JDemetra+ online documentation

Anna Smyk, Alain Quartier-la-Tente, Tanguy Barthelemy, Karsten Webel

12/23/2022

Table of contents

Preface

Welcome to the JDemetra+ online documentation.

JDemetra+ is a software for seasonal adjustment and other time series functions, developed in the framework of Eurostat's "Centre of Excellence on Statistical Methods and Tools" by the National Bank of Belgium with the support of the Bundesbank.

To learn more about this project https://ec.europa.eu/eurostat/cros/content/centre-excellence-statistical-methods-and-tools.

JDemetra+

Introduction

JD+ is a library of algorithms for seasonal adjustment and time series econometrics. You can learn more about the history of the project here (link to below)

link to key references - handbooks (3) - sets of guidelines (2 or 3 ?)

Version 2.2.3 and version 3

approach

as v2 still widely used …

Main functions

Seasonal adjustment algorithms

All are available for low and high frequency data.

	Algorithms	Access	Key features
X13-Arima			
Tramo-Seats			
STL			
State Space Models (STS)			

Algorithms	Access	Key features

Temporal Disaggregation and benchmarking

Algorithms	Access	Key features
Chow-lin		
Fernandez		

Trend-cycle estimation

Algorithms	Access	Key features

Nowcasting

Algorithms	Access	Key features

Structure of this book

This book is divided in four parts, allowing the user to access the resources from different perspectives.

Algorithms

This part provides a step by step description of all the algorithms featured in JD+, grouped by purpose - seasonal adjustment - benchmarking - temporal disaggregation - … links

Tools

JDemetra+ offers 3 kinds of tools - A Graphical User Interface (GUI) which can be enhanced with plug-ins - A set of R packages - A Cruncher for mass production in seasonal adjustment

Underlying Statistical Methods

This part gives details about the underlying statistical methods to foster a more in-depth understanding of the algorithms.Those methods are described in the light and spirit of their use as building blocks of the algorithms presented above, not aiming at all at their comprehensive coverage.

How to use this book

Audience

This book targets the beginner as well as seasoned methodologist interested in using JDemetra+ software for any the purposes listed below.

The documentation is built in layers allowing to skip details and complexity in the first steps

Part I

Algorithms

Seasonal Adjustment

Motivation

The primary aim of the seasonal adjustment process is to remove seasonal fluctuations from the time series.

[insert def SA from b_ov]

Data frequencies

The seasonal adjustment methods available in JDemetra+ aim to decompose a time series into components and remove seasonal fluctuations from the observed time series. The X-11 method considers monthly and quarterly series while SEATS is able to decompose series with 2, 3, 4, 6 and 12 observations per year.

Unobserved Components (UC)

The main components, each representing the impact of certain types of phenomena on the time series (X_t), are:

- The trend (T_t) that captures long-term and medium-term behaviour;
- The seasonal component (S_t) representing intra-year fluctuations, monthly or quarterly, that are repeated more or less regularly year after year;
- The irregular component (I_t) combining all the other more or less erratic fluctuations not covered by the previous components.

In general, the trend consists of 2 sub-components:

- The long-term evolution of the series;
- The cycle, that represents the smooth, almost periodic movement around the long-term evolution of the series. It reveals a succession of phases of growth and recession.

To achieve this goal, seasonal adjustment methods decompose the original time series into components that capture specific movements. These components are: trend-cycle, seasonality and irregularity. The trend-cycle component includes long-term and medium-term movements in the data. For seasonal adjustment purposes there is no need to divide this component into two parts. JDemetra+ refers to the trend-cycle as trend and consequently this convention is used here. For seasonal adjustment purposes both TRAMO-SEATS and X-13ARIMA-SEATS do not separate the long-term trend from the cycle as these two components are usually too short to perform their reliable estimation. Consequently, hereafter TRAMO-SEATS and X-13ARIMA-SEATS estimate the trend component. However, the original TRAMO-SEATS may separate the long-term trend from the cycle through the Hodrick-Precsott filter using the output of the standard decomposition. It should be remembered that JDemetra+ refers to the trend-cycle as trend (T_t), and consequently this convention is used in this document.

TRAMO-SEATS considers two decomposition models:

- The additive model: $X_t = T_t + S_t + I_t$;
- The log additive model: $log(X_t) = log(T_t) + log(S_t) + log(I_t)$.

Apart from these two decomposition types X-13ARIMA-SEATS allows the user to apply also the multiplicative model: $X_t = T_t \times S_t \times I_t$.

A time series x_t, which is a subject to a decomposition, is assumed to be a realisation of a discrete-time stochastic, covariance-stationary linear process, which is a collection of random variables x_t, where t denotes time. It can be shown that any stochastic, covariance-stationary process can be presented in the form:

$x_t = \mu_t + \tilde{x}_t$,

1

where μ_t is a linearly deterministic component and $\tilde{x}_t$ is a linearly interderministic component, such as:

$$\tilde{x}_t = \sum_{j=0}^{\infty} \psi_j a_{t-j}$$

,

2

where $\sum_{j=0}^{\infty} \psi_i^2 < \infty$ (coefficients ψ_j are absolutely summable), $\psi_0 = 1$ and a_t is the white noise error with zero mean and constant variance V_a. The error term a_t represents the one-period ahead forecast error of x_t, that is:

$$a_t = \tilde{x}_t - \hat{x}_{t|t-1}$$

,

3

where

$$\hat{x}_{t|t-1}$$

is the forecast of

$$\tilde{x}_t$$

made at period $t-1$. As a_t represents what is new in

$$\tilde{x}_t$$

in point t, i.e., not contained in the past values of

$$\tilde{x}_t$$

, it is also called innovation of the process. From

3

$$\tilde{x}_t$$

can be viewed as a linear filter applied to the innovations.

The equation 7.1 is called a Wold representation. It presents a process as a sum of linearly deterministic component μ_t and linearly interderministic component $\sum_{j=0}^{\infty}\psi_j a_{t-j}$, the first one is perfectly predictable once the history of the process x_{t-1} is known and the second one is impossible to predict perfectly. This explains why the stochastic process cannot be perfectly predicted.

Under suitable conditions

$$\tilde{x}_t$$

can be presented as a weighted sum of its past values and a_t, i.e.:

$$\tilde{x}_t = \sum_{j=0}^{\infty} \pi_j \tilde{x}_{t-j} + a$$

t

,

4

In general, for the observed time series, the assumptions concerning the nature of the process

1

do not hold for various reasons. Firstly, most observed time series display a mean that cannot be assumed to be constant due to the presence of a trend and the seasonal movements.

Secondly, the variance of the time series may vary in time. Finally, the observed time series usually contain outliers, calendar effects and regression effects, which are treated as deterministic. Therefore, in practice a prior transformation and an adjustment need to be applied to the time series. The constant variance is usually achieved through taking a logarithmic transformation and the correction for the deterministic effects, while stationarity of the mean is achieved by applying regular and seasonal differencing. These processes, jointly referred to as preadjustment or linearization, can be performed with the TRAMO or RegARIMA models. Besides the linearisation, forecasts and backcasts of stochastic time series are estimated with the ARIMA model, allowing for later application of linear filters at both ends of time series. The estimation performed with these models delivers the stochastic part of the time series, called the linearised series, which is assumed to be an output of a linear stochastic process.[1] The deterministic effects are removed from the time series and used to form the final components.

In the next step the linearised series is decomposed into its components. There is a fundamental difference in how this process is performed in TRAMO-SEATS and X-13ARIMA-SEATS. In TRAMO-SEATS the decomposition is performed by the SEATS procedure, which follows a so called ARIMA model based approach. In principle, it aims to derive the components with statistical models. More information is given in the SEATS section. X-13ARIMA-SEATS offers two algorithms for decomposition: SEATS and X-11. The X-11 algorithm, which is described in the X-11 section section, decomposes a series by means of linear filters. Finally, in both methods the final components are derived by the assignment of the deterministic effects to the stochastic components. Consequently, the role of the ARIMA models is different in each method. TRAMO-SEATS applies the ARIMA models both in the preadjustment step and in the decomposition procedure. On the contrary, when the X-11 algorithm is used for decomposition, X-13ARIMA-SEATS uses the ARIMA model only in the preadjustment step. In summary, the decomposition procedure that results in an estimation of the seasonal component requires prior identification of the deterministic effects and their removal from the time series. This is achieved through the linearisation process performed by the TRAMO and the RegARIMA models, shortly discussed in the Linearisation with the TRAMO and RegARIMA models section.The linearised series is then decomposed into the stochastic components with SEATS or X-11 algorithms.

X-13

X-13ARIMA is a seasonal adjustment program developed and supported by the U.S. Census Bureau. It is based on the U.S. Census Bureau's earlier X-11 program, the X-11-ARIMA program developed at Statistics Canada, the X-12-ARIMA program developed by the U.S. Census Bureau.

[1]When the series are non-stationary differentiation is performed before the seasonality tests.

Detecting seasonal patterns

Pre-treatment

Calendar correction

details of regressor building in calendar chapter

rationale

method

tools

Outliers

rationale

method

tools

Reg-Arima Model

Tramo and Reg-Arima are very similar...details in M chapter

Model evaluation

goodness-of-fit

X-11 Decompostion

this part should allow to use x-11 via RJDemetra as well as via GUI

Quick launch with default specifications

Output 1: series

X-11 gives access to a great part of it's intermediate computations

Here we focus on the final components (Table D)

List of series (edit : table with name and meaning)

Retrieve in GUI

Retrieve in R

Output 2: final parameters

Relevant if parameters not set manually

List Final trend filter final seasonal filer

Retrieve via GUI image

Retrieve in R

Output 3: diagnostics

X11 produces the following type diagnostics or quality measures Table with link to detail

Specifications / parameters

List

0.0.0.0.1 * General settings

- **Mode**
  - check if this option still works, if so add and edit instructions from old page)
  - if not but button present : explain that the mode is determined in pre-adjustment (function)
- **Seasonal component**
  - check if still relevant, idem as above
  - in v.2.3 if not ticked, S estimated but options on seasonal filter not available
- **Forecasts horizon**

Length of the forecasts generated by the RegARIMA model - in months (positive values) - years (negative values) - if set to is set to 0, the X-11 procedure does not use any model-based forecasts but the original X-11 type forecasts for one year. - default value: -1, thus one year from the Arima model

- **Backcasts horizon**

Length of the backcasts generated by the RegARIMA model - in months (positive values) - years (negative values) - default value: 0

0.0.0.0.2 * Irregular correction

- **LSigma**
  - sets lower sigma (standard deviation) limit used to down-weight the extreme irregular values in the internal seasonal adjustment iterations, learn more here (LINK to M_ chapter)
  - values in $[0, Usigma]$
  - default value is 1.5
- **USigma**
  - sets upper sigma (standard deviation)
  - values in $[Lsigma, +\infty]$
  - default value is 2.5
- **Calendarsigma**
  - allows to set different **LSigma** and **USigma** for each period
  - values
    - None (default)
    - All: standard errors used for the extreme values detection and adjustment computed separately for each calendar month/quarter
    - Signif: groups determined by cochran test (check)
    - Sigmavec: set two customized groups of periods
- **Excludeforecasts**
  - ticked : forecasts and backcasts from the RegARIMA model not used in Irregular Correction
  - unticked (default): forecasts and backcasts used

0.0.0.0.3 * Seasonality extraction filters

- **Seasonal filter** choice

Specifies which be used to estimate the seasonal factors for the entire series (link to relevant part in M chapter)

- S3 $\times$ 1 – 3 $\times$ 1 moving average.
- S3 $\times$ 3 – 3 $\times$ 3 moving average.
- S3 $\times$ 5 – 3 $\times$ 5 moving average.
- S3 $\times$ 9 – 3 $\times$ 9 moving average.
- S3 $\times$ 15 – 3 $\times$ 15 moving average.
- Stable – a single seasonal factor for each calendar period is generated by calculating a simple average over all values for each period (taken after detrending and outlier correction).
- X11Default – 3 $\times$ 3 moving average is used to calculate the initial seasonal factors and a 3 $\times$ 5 moving average to calculate the final seasonal factors.
- Msr – automatic choice of a seasonal filter. The seasonal filters can be selected for the entire series, or for a particular month or quarter.
- default value: Msr

Check: will user choice be applied to all steps or only to final phase D step

- **Details on seasonal filters**

Sets different seasonal filters by period in order to account for seasonal heteroskedasticity (link to M chapter)

- default value: empty

0.0.0.0.4 * Trend estimation filters

- **Automatic Henderson filter** our user-defined
  - default: length 13
  - unticked: user defined length choice
- **Henderson filter** length choice
  - values: odd number in $[3, 101]$
  - default value: 13

Check: will user choice be applied to all steps or only to final phase D step

Parameter setting in GUI

here v2, adjust to v3 asap

Parameter setting in R packages

extensive help on functions available in package help pages Rcode snippets

In R, to implement any param change, it is required to retrieve current spec, modify it and apply it again (see T R packages chapter for details). (specific link)

here example changing all the settings (just remove irrelevant changes)

Rjdemetra (v2) Edit : here static R code link to a "worked example" wih dynamic code ticked box in GUI corresponds to …? in R

```
#Creating a modified specification, customizing all available X11 parameters
modified_spec<- x13_spec(current_sa_model,
    #x11.mode="?",
    #x11.seasonalComp = "?",
    x11.fcasts = -2,
    x11.bcasts = -1,
    x11.lsigma = 1.2,
    x11.usigma = 2.8,
    x11.calendarSigma = NA, # EDIT with example
      x11.sigmaVector = NA,
    x11.excludeFcasts = NA
    # filters
    x11.trendAuto = NA, # needed inf value ?
    x11.trendma = 23,
    x11.seasonalma = "S3X9
    # details on seasonal filters)

#New SA estimation : apply modified_spec
modified_sa_model<-x13(raw_series,modified_spec)
```

EDIT : link to package help page v2 +v3

STL

SEATS

SSF

Quality assessment

Residual seasonality

Residual calendar effects

Seasonal adjustment of high frequency data

Motivation

Ubiquitous use

Data specificities

Tools

code here and/or link to R packages chapter

Unobserved Components

Identifying seasonal patterns

Spectral analysis

Seasonality tests

Pre-adjustment

Calendar correction

Outliers and intervention variables

Linearization

Decomposition

Extended X-11

STL decomposition

Arima Model Based (AMB) Decomposition

State Space framework

Quality assessment

Residual seasonality

Residual Calendar effects

Outlier detection

(in or outside a seasonal adjustment process)

Motivation

With Reg Arima models

Specific TERROR tool

With structural models (BSM)

Calendar and user-defined corrections

This chapter describes the generating process of calendar regressors, outliers and other input variables. The use of this variables inside a seasonal adjustment process is described in the relevant chapter on SA or on SA of HF data.

Overview of Calendar effects in JDemetra

edit : this has a evolved a lot with v3 definition possibilities via GUI and R have to be re described

The following description of the calendar effects in JDemetra+ is strictly based on PALATE, J. (2014).

A natural way for modelling calendar effects consists of distributing the days of each period into different groups. The regression variable corresponding to a type of day (a group) is simply defined by the number of days it contains for each period. Usual classifications are:

- Trading days (7 groups): each day of the week defines a group (Mondays,...,Sundays);
- Working days (2 groups): week days and weekends.

The definition of a group could involve partial days. For instance, we could consider that one half of Saturdays belong to week days and the second half to weekends.

Usually, specific holidays are handled as Sundays and they are included in the group corresponding to "non-working days". This approach assumes that the economic activity on national holidays is the same (or very close to) the level of activity that is typical for Sundays. Alternatively, specific holidays can be considered separately, e.g. by the specification that divided days into three groups:

- Working days (Mondays to Fridays, except for specific holidays),
- Non-working days (Saturdays and Sundays, except for specific holidays),
- Specific holidays.

Summary of the method used in JDemetra+ to compute trading day and working day effects

The computation of trading day and working days effects is performed in four steps:

1. Computation of the number of each weekday performed for all periods.
2. Calculation of the usual contrast variables for trading day and working day.
3. Correction of the contrast variables with specific holidays (for each holiday add +1 to the number of Sundays and subtract 1 from the number of days of the holiday). The correction is not performed if the holiday falls on a Sunday, taking into account the validity period of the holiday.
4. Correction of the constant variables for long term mean effects, > taking into account the validity period of the holiday; see below > for the different cases.

The corrections of the constant variables may receive a weight corresponding to the part of the holiday considered as a Sunday.

An example below illustrates the application of the above algorithm for the hypothetical country in which three holidays are celebrated:

- New Year (a fixed holiday, celebrated on 01 January);
- Shrove Tuesday (a moving holiday, which falls 47 days before Easter Sunday, celebrated until the end of 2012);
- Freedom day (a fixed holiday, celebrated on 25 April).

The consecutive steps in calculation of the calendar for 2012 and 2013 years are explained below.

First, the number of each day of the week in the given month is calculated as it is shown in table below.

Number of each weekday in different months

Month	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
Jan-12	5	5	4	4	4	4	5
Feb-12	4	4	5	4	4	4	4
Mar-12	4	4	4	5	5	5	4
Apr-12	5	4	4	4	4	4	5
May-12	4	5	5	5	4	4	4
Jun-12	4	4	4	4	5	5	4
Jul-12	5	5	4	4	4	4	5
Aug-12	4	4	5	5	5	4	4

Month	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Sep-12	4	4	4	4	4	5	5
Oct-12	5	5	5	4	4	4	4
Nov-12	4	4	4	5	5	4	4
Dec-12	5	4	4	4	4	5	5
Jan-13	4	5	5	5	4	4	4
Feb-13	4	4	4	4	4	4	4
Mar-13	4	4	4	4	5	5	5
Apr-13	5	5	4	4	4	4	4
May-13	4	4	5	5	5	4	4
Jun-13	4	4	4	4	4	5	5
Jul-13	5	5	5	4	4	4	4
Aug-13	4	4	4	5	5	5	4
Sep-13	5	4	4	4	4	4	5
Oct-13	4	5	5	5	4	4	4
Nov-13	4	4	4	4	5	5	4
Dec-13	5	5	4	4	4	4	5

Next, the contrast variables are calculated (table below) as a result of the linear transformation applied to the variables presented in table below.

Contrast variables (series corrected for leap year effects)

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Jan-12	0	0	-1	-1	-1	-1	0
Feb-12	0	0	1	0	0	0	0.75
Mar-12	0	0	0	1	1	1	0
Apr-12	0	-1	-1	-1	-1	-1	0
May-12	0	1	1	1	0	0	0
Jun-12	0	0	0	0	1	1	0
Jul-12	0	0	-1	-1	-1	-1	0
Aug-12	0	0	1	1	1	0	0
Sep-12	-1	-1	-1	-1	-1	0	0
Oct-12	1	1	1	0	0	0	0
Nov-12	0	0	0	1	1	0	0
Dec-12	0	-1	-1	-1	-1	0	0
Jan-13	0	1	1	1	0	0	0
Feb-13	0	0	0	0	0	0	-0.25
Mar-13	-1	-1	-1	-1	0	0	0
Apr-13	1	1	0	0	0	0	0
May-13	0	0	1	1	1	0	0

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Jun-13	-1	-1	-1	-1	-1	0	0
Jul-13	1	1	1	0	0	0	0
Aug-13	0	0	0	1	1	1	0
Sep-13	0	-1	-1	-1	-1	-1	0
Oct-13	0	1	1	1	0	0	0
Nov-13	0	0	0	0	1	1	0
Dec-13	5	5	4	4	4	4	0

In the next step the corrections for holidays is done in the following way:

- New Year: In 2012 it falls on a Sunday. Therefore no correction is applied. In 2013 it falls on a Tuesday. Consequently, the following corrections are applied to the number of each weekday in January: Tuesday -1, Sunday +1, so the following corrections are applied to the contrast variables: -2 for Tuesday and -1 for the other contrast variables.
- Shrove Tuesday: It is a fixed day of the week holiday that always falls on Tuesday. For this reason in 2012 the following corrections are applied to the number of each weekday in February: Tuesday -1, Sunday +1, so the following corrections are applied to the contrast variables: -2 for the contrast variable associated with Tuesday, and -1 for the other contrast variables. The holiday expires at the end of 2012. Therefore no corrections are made for 2013.
- Freedom Day: In 2012 it falls on a Wednesday. Consequently, the following corrections are applied to the number of each weekday in April: Wednesday -1, Sunday +1, so the following corrections are applied to the contrast variables: -2 for Wednesday and -1 for the other contrast variables. In 2013 it falls on Thursday. Therefore, the following corrections are applied to the number of each weekday in April: Thursday -1, Sunday +1, so the following corrections are applied to the contrast variables: -2 for Thursday, and -1 for the other contrast variables.

The result of these corrections is presented in table below.

Contrast variables corrected for holidays

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Jan-12	0	0	-1	-1	-1	-1	0
Feb-12	-1	-2	0	-1	-1	-1	0.75
Mar-12	0	0	0	1	1	1	0
Apr-12	-1	-2	-3	-2	-2	-2	0
May-12	0	1	1	1	0	0	0
Jun-12	0	0	0	0	1	1	0
Jul-12	0	0	-1	-1	-1	-1	0

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Aug-12	0	0	1	1	1	0	0
Sep-12	-1	-1	-1	-1	-1	0	0
Oct-12	1	1	1	0	0	0	0
Nov-12	0	0	0	1	1	0	0
Dec-12	0	-1	-1	-1	-1	0	0
Jan-13	-1	-1	0	0	-1	-1	0
Feb-13	0	0	0	0	0	0	-0.25
Mar-13	-1	-1	-1	-1	0	0	0
Apr-13	0	0	-1	-2	-1	-1	0
May-13	0	0	1	1	1	0	0
Jun-13	-1	-1	-1	-1	-1	0	0
Jul-13	1	1	1	0	0	0	0
Aug-13	0	0	0	1	1	1	0
Sep-13	0	-1	-1	-1	-1	-1	0
Oct-13	0	1	1	1	0	0	0
Nov-13	0	0	0	0	1	1	0
Dec-13	0	0	-1	-1	-1	-1	0

Finally, the long term corrections are applied on each year of the validity period of the holiday.

- New Year: Correction on the contrasts: +1, to be applied to January of 2012 and 2013.
- Shrove Tuesday: It may fall either in February or in March. It will fall in March if Easter is on or after 17 April. Taking into account the theoretical distribution of Easter, it gives: prob(March) = +0.22147, prob(February) = +0.77853. The correction of the contrasts will be +1.55707 for Tuesday in February 2012 and +0.77853 for the other contrast variables. The correction of the contrasts will be +0.44293 for Tuesday in March 2012, +0.22147 for the other contrast variables.
- Freedom Day: Correction on the contrasts: +1, to be applied to April of 2012 and 2013.

The modifications due to the corrections described above are presented in table below.

Trading day variables corrected for the long term effects

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Jan-12	1	1	0	0	0	0	0
Feb-12	-0.22115	-0.44229	0.778853	-0.22115	-0.22115	-0.22115	0.75
Mar-12	0.221147	0.442293	0.221147	1.221147	1.221147	1.221147	0
Apr-12	0	-1	-2	-1	-1	-1	0
May-12	0	1	1	1	0	0	0

Month	Mon	Tue	Wed	Thu	Fri	Sat	Length
Jun-12	0	0	0	0	1	1	0
Jul-12	0	0	-1	-1	-1	-1	0
Aug-12	0	0	1	1	1	0	0
Sep-12	-1	-1	-1	-1	-1	0	0
Oct-12	1	1	1	0	0	0	0
Nov-12	0	0	0	1	1	0	0
Dec-12	0	-1	-1	-1	-1	0	0
Jan-13	0	0	1	1	0	0	0
Feb-13	0	0	0	0	0	0	-0.25
Mar-13	-1	-1	-1	-1	0	0	0
Apr-13	1	1	0	-1	0	0	0
May-13	0	0	1	1	1	0	0
Jun-13	-1	-1	-1	-1	-1	0	0
Jul-13	1	1	1	0	0	0	0
Aug-13	0	0	0	1	1	1	0
Sep-13	0	-1	-1	-1	-1	-1	0
Oct-13	0	1	1	1	0	0	0
Nov-13	0	0	0	0	1	1	0
Dec-13	0	0	-1	-1	-1	-1	0

Mean and seasonal effects of calendar variables

The calendar effects produced by the regression variables that fulfil the definition presented above include a mean effect (i.e. an effect that is independent of the period) and a seasonal effect (i.e. an effect that is dependent of the period and on average it is equal to 0). Such an outcome is inappropriate, as in the usual decomposition of a series the mean effect should be allocated to the trend component and the fixed seasonal effect should be affected to the corresponding component. Therefore, the actual calendar effect should only contain effects that don't belong to the other components.

In the context of JDemetra+ the mean effect and the seasonal effect are long term theoretical effects rather than the effects computed on the time span of the considered series (which should be continuously revised).

The mean effect of a calendar variable is the average number of days in its group. Taking into account that one year has on average 365.25 days, the monthly mean effects for a working days are, as shown in the table below, 21.7411 for week days and 8.696 for weekends.

Monthly mean effects for the Working day variable

Groups of Working day effect	Mean effect
Week days	365.25/12*5/7 = **21.7411**
Weekends	365.25/12*2/7 = **8.696**
Total	365.25/12 = **30.4375**

The number of days by period is highly seasonal, as apart from February, the length of each month is the same every year. For this reason, any set of calendar variables will contain, at least in some variables, a significant seasonal effect, which is defined as the average number of days by period (Januaries..., first quarters...) outside the mean effect. Removing that fixed seasonal effects consists of removing for each period the long term average of days that belong to it. The calculation of a seasonal effect for the working days classification is presented in the table below.

The mean effect and the seasonal effect for the calendar periods

Period	Average number of days	Average number of week days	Mean effect	Seasonal effect
January	31	31*5/7=22.1429	21.7411	0.4018
February	28.25	28.25*5/7=20.1786	21.7411	-1.5625
March	31	31*5/7=22.1429	21.7411	0.4018
April	30	30*5/7=21.4286	21.7411	-0.3125
May	31	31*5/7=22.1429	21.7411	0.4018
June	30	30*5/7=21.4286	21.7411	-0.3125
July	31	31*5/7=22.1429	21.7411	0.4018
August	31	31*5/7=22.1429	21.7411	0.4018
September	30	30*5/7=21.4286	21.7411	-0.3125
October	31	31*5/7=22.1429	21.7411	0.4018
November	30	30*5/7=21.4286	21.7411	-0.3125
December	31	31*5/7=22.1429	21.7411	0.4018
Total	365.25	260.8929	260.8929	0

For a given time span, the actual calendar effect for week days can be easily calculated as the difference between the number of week days in a specific period and the sum of the mean effect and the seasonal effect assigned to this period, as it is shown in the table below for the period 01.2013 – 06.2013.

The calendar effect for the period 01.2013 - 06.2013

Time period (t)	Week days	Mean effect	Seasonal effect	Calendar effect
Jan-2013	23	21.7411	0.4018	0.8571

Time period (t)	Week days	Mean effect	Seasonal effect	Calendar effect
Feb-2013	20	21.7411	-1.5625	-0.1786
Mar-2013	21	21.7411	0.4018	-1.1429
Apr-2013	22	21.7411	-0.3125	0.5714
May-2013	23	21.7411	0.4018	0.8571
Jun-2013	20	21.7411	-0.3125	-1.4286
Jul-2013	23	21.7411	0.4018	0.8571

The distinction between the mean effect and the seasonal effect is usually unnecessary. Those effects can be considered together (simply called mean effects) and be computed by removing from each calendar variable its average number of days by period. These global means effect are considered in the next section.

Impact of the mean effects on the decomposition

When the ARIMA model contains a seasonal difference – something that should always happen with calendar variables – the mean effects contained in the calendar variables are automatically eliminated, so that they don't modify the estimation. The model is indeed estimated on the series/regression variables after differencing. However, they lead to a different linearised series (y_{lin}). The impact of other corrections (mean and/or fixed seasonal) on the decomposition is presented in the next paragraph. Such corrections could be obtained, for instance, by applying other solutions for the long term corrections or by computing them on the time span of the series.

Now the model with "correct" calendar effects (denoted as C), i.e. effects without mean and fixed seasonal effects, can be considered. To simplify the problem, the model has no other regression effects.

For such a model the following relations hold:

$$y_{\text{lin}} = y - C$$

$$T = F_T\left(y_{\text{lin}}\right)$$

$$S = F_S\left(y_{\text{lin}}\right) + C$$

$$I = F_I\left(y_{\text{lin}}\right)$$

where:

T - the trend;

S - the seasonal component;

I - the irregular component;

F_X - the linear filter for the component X.

Consider next other calendar effects ($\widetilde{C}$) that contain some mean (cm, integrated to the final trend) and fixed seasonal effects (cs, integrated to the final seasonal). The modified equations are now:

$$\widetilde{C} = C + cm + cs$$

$$\tilde{y}_{\text{lin}} = \ y - \widetilde{C} = \ y_{\text{lin}} - cm - cs$$

$$\widetilde{T} = \ F_T\left(\tilde{y}_{\text{lin}}\right) + cm$$

$$\widetilde{S} = \ F_S\left(\tilde{y}_{\text{lin}}\right) + C + cs$$

$$\tilde{I} = \ F_I\left(\tilde{y}_{\text{lin}}\right)$$

Taking into account that F_X is a linear transformation and that[2]

$$F_T\left(\text{cm}\right) = cm$$

$$F_T\left(\text{cs}\right) = 0$$

$$F_S\left(\text{cm}\right) = 0$$

[2]In case of SEATS the properties can be trivially derived from the matrix formulation of signal extraction. They are also valid for X-11 (additive).

$$F_S\left(\text{cs}\right) = cs$$

$$F_I\left(\text{cm}\right) = 0$$

$$F_I\left(\text{cs}\right) = 0$$

The following relationships hold:

$$\widetilde{T} = \ F_T\left(\tilde{y}_{\text{lin}}\right) + cm = F_T\left(y_{\text{lin}}\right) - cm + cm = T$$

$$\widetilde{S} = \ F_S\left(\tilde{y}_{\text{lin}}\right) + C + cs = F_S\left(y_{\text{lin}}\right) - cs + C + cs = S$$

$$\tilde{I} = \ I$$

If we don't take into account the effects and apply the same approach as in the "correct" calendar effects, we will get:

$$\breve{T} = \ F_T\left(\tilde{y}_{\text{lin}}\right) = T - cm$$

$$\breve{S} = \ F_S\left(\tilde{y}_{\text{lin}}\right) + \widetilde{C} = S + cm$$

$$\breve{I} = \ F_I\left(\tilde{y}_{\text{lin}}\right) = I$$

The trend, seasonal and seasonally adjusted series will only differ by a (usually small) constant.

In summary, the decomposition does not depend on the mean and fixed seasonal effects used for the calendar effects, provided that those effects are integrated in the corresponding final components. If these corrections are not taken into account, the main series of the decomposition will only differ by a constant.

Linear transformations of the calendar variables

As far as the RegARIMA and the TRAMO models are considered, any non-degenerated linear transformation of the calendar variables can be used. It will produce the same results (likelihood, residuals, parameters, joint effect of the calendar variables, joint F-test on the coefficients of the calendar variables...). The linearised series that will be further decomposed is invariant to any linear transformation of the calendar variables.

However, it should be mentioned that choices of calendar corrections based on the tests on the individual t statistics are dependent on the transformation, which is rather arbitrary. This is the case in old versions of TRAMO-SEATS. That is why the joint F-test (as in the version of TRAMO-SEATS implemented in TSW+) should be preferred.

An example of a linear transformation is the calculation of the contrast variables. In the case of the usual trading day variables, they are defined by the following transformation: the 6 contrast variables $(\text{No.}\,(\text{Mondays}) - No.\,(\text{Sundays})\,, \ldots No.\,(\text{Saturdays}) - No.(Sundays))$ used with the length of period.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \text{Mon} \\ \text{Tue} \\ \text{Wed} \\ \text{Thu} \\ \text{Fri} \\ \text{Sat} \\ \text{Sun} \end{bmatrix} = \begin{bmatrix} Mon - Sun \\ Tue - Sun \\ Wed - Sun \\ Thu - Sun \\ Fri - Sun \\ Sat - Sun \\ \text{Length of period} \end{bmatrix}$$

For the usual working day variables, two variables are used: one contrast variable and the length of period

$$\begin{bmatrix} 1 & -\frac{5}{2} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \text{Week} \\ \text{Weekend} \end{bmatrix} = \begin{bmatrix} \text{Contrast week} \\ \text{Length of period} \end{bmatrix}$$

The Length of period variable is defined as a deviation from the length of the month (in days) and the average month length, which is equal to 30.4375. Instead, the leap-year variable can be used here (see Regression sections in RegARIMA or Tramo)[3].

Such transformations have several advantages. They suppress from the contrast variables the mean and the seasonal effects, which are concentrated in the last variable. So, they lead to fewer correlated variables, which are more appropriate to be included in the regression model. The sum of the effects of each day of the week estimated with the trading (working) day contrast variables cancel out.

[3] GÓMEZ, V., and MARAVALL, A (2001b).

Handling of specific holidays

check vs GUI (v3) and rjd3 modelling

Three types of holidays are implemented in JDemetra+:

- Fixed days, corresponding to the fixed dates in the year (e.g. New Year, Christmas).
- Easter related days, corresponding to the days that are defined in relation to Easter (e.g. Easter +/- n days; example: Ascension, Pentecost).
- Fixed week days, corresponding to the fixed days in a given week of a given month (e.g. Labor Day celebrated in the USA on the first Monday of September).

From a conceptual point of view, specific holidays are handled in exactly the same way as the other days. It should be decided, however, to which group of days they belong. Usually they are handled as Sundays. This convention is also used in JDemetra+. Therefore, except if the holiday falls on a Sunday, the appearance of a holiday leads to correction in two groups, i.e. in the group that contains the weekday, in which holiday falls, and the group that contains the Sundays.

Country specific holidays have an impact on the mean and the seasonal effects of calendar effects. Therefore, the appropriate corrections to the number of particular days (which are usually the basis for the definition of other calendar variables) should be applied, following the kind of holidays. These corrections are applied to the period(s) that may contain the holiday. The long term corrections in JDemetra+ don't take into account the fact that some moving holidays could fall on the same day (for instance the May Day and the Ascension). However, those events are exceptional, and their impact on the final result is usually not significant.

Fixed day

The probability that the holiday falls on a given day of the week is 1/7. Therefore, the probability to have 1 day more that is treated like Sunday is 6/7. The effect on the means for the period that contains the fixed day is presented in the table below (the correction on the calendar effect has the opposite sign).

The effect of the fixed holiday on the period, in which it occurred

Sundays	Others days	Contrast variables
+ 6/7	- 1/7	1/7 - (+ 6/7)= -1

Easter related days

Easter related days always fall the same week day (denoted as Y in the table below: The effects of the Easter Sunday on the seasonal means). However, they can fall during different periods (months or quarters). Suppose that, taking into account the distribution of the dates for Easter and the fact that this holiday falls in one of two periods, the probability that Easter falls during the period m is p, which implies that the probability that it falls in the period $m+1$ is $1-p$. The effects of Easter on the seasonal means are presented in the table below.

The effects of the Easter Sunday on the seasonal means

Period	Sundays	Days X	Others days	Contrast Y	Other contrasts
m	+ p	- p	0	- 2p	- p
m+1	+ (1-p)	- (1-p)	0	- 2×(1-p)	- (1-p)

The distribution of the dates for Easter may be approximated in different ways. One of the solutions consists of using some well-known algorithms for computing Easter on a very long period. JDemetra+ provides the Meeus/Jones/Butcher's and the Ron Mallen's algorithms (they are identical till year 4100, but they slightly differ after that date). Another approach consists in deriving a raw theoretical distribution based on the definition of Easter. It is the solution used for Easter related days. It is shortly explained below.

The date of Easter in the given year is the first Sunday after the full moon (the Paschal Full Moon) following the northern hemisphere's vernal equinox. The definition is influenced by the Christian tradition, according to which the equinox is reckoned to be on 21 March[4] and the full moon is not necessarily the astronomically correct date. However, when the full moon falls on Sunday, then Easter is delayed by one week. With this definition, the date of Easter Sunday varies between 22 March and 25 April. Taking into account that an average lunar month is 29.530595 days the approximated distribution of Easter can be derived. These calculations do not take into account the actual ecclesiastical moon calendar.

For example, the probability that Easter Sunday falls on 25 March is 0.004838 and results from the facts that the probability that 25 March falls on a Sunday is 1/7 and the probability that the full moon is on 21 March, 22 March, 23 March or 24 March is 5/29.53059. The probability that Easter falls on 24 April is 0.01708 and results from the fact that the probability that 24 April is Sunday is 1/7 and takes into account that 18 April is the last acceptable date for the full moon. Therefore the probability that the full moon is on 16 April or 17 April is 1/29.53059 and the probability that the full moon is on 18 April is 1.53059/29.53059.

The approximated distribution of Easter dates

	Day	Probability
22 March		1/7 * 1/29.53059
23 March		1/7 * 2/29.53059

[4] In fact, astronomical observations show that the equinox occurs on 20 March in most years.

	Day	Probability
24 March	1/7 * 3/29.53059
25 March	1/7 * 4/29.53059
26 March	1/7 * 5/29.53059
27 March	1/7 * 6/29.53059
28 March	1/29.53059
29 March	1/29.53059
...	...
18 April	1/29.53059
19 April	1/7 * (6 + 1.53059)/29.53059
20 April	1/7 * (5 + 1.53059)/29.53059
21 April	1/7 * (4 + 1.53059)/29.53059
22 April	1/7 * (3 + 1.53059)/29.53059
23 April	1/7 * (2 + 1.53059)/29.53059
24 April	1/7 * (1 + 1.53059)/29.53059
25 April	1/7 * 1.53059/29.53059

Fixed week days

Fixed week days always fall on the same week day (denoted as Y in the table below) and in the same period. Their effect on the seasonal means is presented in the table below.

The effect of the fixed week holiday on the period, in which it occurred

Sundays	Day Y	Others days
+ 1	- 1	0

The impact of fixed week days on the regression variables is zero because the effect itself is compensated by the correction for the mean effect.

Holidays with a validity period

When a holiday is valid only for a given time span, JDemetra+ applies the long term mean corrections only on the corresponding period. However, those corrections are computed in the same way as in the general case.

It is important to note that using or not using mean corrections will impact in the estimation of the RegARIMA and TRAMO models. Indeed, the mean corrections do not disappear after differencing. The differences between the SA series computed with or without mean corrections will no longer be constant.

Different Kinds of calendars

see link with GUI

This scenario presents how to define different kinds of calendars. These calendars can be applied to the specifications that take into account country-specific holidays and can be used for detecting and estimating the calendar effects.

The calendar effects are those parts of the movements in the time series that are caused by different number of weekdays in calendar months (or quarters). They arise as the number of occurrences of each day of the week in a month (or a quarter) differs from year to year. These differences cause regular effects in some series. In particular, such variation is caused by a leap year effect because of an extra day inserted into February every four years. As with seasonal effects, it is desirable to estimate and remove calendar effects from the time series.

The calendar effects can be divided into a mean effect, a seasonal part and a structural part. The mean effect is independent from the period and therefore should be allocated to the trend-cycle. The seasonal part arises from the properties of the calendar that recur each year. For one thing, the number of working days of months with 31 calendar days is on average larger than that of months with 30 calendar days. This effect is part of the seasonal pattern captured by the seasonal component (with the exception of leap year effects). The structural part of the calendar effect remains to be determined by the calendar adjustment. For example, the number of working days of the same month in different years varies from year to year.

Both X-12-ARIMA/X-13ARIMA-SEATS and TRAMO/SEATS estimate calendar effects by adding some regressors to the equation estimated in the pre-processing part (RegARIMA or TRAMO, respectively). Regressors mentioned above are generated from the default calendar or the user defined calendar.

The calendars of JDemetra+ simply correspond to the usual trading days contrast variables based on the Gregorian calendar, modified to take into account some specific holidays. Those holidays are handled as "Sundays" and the variables are properly adjusted to take into account the long term mean effects.

Tests for residual trading days

We consider below tests on the seasonally adjusted series (sa_t) or on the irregular component (irr_t). When the reasoning applies on both components, we will use y_t. The functions *stdev* stands for "standard deviation" and *rms* for "root mean squares"

The tests are computed on the log-transformed components in the case of multiplicative decomposition.

TD are the usual contrasts of trading days, 6 variables (no specific calendar).

Non significant irregular

When irr_t is not significant, we don't compute the test on it, to avoid irrelevant results. We consider that irr_t is significant if $stdev(irr_t) > 0.01$ (multiplicative case) or if $stdev(irr_t)/rms(sa_t) > 0.01$ (additive case).

F test

The test is the usual joint F-test on the TD coefficients, computed on the following models:

0.0.0.0.1 * Autoregressive model (AR modelling option)

We compute by OLS:

$$y_t = \mu + \alpha y_{t-1} + \beta TD_t + \epsilon_t$$

0.0.0.0.2 * Difference model

We compute by OLS:

$$\Delta y_t - \overline{\Delta y_t} = \beta TD_t + \epsilon_t$$

So, the latter model is a restriction of the first one ($\alpha = 1, \mu = \mu = \overline{\Delta y_t}$)

The tests are the usual joint F-tests on $\beta \quad (H_0 : \beta = 0)$.

By default, we compute the tests on the 8 last years of the components, so that they might highlight moving calendar effects.

Remark:

In Tramo, a similar test is computed on the residuals of the Arima model. More exactly, the F-test is computed on $e_t = \beta TD_t + \epsilon_t$, where e_t are the one-step-ahead forecast errors.

Algorithms for benchmarking and temporal disaggregation

In this chapter we describe the practical implementation, the underlying theory in a dedicated chapter.(link)

Benchmarking overview

Often one has two (or multiple) datasets of different frequency for the same target variable. Sometimes, however, these data sets are not coherent in the sense that they don't match up. Benchmarking[^1] is a method to deal with this situation. An aggregate of a higher-frequency measurement variables is not necessarily equal to the corresponding lower-frequency less-aggregated measurement. Moreover, the sources of data may have different reliability levels. Usually, less frequent data are considered more trustworthy as they are based on larger samples and compiled more precisely. The more reliable measurements, hence often the less frequent, will serve as benchmark.

In seasonal adjustment methods benchmarking is the procedure that ensures the consistency over the year between adjusted and non-seasonally adjusted data. It should be noted that the [ESS Guidelines on Seasonal Adjustment (2015)] (https://ec.europa.eu/eurostat/documents/3859598/6830795/K GQ-15-001-EN-N.pdf/d8f1e5f5-251b-4a69-93e3-079031b74bd3), do not recommend benchmarking as it introduces a bias in the seasonally adjusted data. The U.S. Census Bureau also points out that "*forcing the seasonal adjustment totals to be the same as the original series annual totals can degrade the quality of the seasonal adjustment, especially when the seasonal pattern is undergoing change. It is not natural if trading day adjustment is performed because the aggregate trading day effect over a year is variable and moderately different from zero*"[^2]. Nevertheless, some users may need that the annual totals of the seasonally adjusted series match the annual totals of the original, non-seasonally adjusted series[^3].

According to the [ESS Guidelines on Seasonal Adjustment (2015)] (https://ec.europa.eu/eurostat/documents/38 GQ-15-001-EN-N.pdf/d8f1e5f5-251b-4a69-93e3-079031b74bd3), the only benefit of this approach is that there is consistency over the year between adjusted and the non-seasonally adjusted data; this can be of particular interest when low-frequency (e.g. annual) benchmarking figures officially exist (e.g. National Accounts, Balance of Payments, External Trade, etc.) and where users' needs for time consistency are stronger.

Tools

Benchmarking with GUI

1. With the pre-defined specifcations the benchmarking functionality is not applied by default following the *ESS Guidelines on Seasonal Adjustment* (2015) recommendations. It means that once the user has seasonally adjustd the series with a pre-defined specifcation the *Benchmarking* node is empty. To execute benchmarking click on the *Specifications* button and activate the checkbox in the *Benchmarking* section.

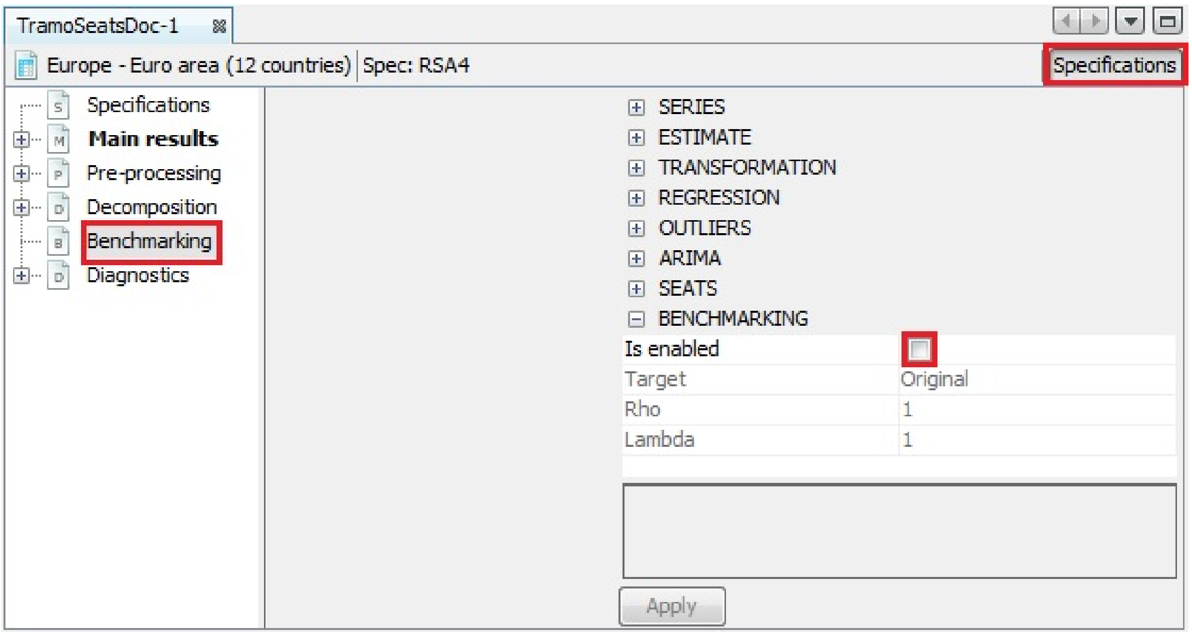

Figure 1: Text

Benchmarking option – a default view

2. Three parameters can be set here. *Target* specifies the target variable for the benchmarking procedure. It can be either the *Original* (the raw time series) or the *Calendar Adjusted* (the time series adjusted for calendar effects). *Rho* is a value of the AR(1) parameter (set between 0 and 1). By default it is set to 1. Finally, *Lambda* is a parameter that relates to the weights in the regression equation. It is typically equal to 0 (for an additive decomposition), 0.5 (for a proportional decomposition) or 1 (for a multiplicative decomposition). The default value is 1.

3. To launch the benchmarking procedure click on the **Apply** button. The results are displayed in four panels. The top-left one compares the original output from the seasonal adjustment procedure with the result from applying a benchmarking to the seasonal

adjustment. The bottom-left panel highlights the differences between these two results. The outcomes are also presented in a table in the top-right panel. The relevant statistics concerning relative differences are presented in the bottom-right panel.

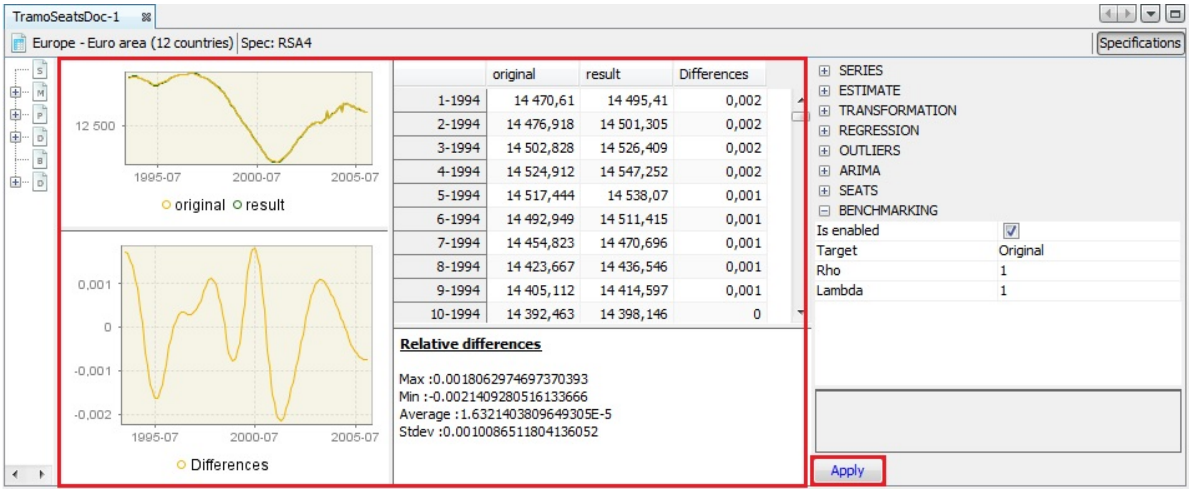

Figure 2: Text

The results of the benchmarking procedure

4. Both pictures and the table can be copied the usual way (see the *Simple seasonal adjustment of a single time series* scenario).

Options for benchmarking results

5. To export the result of the benchmarking procedure (*benchmarking.result*) and the target data (*benchmarking.target*) one needs to once execute the seasonal adjustment with benchmarking using the muli-processing option (see the *Simple seasonal adjustment of multiple time series* scenario. Once the muli-processing is executed, select the *Output* item from the *SAProcessing* menu.

The *SAProcessing* menu

6. Expand the "+" menu and choose an appropriate data format (here Excel has been chosen). It is possible to save the results in TXT, XLS, CSV, and CSV matrix formats. Note that the available content of the output depends on the output type.

Exporting data to an Excel file

7. Chose the output items that refer to the results from the benchmarking procedure, move them to the window on the right and click **OK**.

Exporting the results of the benchmarking procedure

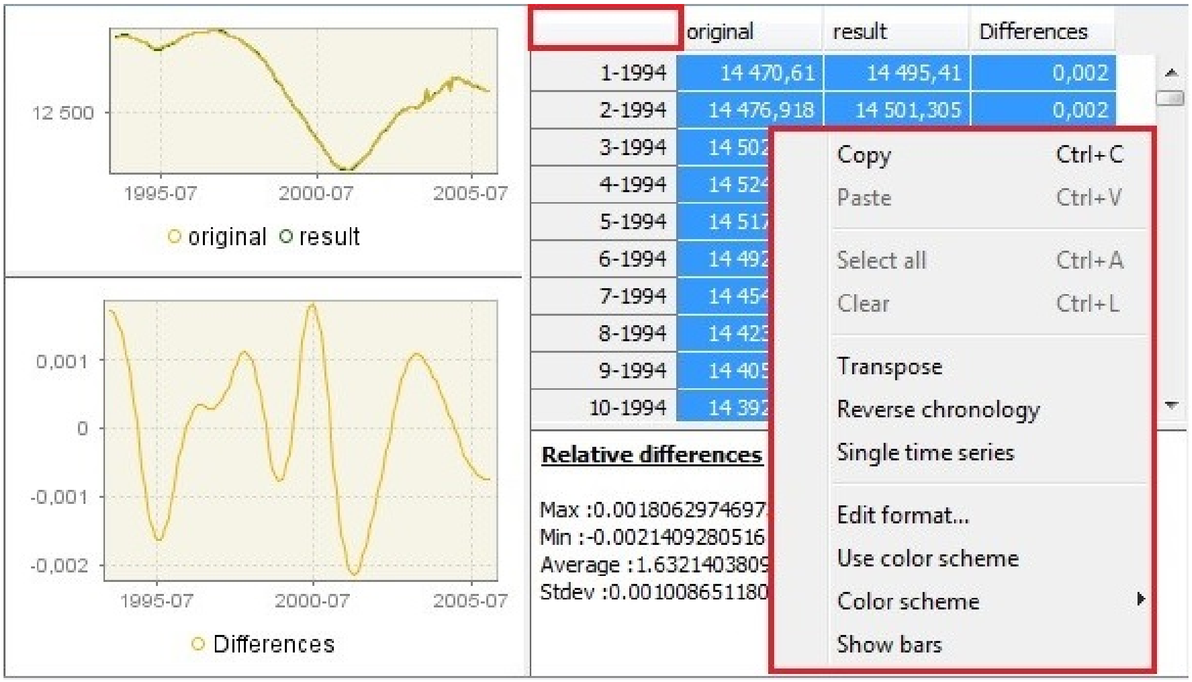

Figure 3: Text

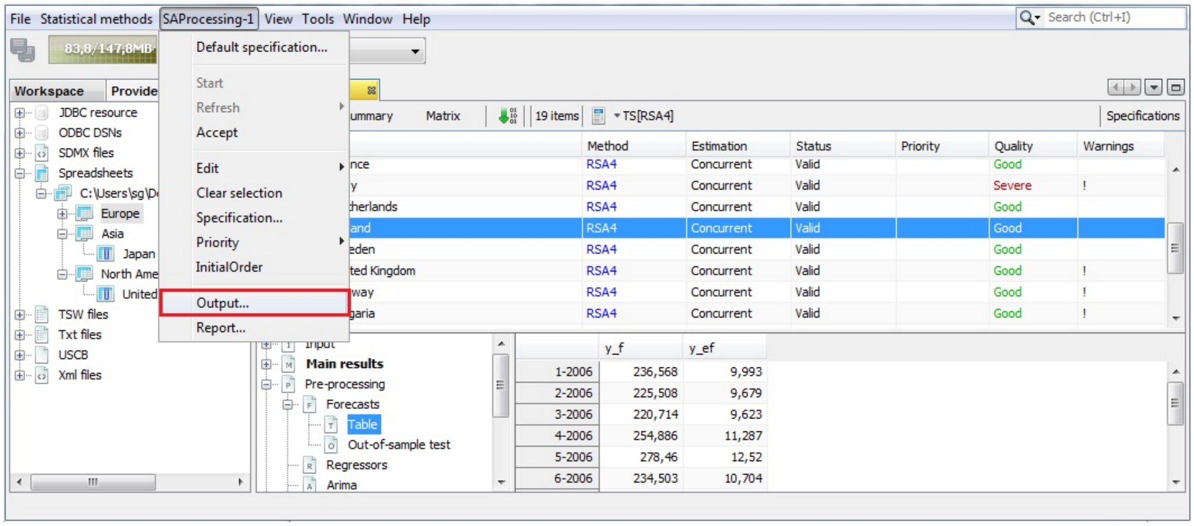

Figure 4: Text

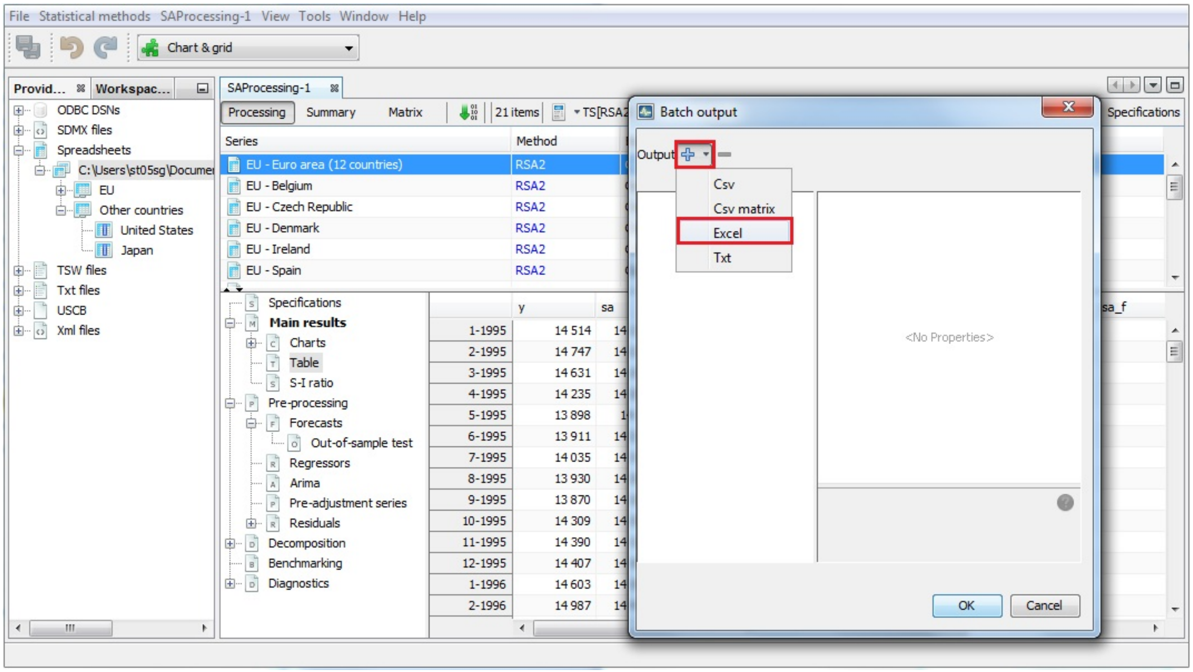

Figure 5: Text

Benchmarking in R

package rjd3bench orga doc - here - in package - example

References

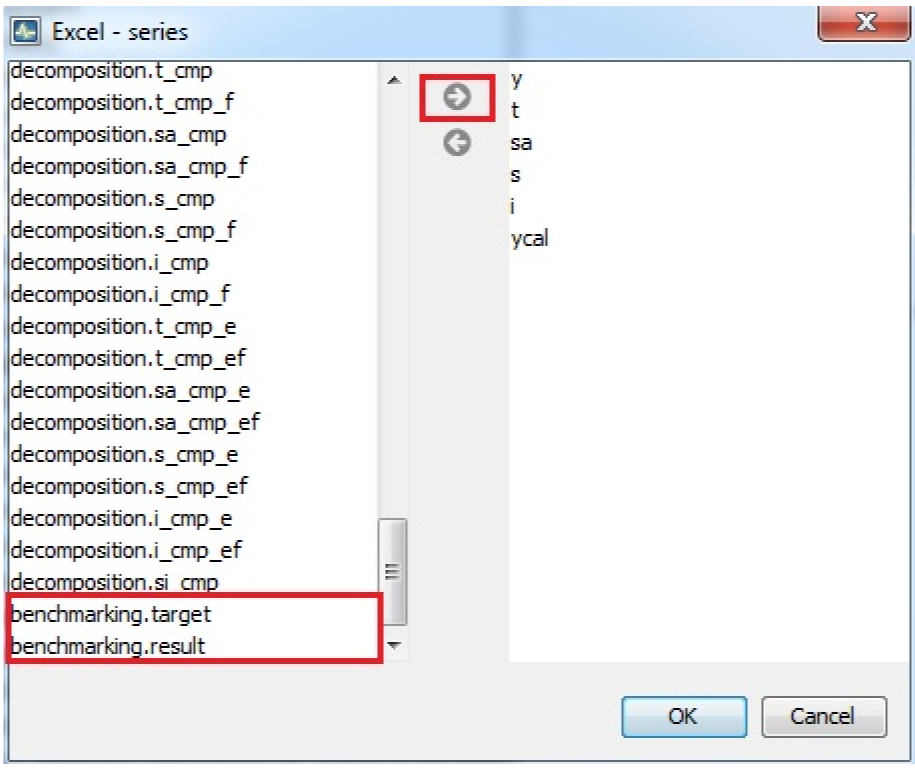

Figure 6: Text

Trend-cycle estimation

Motivation

Estimation Methods

Tools

rjd3 highfeq package

rjdfilters package

Nowcasting

Motivation

Underlying Theory: references ?

Tools

- plug in ?
- R package ?

Part II

Tools

Graphical User Interface

Overview

R packages

Available algorithms

Table 15: Main Algorithms accessible via R packages

Domain	Algorithm	Package	Comments
Seasonal Adjustment	X13-Arima		
	Tramo-Seats		
		rjd3highfreq	
	STL()	rjd3stl	
	Basic Structural Models	rjd3sts	
Trend estimation	Moving Averages, Local Polynomial	rjdfilters	
	idem	rjd3highfreq	
Benchmarking and TD	rjd3bench		

Organisation overview

RJDemetra and JD+ version 2.2.3

Note on RJDemetra

examples in A_sa : v2 and v3 rest of the chapter on version 3

(ecriture standard des noms des packages..cf K) a suite (order)

dependencies

utility packages

general output organisation

Installation procedure

```
#install.packages("remotes")
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")
```

Below you will find a comprehensive list and main functions by categories

for detailed function, you can refer to each package's own R documentation

Utility packages

rjd3toolkit

Contains several utility functions used in other `rjd` packages and several functions to perform test.

Utility

Tests - Normality tests: Bowman-Shenton (`bowmanshenton()`), Doornik-Hansen (`doornikhansen()`), Jarque-Bera (`jarquebera()`)

- Runs tests (randomness of data): mean or the median (`testofruns()`) or up and down runs test (`testofupdownruns()`)
- autocorrelation functions (usual, inverse, partial)
- `aggregate()` to aggregate a time series to a higher frequency

Example

```
library(rjd3toolkit)
set.seed(100)
x = rnorm(1000);y = rlnorm(1000)
bowmanshenton(x) # normal distribution
bowmanshenton(y) # log-normal distribution
testofruns(x) # random data
testofruns(y) # random data
testofruns(1:1000) # non-random data
autocorrelations(x)
autocorrelations.inverse(x)
autocorrelations.partial(x)
```

rjd3modelling

Purpose : creating input variables (regressors) for to be used in Reg-Arima

- create user-defined calendar and trading-days regressors: `calendar.new()` (create a new calendar), `calendar.holiday()` (add a specific holiday, e.g. christmas), `calendar.easter()` (easter related day) and `calendar.fixedday()`)
- outliers regressors (AO, LS, TC, SO, Ramp, intervention variables), calendar related regressors (stock, leap year, periodic dummies and contrasts, trigonometric variables) -> to be added quadratic ramps
- Range-mean regression test (to choose log transformation), Canova-Hansen (`td.ch()`) and trading-days f-test (`td.f()`)
- specification functions for `rjd3x13` and `rjd3tramoseats`

more explanations and example needed here

Example of calendar specification

```
library(rjd3modelling)
fr_cal <- calendar.new()
calendar.holiday(fr_cal, "NEWYEAR")
calendar.holiday(fr_cal, "EASTERMONDAY")
calendar.holiday(fr_cal, "MAYDAY")
calendar.fixedday(fr_cal, month = 5, day = 8,
                  start = "1953-03-20")
```

```
# calendar.holiday(fr_cal, "WHITMONDAY") # Equivalent to:
calendar.easter(fr_cal, offset = 61)

calendar.fixedday(fr_cal, month = 7, day = 14)
# calendar.holiday(fr_cal, "ASSUMPTION")
calendar.easter(fr_cal, offset = 61)
calendar.holiday(fr_cal, "ALLSAINTSDAY")
calendar.holiday(fr_cal, "ARMISTICE")
calendar.holiday(fr_cal, "CHRISTMAS")
```

Use `holidays()` to get the days of the holidays and `htd()` to get the trading days regressors

```
holidays(fr_cal, "2020-12-24", 10,single = T)
s = ts(0, start = 2020, end = c(2020, 11), frequency = 12)
# Trading-days regressors (each day has a different effect, sunday as contrasts)
td_reg <- htd(fr_cal, s = s, groups = c(1, 2, 3, 4, 5, 6, 0))
# Working-days regressors (Monday = ... = Friday; Saturday = Sunday = contrasts)
wd_reg <- htd(fr_cal, s = s, groups = c(1, 1, 1, 1, 1, 0, 0))
# Monday = ... = Friday; Saturday; Sunday = contrasts
wd_reg <- htd(fr_cal, s = s, groups = c(1, 1, 1, 1, 1, 2, 0))
wd_reg
```

more explanations on contrasts and references links to calendar correction chapters

Example of outliers

```
s = ts(0, start = 2000, end = 2005, frequency = 12)
ao = ao.variable(s = s, date = "2001-03-01")
ls = ls.variable(s = s, date = "2001-01-01")
tc = tc.variable(s = s, date = "2001-01-01", rate = 0.7)
so = so.variable(s = s, date = "2003-05-01")
ramp = ramp.variable(s = s, range = c("2001-01-01","2001-12-01"))
plot(ts.union(ao, ls, tc, so, ramp), plot.type = "single",
     col = c("red","lightgreen","orange","blue","black"))
```

rjd3sa package

Seasonality tests:

(for each code snippet example and link to M_Test chapter)

- Canova-Hansen (`seasonality.canovahansen()`)
- X-12 combined test (`seasonality.combined()`)
- F-test on seasonal dummies (`seasonality.f()`)
- Friedman Seasonality Test (`seasonality.friedman()`)
- Kruskall-Wallis Seasonality Test (`seasonality.kruskalwallis()`)
- Periodogram Seasonality Test (`seasonality.periodogram()`)
- QS Seasonality Test (`seasonality.qs()`)

EXP : Always correct the trend and remove the mean before seasonality tests:

```
library(rjd3sa)
y = diff(rjd3toolkit::ABS$X0.2.09.10.M, 1); y = y - mean(y)
seasonality.f(y, 12)
seasonality.friedman(y, 12)
seasonality.kruskalwallis(y, 12)
seasonality.combined(y, 12)
```