ESTP: Selected Webinars for Young Statisticians

JDemetra+ basics: Seasonal adjustment with X13-Arima

Anna Smyk

Online, April 16th 2026

Table of Contents I

Section 1

Opening

Your trainer

Time series methodologist at Insee (Paris, France)

- Consulting and methodological support for the analysis of time series at Insee
- Conception and refactoring of production processes of seasonally adjusted data
- Training: development and delivery of internal and external training courses on JD+ and seasonal adjustment

Coordinator of Eurostat's Center of excellence on Time Series Analysis (COSA project)

In particular:

- In charge of improving JDemetra+ documentation
  https://jdemetra-new-documentation.netlify.app/

Course agenda (1/2)

- What is JDemetra+ ?
- What is Seasonal Adjustment (SA) ? Motivation and main concepts
- Using JDemetra+ for SA: getting familiar with the Graphical user Interface (GUI): a worked example (follow along)
- X13-Arima algorithm: decomposition, linearization, calendar correction
- Applying X13-Arima with JDemetra+ GUI: a worked example (follow along)

Course agenda (2/2)

- Applying X13-Arima with {rjd3x13} R package: a worked example (follow along)
- Generate output from the GUI and using the Cruncher
- Refresh policies: a quick overview
- Production of seasonally adjusted data: summary and review
- Q&A

Section 2

JDemetra+ overview

JDemetra+: a library of algorithms for time series analysis

JDemetra+ a library of algorithms (written in Java) on:

- Seasonal Adjustment (Historical domain)
- Trend and cycle estimation
- Benchmarking and temporal disaggregation
- Revision Analysis
- Nowcasting

They can be accessed via Graphical user-interface (GUI) and/or R packages.

JDemetra+ is an open source software, officially recommended by Eurostat since 2015 for Seasonal Adjustment to ESS members.

JDemetra+ on Github

- Repository dedicated to Java algorithms and Graphical User interface (+ extensions) : https://github.com/jdemetra
- Documentation https://jdemetra-new-documentation.netlify.app/
- Repository dedicated to R packages: https://github.com/rjdverse

For each R package README files with basic examples

Seasonal Adjustment Algorithms in JDemetra+

Algorithm	Version 2.x		Version 3.x	
	Access in GUI	Access in R	Access in GUI	Access in R
X-13 Arima	yes	RJDemetra	yes	`rjd3x13`
Tramo-Seats	yes	RJDemetra	yes	`rjd3tramoseats`
X12plus			yes	`rjd3x11plus`
STL			yes	`rjd3stl`
BSM			yes	`rjd3sts`
SEATS+			upcoming	upcoming

Today focus on **X13-Arima** with GUI and in R.

(Admissible periodicities for low frequency data: p in $2, 3, 4, 6, 12$ in all algorithms, main and additional)

Section 3

What is Seasonal Adjustment (SA)? Motivation and main concepts

Concept of seasonality

Definition of seasonality: **infra-annual variations**, fairly stable, with a **12-months periodicity** for an monthly series, and of **4-months periodicity** for a quarterly series. (Januaries are alike, Februaries are alike, ..., and Decembers are alike)

An ambiguous concept: "fairly stable": no exact criterion to determine when fluctuations can't be considered "stable enough"

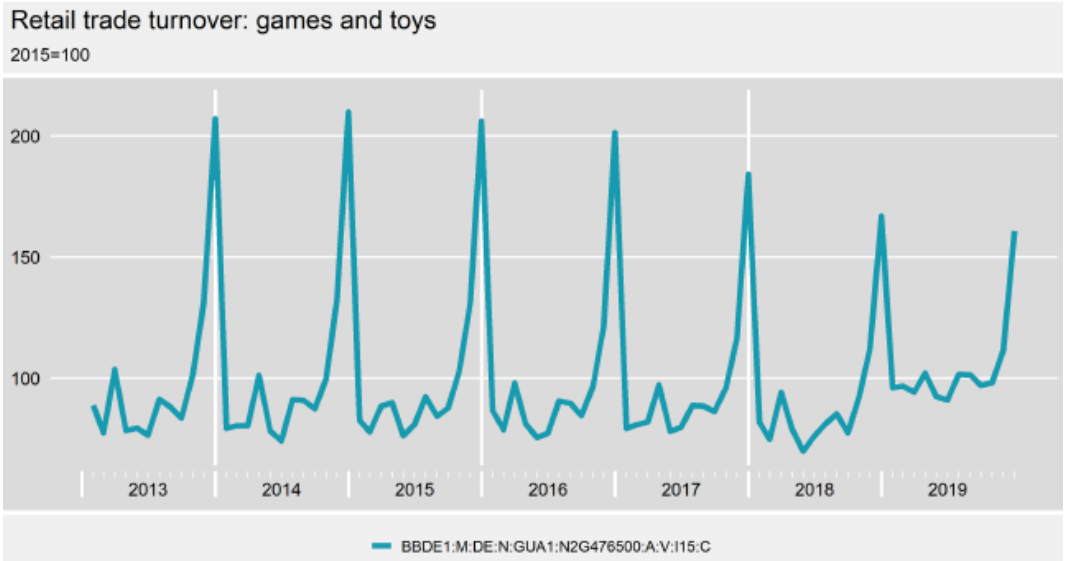
Causes of seasonality: climate, traditions, institutions

Seasonality evolves over time: technical progress, institutional and cultural changes

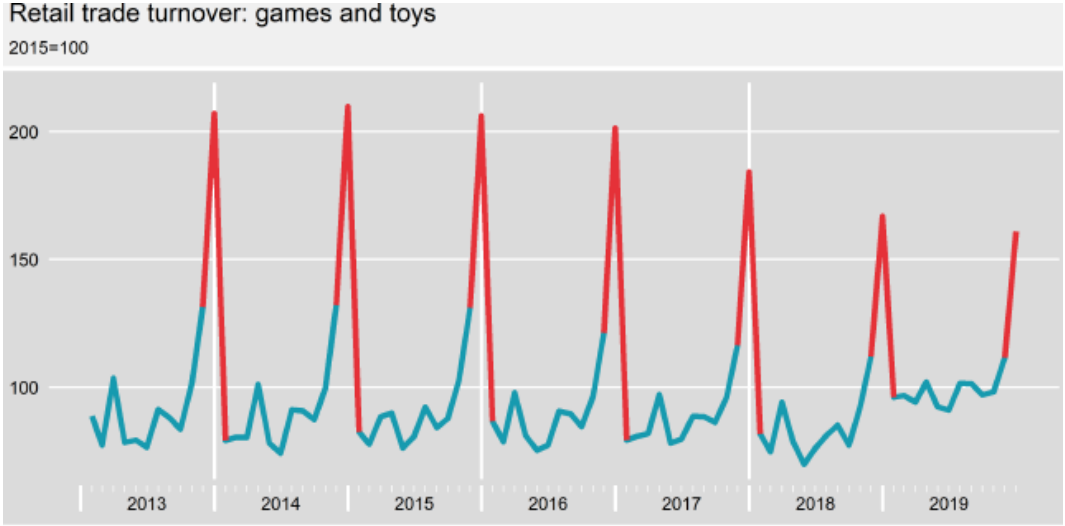
Usual working hypothesis: seasonality evolves slowly ("locally stable")

(the sum of seasonal effects amounts to zero over a year)

Example of seasonal time series (1/2)



Example of seasonal time series (2/2)



The goal of seasonal adjustment

To estimate periodical phenomena and remove them from the raw series.

In order to:

- Identify the short-term evolutions
- Identify medium and long term dynamics
- Highlight expansion and contraction periods, as well as turning points.
- Make comparisons: between economic sectors and between countries with different climates, customs, institutions...

Unobserved components

In order to remove the seasonal part of a series, we must estimate it

To do so we split the raw series **unobservable** components:

- Seasonal component (S) (quasi-periodic)
- Trend-cycle component (T) (long-term evolution)
- Calendar effects (C) (non seasonal calendar variations)
- Irregular component (I) (noise, exceptional events)

Decomposition models (1/2)

The additive model:

$$X_t = T_t + S_t + I_t$$

The multiplicative model:

$$X_t = T_t \times S_t \times I_t$$

The seasonally adjusted series SA = raw series without S_t :

$$SA_t = X_t - S_t \text{ or } SA_t = X_t / S_t$$

When taking the calendar effects into account, the equation becomes :

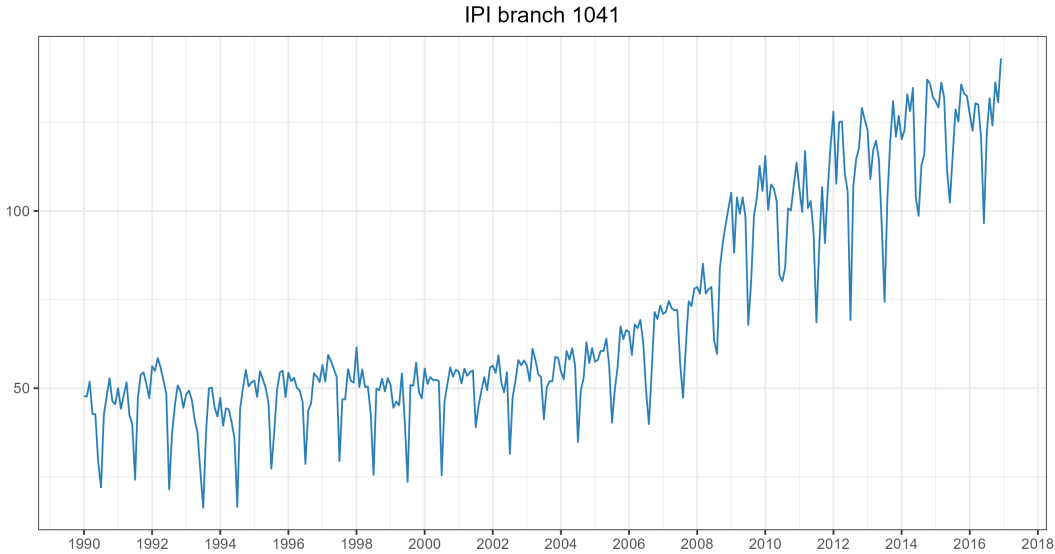
$$X_t = T_t + S_t + I_t + C_t$$

where C_t refer to the calendar effects.

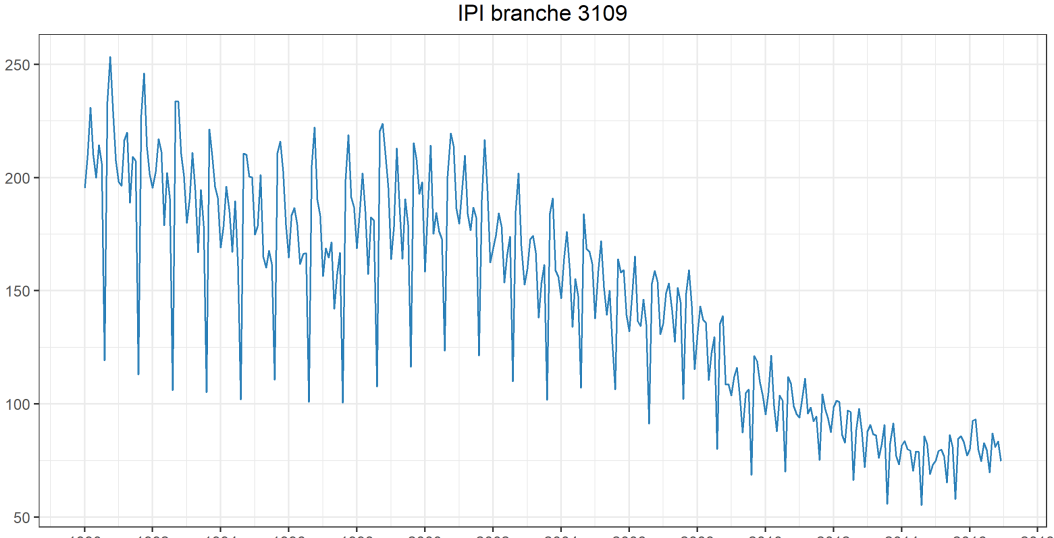
In this case, the SA series = the raw series without S_t and C_t :

$$SA_t = X_t - S_t - C_t \text{ or } SA_t = X_t / (S_t * C_t)$$

Series with an additive model



Series with a multiplicative model



Decomposition models (2/2)

In the most often used version of the decomposition equation, calendar effects (C_t) aren't specifically detailed:

"S_t" is often used as a shortcut for "S_t ("pure" seasonal coefficient) $ + /*{C}_t$ (trading days + easter effect)"

Therefore, the term "seasonally adjusted series" refers to the series corrected for both seasonal and calendar effects.

In JDemetra+, the final coefficient S_t contains both the seasonal and calendar effects.

Seasonal Adjustment in practice: steps

Steps for producing seasonally adjusted data:

- load raw data
- launch estimation
- read diagnostics (seasonality present ? if not sa = raw)
- refine parameters according to diagnostics
- re-estimate (in loop according to diagnostics)
- update the estimation when new data becomes available

Using JDemetra+ GUI for SA

Let's open the Graphical User Interface (GUI)

For installation and resources:

- head over to GitHub repository
  https://github.com/annasmyk/ESTP-ys-JD-webinar

Graphical User Interface organisation

Main window

- Providers
- Workspace
- Results (empty at first)

We will explore this through **Exercise 1** (Workbook) and cover :

- creating and saving a Workspace
- importing data into JDemetra+
- launching a seasonal adjustment process

Addendum to Exercise 1

- import a csv file ('frequency' is 'aggregation frequency')
- Workspace structure as a set of xml files
- what is saved: data and parameters, not results
- link to raw data is saved in the workspace
- menu 'Statistical Methods' (statistical methods)
- Tools> Options' menu to customise "quality report"

Section 4

X13-Arima algorithm

X13-Arima overview

X for eXperience...

X-13 Arima algorithm, available in JD+ GUI, has two modules:

- REG-ARIMA: the pre-adjustment phase
- X11: the decomposition phase

For historical reasons, the decomposition module is still called X11, and even though the current version is X13: the decomposition algorithm hasn't changed much since the X11 era.

In the X11 algorithm the series is split into components via moving averages.

Few elements on moving averages

Moving average: definition

$$MX_t = \sum_{i=-p}^{f} \theta_i X_{t+i}$$

Seasonality suppression: a MA whose order equals the periodicity of the raw series removes a locally stable seasonality

for example: $M_{2\times 12}(X_t)$ for monthly series, $M_{2\times 4}(X_t)$ for quarterly series

Seasonality extraction: is done for each type of periods separately (series of Januaries, then Februaries) with a $M_{3\times k}$ moving average allowing the seasonality to evolve (moving window + decreasing weights)

The iterative principle of X11 (1/2)

Here we show how Moving Averages are used in practice in X11:

First estimation of the adjusted series:

1. Estimation of the **trend-cycle** with a 2×12 MA:

$$TC_t^{(1)} = M_{2\times12}(X_t)$$

2. Estimation of the **seasonal + irregular** component:

$$(S_t + I_t)^{(1)} = X_t - TC_t^{(1)}$$

3. Estimation of the **seasonal** component by applying a 3×3 MA to **each month**:

$$S_t^{(1)} = M_{3\times3}\left[(S_t + I_t)^{(1)}\right] \text{ and normalization } Snorm_t^{(1)} = S_t^{(1)} - M_{2\times12}\left(S_t^{(1)}\right)$$

4. First estimation of the seasonally adjusted series:

$$Xsa_t^{(1)} = (TC_t + I_t)^{(1)} = X_t - Snorm_t^{(1)}$$

The iterative principle of X11 (2/2)

Second estimation of the adjusted series:

1. Refined estimation of the **trend-cycle** with a Henderson filter, which yields a better approximation fo trends than 2×12 MA, but cannot be applied on seasonal series:

$$TC_t^{(2)} = H_{13}(Xsa_t^{(1)})$$

2. Refined estimation of the **seasonal + irregular** part:

$$(S_t + I_t)^{(2)} = X_t - TC_t^{(2)}$$

3. Refined estimation of the **seasonal** component by applying a 3×5 MA (generally) to **each month/quarter**:

$$S_t^{(2)} = M_{3\times5}\left[(S_t + I_t)^{(2)}\right] \text{ and normalization } Snorm_t^{(2)} = S_t^{(2)} - M_{2\times12}\left(S_t^{(2)}\right)$$

4. Final estimation of the seasonally adjusted series:

$$Xsa_t^{(2)} = X_t - Snorm_t^{(2)}$$

Pre-adjustment

Rationale for pre-adjustment:

- moving averages are sensitive to non linearities : linearize the series
- symmetrical moving averages can not be applied at the end of the series: extend the series by forecast (Arima model)

Decomposition will not be performed on the raw series but on the linearized series.

Estimating a linearized and extended series

In X13-ARIMA with Reg-ARIMA model

- **Reg**: stands for regression and allows to correct for deterministic effects
- **Arima** is a class of models allowing to model the residuals in this

All deterministic effects can be modelled with such a regression

- outliers (example AO: dummy variable 000..1...0000, LS)
- calendar effects (the calendar is fully deterministic, specific regressors see below)

Reg-Arima model

Reg-Arima model

$$Y_t = \sum \hat{\alpha}_i O_{it} + \sum \hat{\beta}_j C_{jt} + X_t$$

Linearized series: $X_t = Y_t - \sum \hat{\alpha}_i O_{it} - \sum \hat{\beta}_j C_{jt}$

The regression's BIG residual is modelled with an ARIMA model. (If no deterministic effects, this BIG residual is the raw series)

$$(X_t) \sim ARIMA(p, d, q)(P, D, Q)$$

The Arima structure allows to forecast the series.

Reallocation of preadjustment effects

Beware: calendar effects and outliers have a very different status

- outliers effects will be added back in the SA series (see Covid effects)

$I = I_{lin} + out_i$ (if Additive Outlier)

$T = T_{lin} + out_t$ (if Level shift)

- calendar effects will be permanently removed (appear in the final "S" coefficient)

$S = S_{lin} + cal$

$Y_{lin} = S_{lin} + T_{lin} + I_{lin}$ (from X-11)

$Y = S + T + I$ (Final: see main results)

Worked example: SA without calendar correction

Let's get in the GUI

worked example : Exercise 2

Reading the diagnostics in GUI

Order for reading the diagnostics

- charts
- seasonality tests (if not obvious)
- main results
  - residual seasonality
  - residual calendar effects
  - SI ratios
- summary of pre-processing
- summary of decomposition
- details
  - pre-processing
  - decomposition
- regressors
- intermediate series

Appendix: Pre-defined specifications (see Biblio/JD+ _Pre-defined specifications.pdf)

Calendar correction via linear regression

- Why ? The calendar is heterogeneous, many calendar effects are seasonal but some are not. It prevents two periods of the same type from being comparable before extracting the seasonal effect.
- Example: in IPI series a January with 5 Saturdays, Sundays, Mondays will have a lower production than one with 5 Mondays, Tuesdays, Wednesdays
- Effect of the correction: make periods of the same type comparable
- Method: regressors counting each type of days, holidays are counted as Sundays

Calendar correction in JDemetra+

How to correct for calendar effects using JDemetra+: let's see in the GUI (same features in R packages, see below)

- default regressors (limited number + no national holidays)
- set a customize calendar and use JD+ regressors
- external regressors (Generate your own regressors, possible in R)

Example of regressor sets

- Grouping together different types of days leads to several sets of regressors (explanatory variables).
- Here all sets used at Insee. The variables are formulated in "contrasts" to Sundays (econometric reasons).

Sets of regressors	Hypotheses	Reference (contrast)	Number of regressors
REG1	(Worked Monday = ...= Worked Friday) and (Saturday = Sunday = Bank holidays)	Saturday, Sunday and Bank holidays	1 + LPY
REG2	(Worked Monday = ...= Worked Friday), (Saturday) and (Sunday = Bank holidays)	Sunday and Bank holidays	2 + LPY
REG3	(Worked Monday), (Worked Tuesday = ...= Worked Friday) and (Saturday = Sunday = Bank holidays)	Saturday, Sundays et Bank holidays	3 + LPY
REG5	(Worked Monday), ..., (Worked Friday) and (Saturday = Sunday = Bank holidays)	Saturday, Sunday and Bank holidays	5 + LPY
REG6	(Worked Monday), ..., (Worked Friday), (Saturday) and (Sunday = Bank holidays)	Sunday and Bank holidays	6 + LPY

SA with calendar correction

Let's go through Exercise 3

Most common manual fine tuning operations:

- act on pre-adjustment and spans (series, model)
  - regression variables (calendar, outliers)
  - arima model, mostly to simplify to Airline
- act on decomposition filter length if residual seasonality left
  - for most evolutive periods: shorten the filter

SA with JDemetra+ in R

Using {rjd3x13} R package : a worked example (see code in qmd file)

Section 5

Production utilities and review

Generate Output: definition

Output: 3 types of objects

- series (final and intermediate)
- parameters (automatic, user-defined or resulting from the estimation process)
- diagnostics

Gathered in 2 data structures

- time series organised in data tables
- individual data (for each series: parameters and diagnostics)

Generate output: practice

Let's go through an example !

- Generate output from GUI (demo)
- Use the Cruncher: update workspaces (demo and R code)

The Cruncher

- an additional "executable" module, not an R package
- allows to update a JDemetra+ workspace and export the results

(series and diagnostics, parameters), without having to open the GUI

Hence, suitable for a production process

Infra annual production

When a new data point is available, the producer can use different revision policies:

- forecast seasonal factors (or even the previous year's when no calendar effect)
- automatic classification as an outlier (AO)
- Method "current": the model parameters are not re-identified nor re-estimated, the current model is just applied to the new data point
- partial-concurrent methods: the model parameters are completely re-identified only once a year, but they are re-estimated each time a new point is available.
- Concurrent method: parameters are re-identified and re-estimated with each new point (not recommended on an infra annual basis)

Infra Annual Production

- Refresh policies: an overview (GUI and cruncher)
  - see file Biblio/JD+ - Refresh_Policies_Summary.pdf
  - see GUI options
- Data can also be refreshed in R (not covered here)

SA production steps summary and review

Steps for producing seasonally adjusted data:

- load raw data
- launch estimation
- read diagnostics (seasonality present ? if not sa = raw)
- refine parameters according to diagnostics
- re-estimate (in loop according to diagnostics)
- update the estimation when new data becomes available (refresh policies)

Section 6

Conclusion and useful links

JDemetra+ as a tool for seasonal adjustment

Selection of algorithms: X13-Arima, Tramo-Seats...

Selection of tools: GUI, Cruncher and R packages

What should I use ?

- the GUI and Cruncher require a Workspace structure (cumbersome ?)
- R is more flexible (TS objects)

Use the GUI if manual fine tuning, visual feedback needed

Use the Cruncher if large data sets

Useful Links

To get the Software:

- R Packages giving access to JDemetra+: https://github.com/rjdverse
- Graphical User Interface: https://github.com/jdemetra

Documentation and news:

- Online documentation: https://jdemetra-new-documentation.netlify.app/
- Blog: https://jdemetra-universe-blog.netlify.app/
- YouTube channel (Tutorials, Webinars): https://www.youtube.com/@TSwithJDemetraandR

After the webinar

Assistance with JDemetra+ use and SA production process set up can be provided

As well as additional material, specifically on algorithms (X-11, Reg-Arima, Tramo-Seats)

If you have any questions, just email me

- anna.smyk@insee.fr
- https://github.com/annasmyk

THANK YOU FOR YOUR ATTENTION