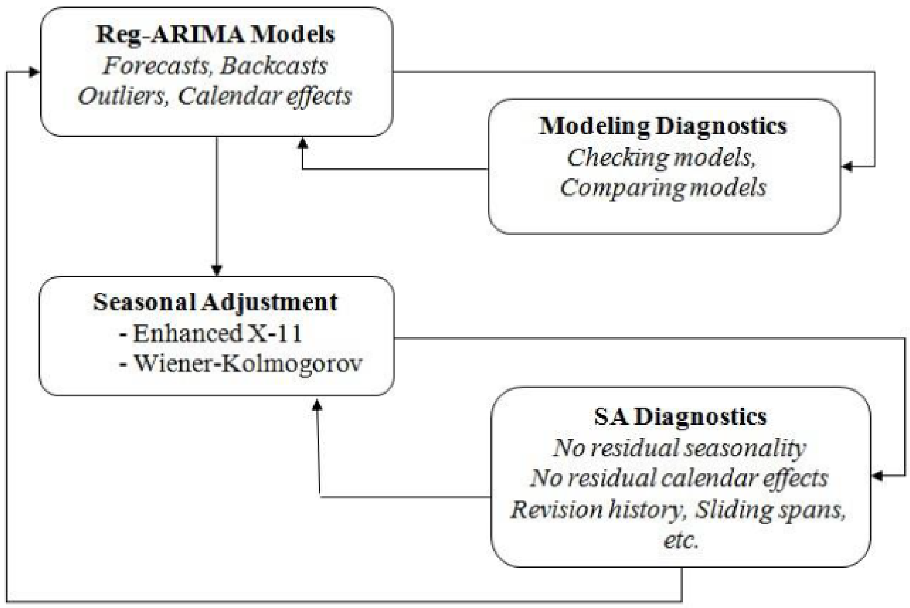


Figure 1: X-13-ARIMA-SEATS and TRAMO-SEATS+ 2-step process: pre-adjustment and decomposition.

The **first step** of seasonal adjustment 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.

[2]The program X-13ARIMA-SEATS is a produced, distributed, and maintained by the US-Census Bureau.

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+ $d_s \leq 1$);

An automatic modelling is also implemented in both methods to: determine the decomposition of the series, detect outliers and calendar effects and to adjust residuals to an ARIMA models. A detailed description can be found in Gómez and Maravall (1998).

In the **second part** of seasonal adjustment, called the **decomposition**, the pre-adjusted series (y) 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 (? à discuter).

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 (Gómez and Maravall 1996; Caporello and Maravall 2004). Whereas in X-12ARIMA/X-13ARIMA, the X-11 algorithm decomposes the time series by means of linear filters (Findley *et al.* 1998; Ladiray and Quenneville 2001).

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

2. JDemetra+ and RJDemetra

JDemetra+ is a new tool for seasonal adjustment (SA) developed by the National Bank of Belgium (NBB) in cooperation with the Deutsche Bundesbank and Eurostat in accordance with the Guidelines of the European Statistical System (ESS) (Eurostat 2015). It implements the concepts and algorithms used in the two leading SA methods: TRAMO-SEATS+ and X-12ARIMA/X-13ARIMA-SEATS. Those methods have been re-engineered using an object-oriented approach that enables easier handling, extensions and modifications.

JDemetra+ has been officially recommended, since 2 February 2015, to the members of the ESS and the European System of Central Banks as software for seasonal and calendar adjustment of official statistics.

Besides seasonal adjustment, JDemetra+ bundles other time series models that are useful in the production or analysis of economic statistics, including for instance outlier detection, nowcasting, temporal disaggregation or benchmarking. More details on the methodology used in JDemetra+ can be found in the JDemetra+ manuals and user guides (Grudkowska 2015a,b).

The package **RJDemetra** provides a R interface to the seasonal adjustment software JDemetra+. **RJDemetra** uses Java libraries of JDemetra+, thus it relies on the **rJava** (Urbanek 2018) package and Java SE 8 or later version is required. It allows to:

- perform seasonal adjustment with TRAMO-SEATS+ and X-12ARIMA/X-13ARIMA-SEATS with pre-defined (section 3) and user-defined specification (section 5);
- acces to all the output available in JDemetra+ (section XXX);
- import and export JDemetra+ workspaces (section 6).

It can be installed from CRAN:

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

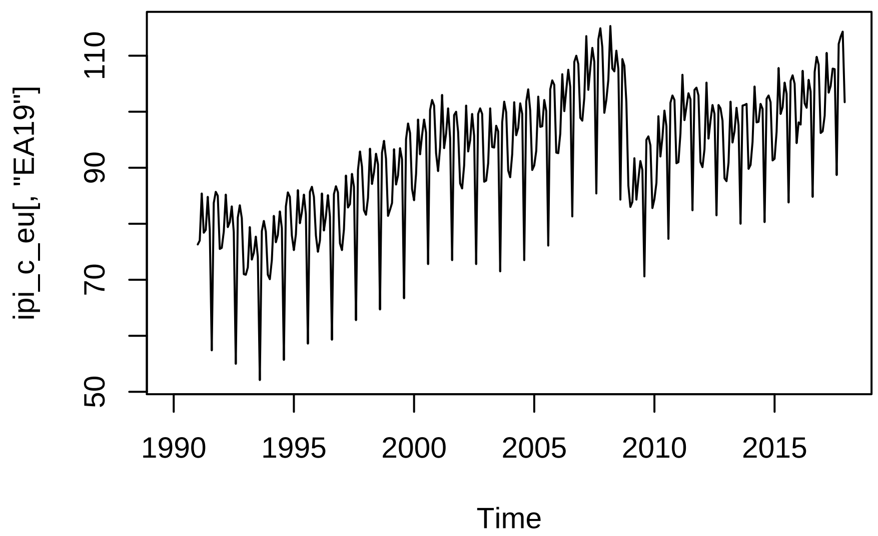
The development version can be installed from GitHub with **devtools** (Wickham, Hester, and Chang 2018):

```
R> devtools::install_github("nbbrd/rjdemetra", args = "--no-multiarch")
```

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"])
```



2.2. Print styling

By default, a color styling is used for the print methods of the objects created by **RJDemetra**. It can causes troubles with some outputs (for example with **rmarkdown** (Xie, Allaire, and Grolemund 2018)) and can be disabled in each print function with the argument `enable_print_style = FALSE` or setting the global option `enable_print_style`:

```
R> options(enable_print_style = FALSE)
```

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 pre-defined specifications correspond to most commonly used specifications and users are recommended to start their analysis with one of them. They 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). With more details, setting described in tables tables 1 and 2 are:

- Transformation test: a test to choose between an additive decomposition (no transformation) and a multiplicative decomposition (logarithmic transformation).

- Pre-adjustment for leap-year: in the case of a multiplicaive decomposition; a correction of the February values is applied to the original series (before transformation). The original values in February are multiplied by $\frac{28.25}{29}$ for leap years, by $\frac{28.25}{28}$ for non-leap years and values for other months are not modified. In the case of multiplicative models, this is equivalent to adding a leap year regressor (Bell 1992).
- Working days: a pre-test is made for a presence of a working day effect.

- Trading days: a pre-test is made for a presence of a trading day effect.

- Easter: a pre-test for a presence of the Easter effect. The default length of the Easter effect is 6 days (for TRAMO-SEATS specifications) and 8 days (for X-13ARIMA-SEATS specifications).

- Outliers: an automatic identification of three types of outliers: AO (additive outliers), LS (level shifts) and TC (transitory changes), using a default critical value.

- ARIMA model: the choice between fixing the ARIMA model structure to (0,1,1)(0,1,1) (Airlin model) or searching for the ARIMA model using an automatic model identification procedure. The Airline model is used as a default model in several TRAMO-SEATS+ and X-13ARIMA-SEATS specifications because it has been shown in many studies that this model is appropriate for many real seasonal monthly or a quarterly time series. Moreover, the Airline model approximates well many other models and provides an excellent "benchmark" model (Maravall 2009).

Four functions can be used in **RJDemetra** to perform an estimation with the pre-defined specification:

- RegARIMA
  - X-13ARIMA method: `regarima_def_x13()`
  - TRAMO-SEATS method: `regarima_def_tramoseats()`
- Seasonal adjustment
  - X-13ARIMA method: `x13_def()`
  - TRAMO-SEATS method: `tramoseats_def()`

For examples:

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		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> 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 = "RSA3", userdefined = NULL)
R> sats <- tramoseats_def(myseries, spec = "RSAfull", userdefined = NULL)
```

In section 5 it is presented how to modify model specifications, including the possibility to incorprate user-defined regressors.

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 estimation 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 sub-section 4.5). 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

The regarima object contains the provided model specification (`specification`; level 2 of the `sa_object`), the estimated coefficients for the arima processes (`arima.coefficients`) and regressors (`regression.coefficients`), including arma orders (`arma`). It comprises also model quality measures (`loglik`), regarima specification after it's estimation with the estimated effects (e.g. linearized input series or outliers)(`model`), the residuals of the regarima model (`residuals`), several tests' results for the residuals (`residuals.stat`) and finally the forecast of the pre-adjusted series (`forecasts`). All this information can be extracted individually by the user or a predefined output can be used by calling `print()` or `summary()` (for more detailed output) functions. Also a set of graphs is defined within the function `plot()`.

For regarima by default the first six graphs are displayed, but specific ones can be chosen within the argument `which`. Table x summarizes all the graphs available for the `sa_object`, as well as its `plot()` options.

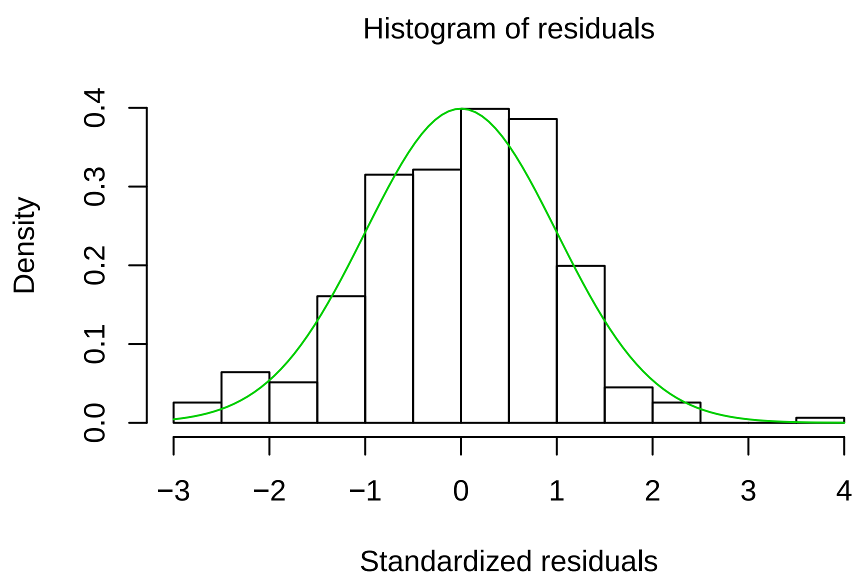
```
R> sax13$regarima

y = regression model + arima (1, 1, 2, 0, 1, 1)
Log-transformation: no
Coefficients:
          Estimate Std. Error
Phi(1)     -0.7603      0.096
Theta(1)   -1.1757      0.095
Theta(2)    0.4551      0.053
BTheta(1)  -0.5433      0.049

             Estimate Std. Error
AO (1-2016)     4.291      0.883
LS (1-2009)    -6.210      0.947
LS (11-2008)   -5.806      0.948
TC (3-2009)    -3.967      0.908


Residual standard error: 1.187 on 311 degrees of freedom
Log likelihood = -496.8, aic =  1012 aicc =  1012, bic(corrected for length) = 0.4898

R> plot(sax13$regarima, which = 2)
```



4.2. Decomposition

As afore-mentioned the decomposition method differs between TRAMO-SEATS+ and X-12ARIMA/X-13ARIMA, where SEATS is based on ARIMA-model and X11-algorithm on linear filters. Therefore also the composition of this object differs when estimating with the two methods (hence the two decomposition objects in the table 3). The only common part is the first two sub-objects with the model specification (`specification`) and information on the decomposition mode (`mode`; e.g.: additive).

Then, the `decomposition_X11` object comprises quality measures on the decomposition (`mstats`), namely the M and Q statistics. It contains also the final unmodified si-ratios $d8$ and final seasonal factors $d10$ (`si_ratio`), as well as the information on the final seasonal filter (`s_filter`) and trend filter(`t_filter`). The code below presents the output for X11 decomposition:

```
R> sax13$decomposition

Monitoring and Quality Assessment Statistics:
      M stats
M(1)    0.028
M(2)    0.033
M(3)    0.338
M(4)    0.376
M(5)    0.366
M(6)    0.067
M(7)    0.066
M(8)    0.157
M(9)    0.073
M(10)   0.145
M(11)   0.120
Q       0.156
Q-M2    0.171

Final filters:
Seasonal filter:  3x5
Trend filter:  13 terms Henderson moving average
```

As a reminder, in SEATS it is assumed that each component of the linearized series (received from TRAMO) is an outcome of a linear stochastic process and SEATS estimates an ARIMA model for each component (i.e. trend, seasonal, transitory and irregular). Subsequently, the `decomposition_SEATS` object contains the information on the estimated ARIMA models (`model`), the linearized components, as obtained from TRAMO (`linearized`), and the theoretical components calculated from the ARIMA models (`components`). The code below presents the output for the SEATS decomposition, with the information on the ARIMA models:

```
R> sats$decomposition

Model
AR :  1 + 0.094056 B - 0.158875 B^2 - 0.294600 B^3
```

```
D :  1 - B - B^12 + B^13
MA :  1 - 0.510600 B^12


SA
AR :  1 + 0.094056 B - 0.158875 B^2 - 0.294600 B^3
D :  1 - 2.000000 B + B^2
MA :  1 - 0.937923 B - 0.015753 B^2 - 0.007005 B^3 + 0.015440 B^4 - 0.001104 B^5
Innovation variance:  0.5777661

Trend
AR :  1 - 0.711394 B
D :  1 - 2.000000 B + B^2
MA :  1 - 0.312018 B - 0.965498 B^2 + 0.346519 B^3
Innovation variance:  0.07090723

Seasonal
D :  1 + B + B^2 + B^3 + B^4 + B^5 + B^6 + B^7 + B^8 + B^9 + B^10 + B^11
MA :  1 + 1.314878 B + 1.722951 B^2 + 2.227262 B^3 + 2.229462 B^4 + 2.119339 B^5 + 1.92645
Innovation variance:  0.08441922

Transitory
AR :  1 + 0.805449 B + 0.414116 B^2
MA :  1 - 0.452913 B - 0.547087 B^2
Innovation variance:  0.02525112

Irregular
Innovation variance:  0.07006354
```

4.3. Final

The final object has a simple structure as it includes the input series, final seasonally adjusted series and the final components (i.e. t - trend-cycle, s - seasonal component and i - irregular component) (`series`), as well as their forecasts (`forecasts`).

```
R> sats$final

Last observed values
             y       sa        t            s           i
Jan 2017  96.5 102.9317 102.6366  -6.431739755  0.29515069
Feb 2017  99.3 102.3815 102.7523  -3.081452963 -0.37082579
Mar 2017 110.5 103.2230 103.0587   7.276971010  0.16431902
Apr 2017 103.4 103.3950 103.4579   0.004977088 -0.06292068
May 2017 104.6 104.1023 103.8400   0.497657700  0.26230153
Jun 2017 107.7 103.8403 104.2863   3.859652578 -0.44596785
Jul 2017 107.6 105.1862 104.9032   2.413785110  0.28301107
```

```
Aug 2017  88.7 105.5484 105.5999 -16.848375561 -0.05151353
Sep 2017 112.1 106.2889 106.3305   5.811052841 -0.04159101
Oct 2017 113.4 106.9003 107.1101   6.499696169 -0.20982123
Nov 2017 114.3 108.3110 107.7487   5.988950554  0.56232415
Dec 2017 101.7 107.7534 108.1104  -6.053350690 -0.35709913

Forecasts:
               y_f     sa_f      t_f        s_f           i_f
Jan 2018 102.49061 108.4010 108.4016  -5.9103802 -0.0005820963
Feb 2018 105.85284 108.8666 108.7446  -3.0137577  0.1219532717
Mar 2018 116.21918 108.9840 109.0820   7.2351710 -0.0979861094
Apr 2018 108.90762 109.4437 109.4153  -0.5360850  0.0284199810
May 2018 110.11320 109.7634 109.7457   0.3498301  0.0176868066
Jun 2018 114.47710 110.0480 110.0740   4.4290998 -0.0260150048
Jul 2018 112.47723 110.4145 110.4009   2.0627149  0.0136293681
Aug 2018  93.98244 110.7265 110.7267 -16.7440681 -0.0002045225
Sep 2018 116.57469 111.0463 111.0518   5.5283721 -0.0054794124
Oct 2018 118.30580 111.3809 111.3764   6.9249450  0.0044980848
Nov 2018 117.56621 111.6992 111.7005   5.8670241 -0.0013538638
Dec 2018 105.59658 112.0237 112.0245  -6.4271099 -0.0007722619
```

4.4. Diagnostics

This part of the `sa_object` includes several diagnostics on the presence of seasonality in the input series and on the quality of the seasonal adjustment.

The test for the seasonality presence (`combined_test`) are performed both on the entire series and in the last 3 years.

Whereas, the quality checking is grouped into two sets. The first looks at the contribution of each estimated component to the variance of the original series (`variance_decomposition`). The second verifies, with different tests, that there is no seasonal pattern left in the seasonally adjusted series and in the irregular component (`residuals_test`).

All the checks (except `combined_seasonality_test`), together with a detailed description, are displayed when printing the diagnostics object.

```
R> # sats$diagnostics
```

4.5. user defined

As presented in the table 3 and in the previous sections the `sa_object` has a defined structure with a defined content. Nevertheless the user can extract additional output from the seasonal adjustment estimation that will be stored under user_defined object in a form of a list. In order extract the additional output the extra variables need to be defined as characters under the argument `userdefined` of the functions `x13_def()`, `tramoseats_def()`, `x13()` or `tramoseats()` (the latter two functions are presented in the next section 5).

For example, to extract the additional tables $d16$ and $c10$ the following need to be specified in the function argument:

```
R> sa_usrdef <- x13_def(myseries, spec = "RSA3",
R+ userdefined = c("decomposition.d16","decomposition.c10"))
R> sa_usrdef$user_defined

Names of additional variables (2):
decomposition.d16, decomposition.c10
```

The list of all available variables can be extracted with the following functions:

- `user_defined_variables("X13-ARIMA")`
- `user_defined_variables("TRAMO-SEATS")`

5. Model specification: creation and modification

See table 4.

5.1. 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;

Table 4: Model specification generation and modification

Estimation	Functions	Specification (spec), class	Functions to generate/modify the spec	Input for function in (4)
(1)	(2)	(3)	(4)	(5)
SA: X-13-ARIMA	x13_def	predefined - see table 2	N/A	N/A
	x13	c(SA_spec, X13)	x13_spec_def x13_spec	predefined spec - see table 2 sa_object of class c(SA, X13) spec of class c(SA_spec, X13)
SA: TRAMO-SEATS	tramoseats_def	predefined - see table 1	N/A	N/A
	tramoseats	c(SA_spec, TRAMO_SEATS)	tramoseats_spec_def	predefined spec - see table 1
			tramoseats	sa_object of class c(SA, TRAMO_SEATS) spec of class c(SA_spec, TRAMO_SEATS)
RegARIMA: X13-ARIMA	regarima_def_x13	predefined - see table 2	N/A	N/A
	regarima	c(regarima_spec, X13)	regarima_spec_def_x13 regarima_spec_x13	predefined - see table 2 regarima object of class c(regarima, X13) spec of class c(regarima_spec, X13)
RegARIMA: TRAMO-SEATS	regarima_def_tramoseats	predefined - see table 1	N/A	N/A
	regarima	c(regarima_spec, TRAMO_SEATS)	regarima_spec_def_tramoseats	predefined - see table 1
			regarima_spec_x13	regarima object of class c(regarima, TRAMO_SEATS) spec of class c(regarima_spec, TRAMO_SEATS))

- `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 time 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.

For instance, to import the workspace created in section 6.1 and to get the first multiprocessing and the first seasonal adjustment model:

```
R> wk <- load_workspace(file =  "workspace.xml")
R> mp1 <- get_object(wk, 1)
R> sa_item1 <- get_object(mp1, 1)
```

To get the number of seasonal adjustment models in the multiprocessing:

```
R> count(mp1)
```

```
[1] 2
```

And the name of the first seasonal adjustment model in JDemetra+:

```
R> get_name(sa_item1)
```

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

Raw time series and seasonal adjustment model can now be imported:

```
R> raw_ts <- get_ts(sa_item1)
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 time 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.

For example to get all raw time series of the workspace and all seasonal adjustmen models of the first multi-processing:

```
R> all_raw_ts <- get_ts(wk)
R> sa_models_of_mp1 <- get_model(mp1, workspace = wk)
```

The imports of seasonal adjustment models from a workspace works well when it has been created throw **RJDemetra**. They may be some troubles when importing a workspace created with JDemetra+, in particular:

- **RJDemetra** doesn't support yet user-defined calendars. A seasonal adjustment model defined with a specific calendar or user-defined calendar regressors will be partially imported. The result will be correct but changing the specification (throw `x13_spec()` or `tramoseats_spec()`) will erase user-defined calendars.

- Seasonal adjustment models with ramp effect or intervention variables will be partially imported: the result of the imported model will be correct but changing the specification (throw `x13_spec()` or `tramoseats_spec()`) will erase them.

- Seasonal adjustment models with no pre-processing (X-11 specification) are not supported: `NULL` object will be returned.

7. Advanced usage and examples

8. Conclusion

Acknowledgments

References

Bell WR (1992). "Alternative Approaches to Length of Month Adjustment." *Statistical Research Division. US Bureau of the Census Statistical Research Division Report Number: Census/SRD/RR*, **92**(01). URL https://www.census.gov/srd/papers/pdf/rr92-17.pd.

Caporello G, Maravall A (2004). "Program TSW: Revised reference manual." *Technical Report, Research Department*, Banco de España. URL https://www.bde.es/f/webbde/SES/Secciones/Publicaciones/PublicacionesSeriadas/DocumentosOcasionales/04/Fic/do0408e.pdf.

Eurostat (2015). "ESS Guidelines on Seasonal Adjustment." *Technical report*, Eurostat Methodologies and Working Papers, European Commission. doi:10.2785/317290. URL http://ec.europa.eu/eurostat/web/products-manuals-and-guidelines/-/KS-GQ-15-001.

Findley DF, Monsell BC, Bell WR, Otto MC, Chen BC (1998). "New Capabilities and Methods of the X-12-ARIMA Seasonal-Adjustment Program." *Journal of Business & Economic Statistics*, **16**(2), 127–152. ISSN 07350015. URL http://www.jstor.org/stable/1392565.

Gómez V, Maravall A (1996). "Programs TRAMO and SEATS. Instructions for the User." *Working papers 9628*, Banco de España.

Gómez V, Maravall A (1998). "Automatic Modeling Methods for Univariate Series." *Working papers 9608*, Banco de España. URL https://EconPapers.repec.org/RePEc:bde:wpaper:9808.

Grudkowska S (2015a). "JDemetra+ reference manual." *Department of Statistics, Narodowy Bank Polski, Warsaw.* URL https://ec.europa.eu/eurostat/cros/system/files/jdemetra_reference_manual_version_2.1_0.pdf.

Grudkowska S (2015b). "JDemetra+ user guide." *Department of Statistics, Narodowy Bank Polski, Warsaw.* URL https://ec.europa.eu/eurostat/cros/system/files/jdemetra_user_guide_version_2.2.pdf.

Ladiray D, Quenneville B (2001). *Seasonal adjustment with the X-11 Method*. Springer-Verlag Inc, Berlin; New York.

Maravall A (2009). "Identification of Reg-ARIMA Models and of Problematic Series in Large Scale Applications Program TSW (TRAMO-SEATS for Windows)." URL https://www.bde.es/f/webbde/SES/servicio/Programas_estadisticos_y_econometricos/Notas_introductorias_TRAMO_SEATS/ficheros/Large_Scale_TSW.pdf.

Urbanek S (2018). ***rJava**: Low-Level R to Java Interface*. R package version 0.9-10, URL https://CRAN.R-project.org/package=rJava.

Wickham H, Hester J, Chang W (2018). ***devtools**: Tools to Make Developing R Packages Easier*. R package version 2.0.1, URL https://CRAN.R-project.org/package=devtools.

Xie Y, Allaire J, Grolemund G (2018). *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida. ISBN 9781138359338, URL https://bookdown.org/yihui/rmarkdown.

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