Seasonal Adjustment by X-13ARIMA-SEATS in **R**

Christoph Sax
University of Basel

Dirk Eddelbuettel
University of Illinois at Urbana-Champaign

Abstract

seasonal is a powerful interface between **R** and X-13ARIMA-SEATS, the seasonal adjustment software developed by the United States Census Bureau. It offers access to almost all features of X-13, including seasonal adjustment via the X-11 and SEATS approaches, automatic ARIMA model search, outlier detection, and support for user-defined holiday variables such as the Chinese New Year or Indian Diwali. The required X-13 binaries are provided by the **x13binary** package, which facilitates a fully-automatic installation on most platforms. A demo website (at `http://www.seasonal.website`) supports interactive modeling of custom data. A graphical user interface in the **seasonalview** package offers the same functionality locally. X-13 can handle monthly, quarterly or bi-annual time series.

This is a version of our article in the *Journal of Statistical Software* (Sax and Eddelbuettel 2018).

Keywords: seasonal adjustment, time series, X-13ARIMA-SEATS, R.

1. Introduction

Many time series exhibit a regular seasonal pattern over the year. US unemployment, for example, is usually higher from January to March, and again in June and July. Similarly, retail sales tend to peak with the Christmas season.

To model the underlying structure of these series, any regular (seasonal) patterns are estimated and removed from the data. For example, to see if the economy is moving out of a recession during certain months, one wants the labor market data be free from such seasonal effects. Seasonal adjustment decomposes a time series into a trend, a seasonal and an irregular component and removes the seasonal component from the data.

In official statistics, seasonal adjustment has a long tradition. The original X-11 software was developed by the US Census Bureau in the 1960s, and later improved by Statistics Canada (Dagum 1980). Subsequent software packages by the US Census Bureau were called X-12-ARIMA (Findley, Monsell, Bell, Otto, and Chen 1998) and X-13ARIMA-SEATS (or X-13, for short) (Monsell 2007). Today, X-11 is still used as a name for filter-based seasonal adjustment methods within X-13. Meanwhile, TRAMO-SEATS, developed by the Bank of Spain (Caporello, Maravall, and Sánchez 2001), offers an alternative model-based approach to seasonal adjustment.

In its most recent version, X-13 offers these two seasonal adjustment methods in a single command-line tool, written in **Fortran**. The National Bank of Belgium has created an alternative **Java**-based implementation called JDemetra+ (National Bank of Belgium, Deutsche
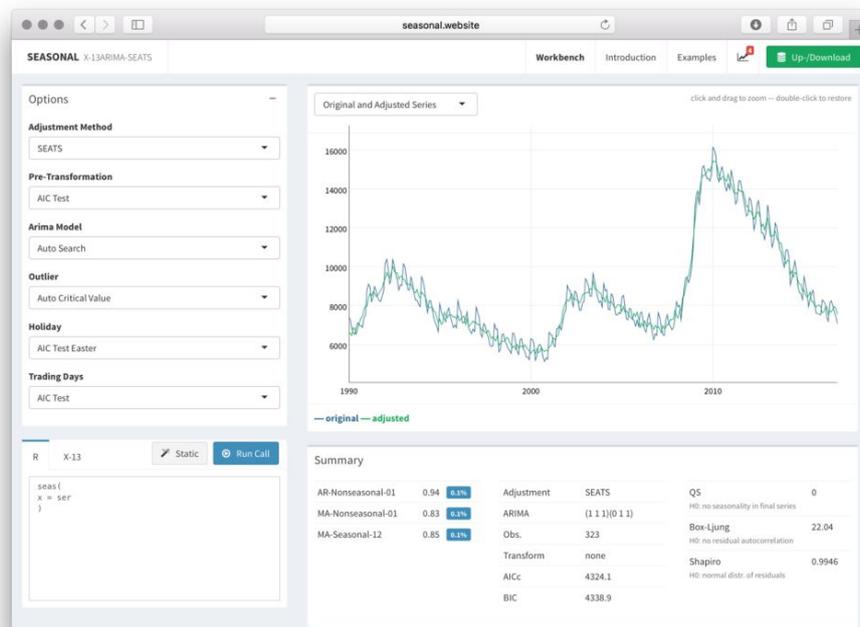
Figure 1: The website `http://www.seasonal.website` gives a full-featured preview of the modeling facilities of **seasonal**, including the possibility to upload and adjust time series.

Bundesbank, Eurostat 2017) which is also widely deployed by statistical agencies. One of either the TRAMO-SEATS or X-11 method of seasonal adjustment is used by almost all (government) statistical offices throughout the world.

R (R Core Team 2017) offers several possibilities to perform seasonal adjustment in the **stats** package included with R. The `decompose` function uses filtering to split a time series into a trend, a seasonal and an irregular component. An alternative method is `stl`, which uses local regressions (and has some extensions offered by the **stlplus** package by Hafen (2016)). While both methods allow a multiplicative or an additive decomposition, these methods are somewhat limited in modeling actual data series.

The programs used by statistical offices provide an extensive toolbox to deal with many advanced (and more irregular) aspects of seasonality, such as trading day adjustments, moving holiday adjustments or automated outlier detection.

To access these features, the **x12** package (Kowarik, Meraner, Templ, and Schopfhauser 2014) interfaces to the older X-12-ARIMA binaries, and can also be used with parts of X-13. Like **seasonal** (Sax 2018), it now depends on **x13binary** (Eddelbuettel and Sax 2017a) to provide the binaries. **x12** hard-codes its available options, which offers limited to no extensibility beyond these pre-programmed options.

The **seasonal** package addresses these shortcomings. Options are translated dynamically into X-13 spec files, which allows the interface to easily evolve with future developments of X-13. **seasonal** depends on the **x13binary** package which provides the binaries of X-13, and removes the need for any manual installation or local compilation from the Fortran sources. This makes it easy for other CRAN packages such as **ggseas** (Ellis 2017) or **gunsales** (Aisch, Keller, and Eddelbuettel 2017) to depend on it. Finally, the syntax of **seasonal** is as R-like as possible; using it should not feel different from using R's built-in functions.

Users new to seasonal adjustment or X-13 should benefit from the automated procedures of **seasonal**, which enables them to quickly produce reasonably good seasonal adjustments of time series. On the other hand, users already familiar with X-13 may benefit from the flexible input and output structure of the package. It permits use of (almost) all of the several hundred of possible arguments to X-13, and can import (almost) all of the nearly two hundred output series that can be generated by X-13.

The package contains several tools to make seasonal adjustment easier as for example a function for easy construction of moving holiday regressors such as Chinese New Year or Indian Diwali. It also offers the capability to import existing X-13 model specs from the command-line tool to R. The **seasonalview** (Sax 2017) package, a powerful graphical user interface based on **shiny** (Chang, Cheng, Allaire, Xie, and McPherson 2017), facilitates the use of X-13 both for beginners and advanced users.

This article does not provide a general overview of seasonal adjustment as there are many good introductions to the topic; e.g., Findley (2005) or Ladiray and Quenneville (2012) for X-11, or Dagum and Bianconcini (2016) for SEATS (signal extraction in ARIMA time series). A very good introduction from a practitioner's viewpoint is provided by UK Office for National Statistics (2007). The canonical reference for X-13 is the official manual by the US Census Bureau (2017), which describes all available options and outputs in detail, and which will be referenced frequently in this article and in the documentation of the package.

Last but not least, to try out X-13 and **seasonal** without installing any software locally, a full-featured website (at `http://www.seasonal.website`) allows uploading and modeling of

time-series interactively, making use of the complete feature set of **seasonal**. Figure 1 shows a screenshot of the website.

The rest of the paper is organized as follows. An introductory example provides overall motivation in the following section. Input and Output formats are discussed in Sections 3 and 4. Section 5 gives an overview of the different graphics choices, before Section 6 introduces the graphical user interface. The creation of customized holiday variables, the use of X-13 and **seasonal** in larger-scale production settings, and the (scripted) import of X-13 model and series specifications are discussed in Sections 7 to 9. Section 10 reviews the automated deployment by the **x13binary** package. Section 11 concludes.

2. An introductory example

As **seasonal** relies (and depends) on the **x13binary** package to access prebuilt binaries of X-13, installing it from CRAN is as easy as installing any other R package:

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

A discussion of the aspects related to automated deployment of the X-13 binary is provided in Section 10.

The `seas` function provides the core functionality of the package. By default, `seas` calls the automatic procedures of X-13 to perform a seasonal adjustment that works well in most circumstances:

```
R> library("seasonal")
R> m <- seas(unemp)
```

The first argument of `seas` is a time series of class `"ts"`. The `unemp` example series measures US unemployment and is included in **seasonal**. The function returns an object of class `"seas"` that contains the necessary information on the adjustment performed on this time series.

There are several functions and methods for `"seas"` objects. The `final` function returns the adjusted series. The `plot` method shows a plot with the unadjusted and the adjusted series (see Figure 2 for an example). The `summary` method displays an overview of the model, very similar to the one produced by R's `lm` function:

```
Call:
seas(x = unemp)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
AR-Nonseasonal-01   0.9436     0.0344    27.4   <2e-16 ***
MA-Nonseasonal-01   0.8254     0.0565    14.6   <2e-16 ***
MA-Seasonal-12      0.8507     0.0336    25.3   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```
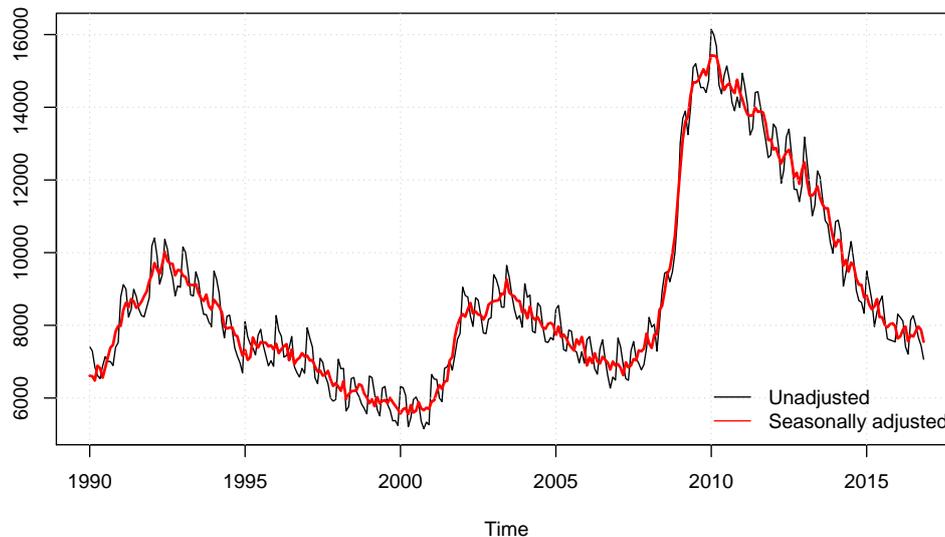
Figure 2: Seasonal adjustment of US unemployment, using the defaults of **seasonal**. The result is close but not identical to the official seasonally adjusted series.

```
SEATS adj.  ARIMA: (1 1 1)(0 1 1)  Obs.: 323  Transform: none
AICc: 4.32e+03, BIC: 4.34e+03  QS (no seasonality in final):   0
Box-Ljung (no autocorr.):   22   Shapiro (normality): 0.995
```

By default, a call to `seas` also invokes the following automatic procedures of X-13:

- Transformation selection (log / no log);

- Detection of trading day and Easter effects;

- Outlier detection;

- ARIMA model search.

In the example above, X-13 opts to perform no pre-transformation, does not adjust for weekday or Easter effects, detects no outliers and models the series by a `(1 1 1)(0 1 1)` seasonal ARIMA (autoregressive integrated moving average) model. This can be seen from the summary output: There are no coefficients for Easter or outliers, and the ARIMA specification and the transformation are shown at the bottom. By default, `seas` calls the SEATS adjustment procedure (which decomposes the ARIMA model). To perform the alternative X-11 adjustment procedure[1], the following option can be used (see the next section for details on how to use arbitrary options with X-13):

```
R> seas(unemp, x11 = "")
```

---

[1]The X-11 method is also used by the US Bureau of Labor Statistics (BLS), in order to calculate official seasonally adjusted unemployment numbers. The numbers obtained in the examples above will not match the BLS numbers, for various reasons. One reason is that the BLS is adjusting unemployment by industry, before aggregating them. The BLS also applies a more sophisticated modeling of trading days and outliers. For details, see US Bureau of Labor Statistics (2016).

Instead of relying on automated X-13 procedures, inputs may be entered 'statically', as in this example:

```
R> seas(x = unemp,
+       arima.model = "(1 1 1)(0 1 1)",
+       regression.aictest = NULL,
+       outlier = NULL,
+       transform.function = "none"
+       )
```

Here, the specification of the 'arima.model' overrides and turns off the automated model selection. AIC testing of the regressors, outlier detection and transformation are disabled. The next section describes the handling of mutually exclusive inputs in more detail.

The `static` function substitutes all automated procedures by their static equivalents. The call above can be generated easily from our initial, fully automated model:

```
R> static(m)
```

With the **seasonalview** package installed, the `view` command offers an easy way to analyze and modify a seasonal adjustment procedure, in the same way as it is possible on the website at http://www.seasonal.website:

```
R> view(m)
```

and is discussed further below.

3. Input

The X-13 software has several hundred options, and is capable of producing almost two hundred different output series. In order to keep the program code concise and to prepare for future developments of X-13, **seasonal** passes these arguments directly to X-13, rather than hard-coding them as R arguments.

As for the input, it is possible to use almost[2] the complete syntax of X-13 via the ... argument in the `seas` function. The input of X-13 is organized in *specs* and *arguments*, with each spec optionally containing some arguments (in the X-13 sense). These spec-argument combinations can be added to `seas` by separating the spec and the argument by a dot (.). For example, in order to set the 'variables' argument of the 'regression' spec equal to `td` and `ls2009.11`, and turn off the default 'aictest' argument, the input to `seas` can be specified as follows:

```
R> seas(unemp,
+       regression.variables = c("td", "ls2009.11"),
+       regression.aictest = NULL
+       )
```

---

[2]The only exception is the 'composite' spec, which is at odds with the univariate syntax of `seas` but is straightforward to replicate in basic R. For an example, see `help("seas")`.

Note that R vectors may be used as an input. If a spec is added without any arguments (in the X-13 sense), the spec should be set equal to an empty string. Several defaults of `seas` are empty strings, such as the default `seats = ""` (see `help(seas)` for details). Note the difference between `""` (meaning the spec is enabled but has no arguments, in the X-13 sense) and `NULL` (meaning the spec is disabled).

It is possible to manipulate almost all inputs to X-13 in that way. Chapter 7 of the X-13 manual (US Census Bureau 2016) describes all available spec-argument combinations in detail. The comprehensive list of examples available at http://www.seasonal.website/examples. html offers numerous illustrations about the mapping of both functionality and syntax between X-13 and **seasonal**.

For instance, the first example in Section 7.2 from the X-13 manual (US Census Bureau 2017),

```
series       { title = "Monthly sales"
               start = 1976.jan
               file = "ussales.dat"
             }
regression  { variables = (td seasonal) }
automdl      {   }
estimate     {   }
x11          {   }
```

can be computed in R as follows (using a dataset from R for simplicity):

```
R> seas(AirPassengers,
+       x11 = "",
+       regression.variables = c("td", "seasonal"),
+       regression.aictest = NULL
+       )
```

`seas` takes care of the 'series' spec, and no input beside the time series has to be provided. As `seas` uses the SEATS procedure by default, the use of X-11 has to be specified explicitly. When the 'x11' spec is added as an input (as above), the mutually exclusive default 'seats' spec is automatically disabled. `automdl` and `estimate` are used by default. With `regression.variables`, an additional spec-argument is added to the input, which includes trading day and stable seasonal regression effects. Because `seas` performs an AIC test on the regression variables by default, it is disabled by setting `regression.aictest = NULL`.

There are some mutually exclusive specs in X-13. If more than one mutually exclusive spec is included in `seas`, specs are overwritten according the following priority rules:

- Model selection:

  1. `arima`,
  2. `pickmdl`,
  3. `automdl` (default).

- Adjustment procedure:

1. `x11`,
2. `seats` (default).

This is why the default SEATS procedure in the introductory example was overwritten by the specification of `x11 = ""`. For details on the differences between these methods, consider the manual of X-13 (US Census Bureau 2017).

`seas` also accepts additional time series as external regressors or transformation variables via the `xreg` and `xtrans` arguments. The following example includes an external regressor for a level shift outlier from November 2009 to end of the series:

```
R> lshift <- ts(-1, start = c(1990, 1), end = c(2019, 11), freq = 12)
R> window(lshift, start = c(2009, 11)) <- 0
R> seas(unemp, xreg = lshift, outlier = NULL)
```

This can be used to replicate the built-in outlier adjustment of X-13:

```
R> seas(unemp, regression.variables = "ls2009.11", outlier = NULL)
```

As an alternative to the `...` argument, spec-arguments can also be supplied as a named list which is particularly useful for non-interactive / programming use:

```
R> seas(list = list(x = unemp, x11 = ""))
```

Additionally, `"seas"` objects can be altered using the `update` function method. It uses the same syntax as the `seas` function. The following example changes the ARIMA model and switches the adjustment method to X-11:

```
R> update(m, arima.model = "(0 1 1)(0 1 1)", x11 = "")
```

A common use of `update` involves recomputing with new data. Using new data with an existing specification can be done using the update function::

```
R> update(m, x = unemp)
```

Alternatively, if only the final seasonally adjusted series is required, the `predict` function method can be used with the familiar `newdata` argument:

```
R> predict(m, newdata = unemp)
```

4. Output

Comparable to the processing of input arguments, **seasonal** has a flexible, non hard-coded mechanism to extract data from X-13. With the `series` function, it is possible to import almost all output that can be generated. For example, the following command returns the forecasts of the ARIMA model as a `"ts"` time series:

```
R> m <- seas(unemp)
R> series(m, "forecast.forecasts")
```

X-13 requires the desired output to be specified in the input. Because the `forecast.save = "forecasts"` argument has not been specified in the model call, `series` *re-evaluates* the call with the 'forecast' spec enabled. It is also possible to return more than one output table at the same time:

```
R> series(m, c("forecast.forecasts", "s12"))
```

The user can use either the unique short names of X-13 (such as `s12`, designating the final trend component from SEATS), or the long names (such as `forecasts`, designating the forecasts of the ARIMA model). Because the long table names are not unique, they need to be combined with the spec name (`forecast`). See `help(series)` for a list of all available options.

Note that re-evaluation doubles the overall computation time, as X-13 has to run twice. To only run once and to speed it up, the model call should include the output explicitly in non-interactive use:

```
R> m.save <- seas(unemp, forecast.save = "forecasts")
R> series(m.save, "forecast.forecasts")
```

Some specs such as 'slidingspans' and 'history' can result in time-consuming computations. In interactive use, the re-evaluation feature is especially useful, as it allows to separate the time-consuming diagnostic computations from the fast basic model call:

```
R> m <- seas(unemp)
R> series(m, "history.saestimates")
R> series(m, "slidingspans.sfspans")
```

The `out` function shows the content of the main X-13 output in the browser, which gives access to every detail of the estimation.

```
R> out(m)
```

In order to process statistics in R, the `udg` function (which stands for *unified diagnostic file*) gives a flexible access to all statistics from X-13:

```
R> udg(m)
```

There are also wrappers that implement methods for the standard model diagnostics in R: `AIC`, `BIC`, `logLik`, `nobs`.

5. Graphs
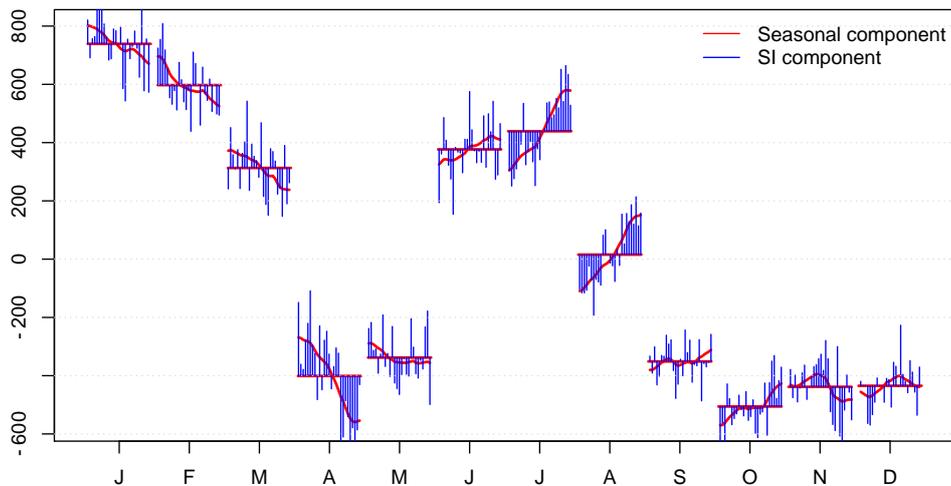
Figure 3: The `monthplot` output shows the seasonal and seasonal-irregular (SI) component of US unemployment for each month. The SI-component shows the detrended observations as the combined seasonal and irregular component. The straight line shows the average seasonal component over the whole sample period. US unemployment is usually higher from January to March, and again in June and July – the winter peak got weaker, the summer peak got more accentuated over time.

Several graphical tools are available to analyze a `"seas"` object. The main `plot` method draws the seasonally adjusted and unadjusted series (as shown in Figure 2). If outliers were detected, these will also be indicated. The `monthplot` method produces a monthwise plot (or quarterwise, with the same misleading name as the generic R function) of the seasonal and the SI component:

```
R> monthplot(m)
```

An example of an output from `monthplot` is given in Figure 3.

Also, many standard R functions can be used to analyze a `"seas"` object:

```
R> pacf(resid(m))
R> spectrum(resid(m))
R> plot(density(resid(m)))
R> qqnorm(resid(m))
```

The `identify` method can be used to interactively select (or unselect) outliers by pointing and clicking; clicking several times on an outlier permits to loop through different outlier types:

```
R> identify(m)
```

6. Graphical user interface
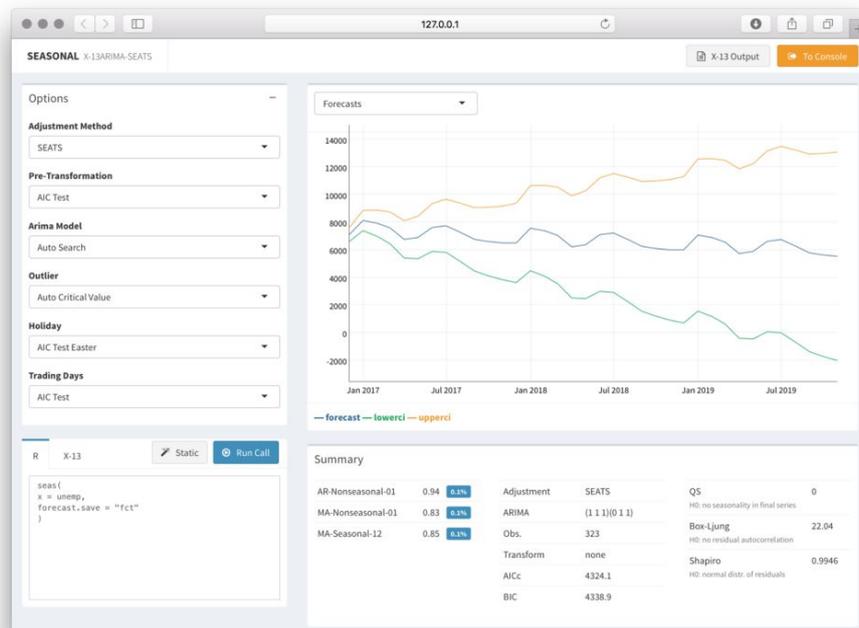
Figure 4: The `view` function calls a graphical user interface that allows an interactive manipulation of most options of X-13.

The **seasonalview** package provides a graphical interface for choosing a seasonal adjustment model using **shiny** (Chang *et al.* 2017). It provides the identical look and feel as the demo website at `http://www.seasonal.website`. To install the latest version of **seasonalview**, use:

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

The goal of the `view` function is to summarize all relevant options, plots and statistics that form part of the seasonal adjustment process. `view` uses a `"seas"` object as its main argument and returns an updated `"seas"` object. Figure 4 depicts a screenshot of the `view` function in action.

```
R> view(m)
```

Frequently-used options can be modified using the drop-down selectors in the upper-left panel. Each change will result in a re-estimation of the seasonal adjustment model. The R-Call, the output and the summary are updated accordingly.

Alternatively, the R-Call can be modified manually in the lower left panel. Press 'Run Call' to re-estimate the model and to adjust the options selectors, the output, and the summary. With the 'To Console' button, `view` is closed and an updated model is returned to the R console. The 'Static' button substitutes automatic procedures by their corresponding static spec-argument options, in the same way as the `static` function. Users familiar with X-13 can

also inspect and modify the X-13-Call, which can be accessed by the tab next to the R-Call. This also serves as a learning tool for users familiar with **seasonal** but not with X-13, and vice versa.

The views in the upper right panel can be selected from the drop-down menu.

The lower right panel shows the summary, including the same information as the `summary` function method applied to a `"seas"` object. The 'X-13 output' button opens the complete output of X-13 in a separate tab or window.

7. Customized holidays

seasonal includes `genhol`, a function that makes it easy to model user-defined holiday regression effects, such as the Chinese New Year, Indian Diwali or complex pre- and post-holiday adjustments for Easter. `genhol` is an R replacement for the equally named software by the Census Bureau. The function uses an object of class `"Date"` as its first argument, which specifies the occurrence of the holiday, and returns a time series that can be used as a regressor.

Dates for some major moving holidays are included in the package (`easter` (Easter), `cny` (Chinese New Year), `diwali` (Indian Diwali)) and can be used with `genhol`. For example, in order to adjust Indian industrial production for Diwali effects, use:

```
R> seas(iip,
+       x11 = "",
+       xreg = genhol(diwali, start = 0, end = 0, center = "calendar"),
+       regression.usertype = "holiday"
+       )
```

`iip` is an example series and included in the package. Several examples, which cover Chinese New Year and complex pre- and post-holiday adjustments for Easter, are provided in the help page which can be accessed from R via `help("genhol")`.

8. Large-scale production use

While **seasonal** offers a fast and convenient way to adjust a single time series in R, it is equally well suited for processing large numbers of time series. Two kinds of seasonal adjustments are often used in a large-scale production environment:

1. a recurring application of a given adjustment model to a time series;

2. an automated adjustment to a large number of time series.

This section shows how both tasks can be accomplished with **seasonal** and basic R.

8.1. Storing calls and batch processing

Seasonal adjustment models can be stored (using `save`) and re-evaluated at a later date when new data becomes available. Here, the `unemp` example series is shortened by one month, so that the following represents a model estimation with data available up to October 2016:

```
R> unemp.oct16 <- window(unemp, end = c(2016, 10))
R> m.oct16 <- seas(unemp.oct16)
```

The `static` function has been introduced in the introductory example. It is particularly useful to save the specifics of a model (e.g., outliers, ARIMA specification) for future use. For example, in the model above, the automated procedures decided to use an (1 1 2)(0 1 1) ARIMA specification and to perform no pre-transformation or weekday or Easter adjustments. The static function 'hard-codes' these findings into the model call, so that these decisions would not be re-evaluated at a later date:

```
R> m.oct16.static <- static(m.oct16, evaluate = TRUE)
```

With `evaluate = TRUE`, `static` returns an object of class `"seas"`, rather than just the call. Using data up to October 2016, the automated model, `m.oct16`, produces the same results as the static model, `m.oct16.static`. By default, evaluating the static model still re-estimates the ARIMA model parameters and the X-11 filters (if applicable). To avoid re-estimation, the arguments `coef` and `x11.filter` can be set to `TRUE`.

Both models can be re-evaluated when new data becomes available. Here, the `update` function is used to update the existing models with the full `unemp` series, which ranges up to November 2016:

```
R> update(m.oct16, x = unemp)
R> update(m.oct16.static, x = unemp)
```

There is a crucial difference between these models at this later date: while the automated model search has opted for an (1 1 2)(0 1 1) ARIMA specification in October, it opts for an (1 1 1)(0 1 1) model in November, as has been demonstrated in the introductory example. By hard-coding the ARIMA specification into the call, it is ensured that the ARIMA model (and the other specifications of the model) do not change from month to month. Alternatively, in some production settings, it may be useful to copy the static call to a script, and re-run it when new data becomes available.

8.2. Automated adjustment of multiple series

X-13 can also be applied programmatically to a large number of series using automated adjustment methods. This can be accomplished with a loop or one of the `*apply` functions in R. A good pattern of "defensive programming" is to wrap the call to `seas` in a `try` statement. Should an error occur, it will be trapped and not lead to a break in the execution. One needs to develop an error handling strategy for these cases. Possibilities for handling such error messages when processing one or more series are to either drop them, use them without adjustment or switch to a different automated routine.

For example, time series can be collected in a `list` and looped over, using `lapply`:

```
R> dta <- list(fdeaths = fdeaths, mdeaths = mdeaths)
R> ll <- lapply(dta, function(e) try(seas(e, x11 = "")))
```

Failing models can be listed as follows:

```
R> is.err <- sapply(ll, class) == "try-error"
R> ll[is.err]
```

The following returns final series of successful evaluations:

```
R> do.call(cbind, lapply(ll[!is.err], final))
```

To speed up the adjustment of multiple series, this process above is well suited for parallelization. The vignette, which can be accessed via `vignette("seasonal")`, provides an example.

In many production settings, it is preferable to store the static model and re-evaluate with new data at a later point. This can be achieved as follows:

```
R> ll.static <- lapply(ll, static, evaluate = TRUE)
R> Map(predict, ll.static, newdata = dta)
```

9. Import X-13 models and series

Users of the X-13 command-line tool have two utility functions to import existing `.spc` files and data. The `import.spc` function will construct the corresponding call to `seas` and the calls for importing the data from an existing X-13 `.spc` file.

For example, the following example imports the `Testairline.spc` file that is shipped with X-13 (and also included in **seasonal**):

```
R> import.spc(system.file("tests", "Testairline.spc", package="seasonal"))
```

If data is stored outside the `.spc` file (as it usually will be), the calls will make use of the `import.ts` function to import the data referenced in the `.spc` file as R time series. An example is given in the documentation (see `help("import.ts")`).

10. Deployment via x13binary

X-13 has a long and proud history of use and deployment by countless users, including many governmental statistical agencies as discussed above. During most of the several decades of widespread use, analysts deploying X-13 had to *manually* download the corresponding program from the Census website. (Or rely on their IT department to do it for them.)

The rise of R to pre-eminence as a statistical environment has highlighted how useful and important automated deployment via package repositories is. Thousands of packages are available, and can be installed via a simple command, function call or click of a button. This triggers either a download of a prebuilt binary version in the case of the Windows and macOS operating systems, or the download and automated installation from source for machines running Unix or Linux.

Early use of **seasonal**, however, relied on the standard installation procedure devised by US Census for X-13. Users either had to download a binary pre-built for a number of operating

systems, or build a binary from the supplied Fortran source code. After that first manual step, a second step was required to communicate the presence, and location, of the binary to the package using it, in this case **seasonal**. This process was both overly manual, error-prone and platform-dependent. But it was not for lack of a suitable license. Works by the US Government are generally in the public domain and can be re-distributed. So a better solution was always conceivable.

The **x13binary** package (Eddelbuettel and Sax 2017a) addresses this need by utilizing the tried-and-tested CRAN package mechanism. By being set-up as a package, two entry points for customizations are available: build-time, and run-time. At package build-time, the script `configure` determines the type of operating system it is being used on, and acts accordingly to provide either a binary, or creates one via compilations. At run-time when the package is loaded, it provides several convenience functions to assess the validity of its X-13 installation, or to communicate the path of the binary to other packages requiring it—as for example **seasonal**.

The **x13binary** relies on another external resource to provide it with "always accessible" sources or binaries in a structured form. The **x13prebuilt** repository (Eddelbuettel and Sax 2017b) provides this in the form of statically compiled binaries for Linux and macOS, as well as a zip archive with a Windows binary. **x13binary** is set up to utilize these files and automatically place them in the expected location for the respective operating system and variant. Taken together, packages **x13binary** and **x13prebuilt** ensure that users executing the step recommended in the introductory section

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

will be provided with a fully functional installation including X-13 without any manual invention. Thanks to the R package system, **seasonal** can depend on **x13binary** which by utilizing **x13prebuilt** will ensure a correct and complete installation ready to be deployed for seasonal adjustment.

If the binaries cannot be downloaded (e.g., because of a firewall), it is still possible to manually download the binaries and tell **seasonal** where to find them. `help("seasonal")` provides additional details on this. Updating **x13binary** follows the schedule of the US Census Bureau, which usually updates once a year. Version numbers of **x13binary** are aligned with version numbers of the Census Bureau, and older versions can be easily installed from the CRAN archives.

11. Conclusions

X-13ARIMA-SEATS is a very powerful tool for seasonal adjustment. It is used by numerous statistical offices around the world. **seasonal** brings the full functionality of X-13 to R. This considerably lowers the barrier to increased use of these powerful seasonal adjustment methods given the widespread and ever increasing popularity of R.

The **seasonal** package is designed so that the use of its principal function `seas` feels very similar to key modeling functions of R such as the ubiquitous `lm` method. Data structures in the **seasonal** package interact seamlessly with the built-in time series objects for R which makes it easy to construct custom regressors or holiday variables. The input and output facilities

are flexible enough to support the full functionality of X-13 and permit easy adoption to future versions. In addition, the **seasonalview** package adds a graphical user interface which simplifies the manipulation of seasonal adjustment models. The **x13binary** package automates dependency resolution and installation of the required binaries via the time-tested R package system, and allows other packages to extend **seasonal**.

Acknowledgments

seasonal was originally developed for use at the Swiss State Secretariat of Economic Affairs. It has been greatly improved over time thanks to suggestions and support from Matthias Bannert, Freya Beamish, Vidur Dhanda, Alain Galli, Ronald Indergand, Preetha Kalambaden, Stefan Leist, James Livsey, Pinaki Mukherjee, Bruno Parnisari, five anonymous reviewers and many others. The related work on the **x13binary** package facilitated (automated) deployment thanks to the R package system, CRAN as well as GitHub for the **x13prebuilt** repository. Help and support by Brian Monsell and the United States Census Bureau is gratefully acknowledged as is, of course, their seasonal adjustment software.

References

Aisch G, Keller J, Eddelbuettel D (2017). **gunsales**: *Statistical Analysis of Monthly Background Checks of Gun Purchases*. R package version 0.1.2, URL https://CRAN.R-project.org/package=gunsales.

Caporello G, Maravall A, Sánchez FJ (2001). "Program TSW Reference Manual." *Technical Report 0112*, Banco de España Madrid. URL https://ideas.repec.org/p/bde/wpaper/0112.html.

Chang W, Cheng J, Allaire J, Xie Y, McPherson J (2017). **shiny**: *Web Application Framework for R*. R package version 1.0.5, URL https://CRAN.R-project.org/package=shiny.

Dagum EB (1980). *The X-11-ARIMA Seasonal Adjustment Method*. Statistics Canada, Seasonal Adjustment and Time Series Staff.

Dagum EB, Bianconcini S (2016). "Seasonal Adjustment Based on ARIMA Model Decomposition: TRAMO-SEATS." In *Seasonal Adjustment Methods and Real Time Trend-Cycle Estimation*, pp. 115–145. Springer.

Eddelbuettel D, Sax C (2017a). **x13binary**: *Provide the 'x13ashtml' Seasonal Adjustment Binary*. R package version 1.1.39-1, URL https://CRAN.R-project.org/package=x13binary.

Eddelbuettel D, Sax C (2017b). **x13prebuilt**: *Pre-built Binaries of X-13ARIMA-SEATS*. URL https://github.com/x13org/x13prebuilt.

Ellis P (2017). **ggseas**: *'stats' for Seasonal Adjustment on the Fly with **ggplot2***. R package version 0.5.1, URL https://CRAN.R-project.org/package=ggseas.

Findley DF (2005). "Some Recent Developments and Directions in Seasonal Adjustment." *Journal of Official Statistics*, **21**(2), 343. URL https://www.census.gov/ts/papers/recentdevelopmentsjos.pdf.

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.

Hafen R (2016). **stlplus**: *Enhanced Seasonal Decomposition of Time Series by Loess*. R package version 0.5.1, URL https://CRAN.R-project.org/package=stlplus.

Kowarik A, Meraner A, Templ M, Schopfhauser D (2014). "Seasonal Adjustment with the R Packages **x12** and **x12GUI**." *Journal of Statistical Software*, **62**(1), 1–21. ISSN 1548-7660. doi:10.18637/jss.v062.i02. URL https://www.jstatsoft.org/index.php/jss/article/view/v062i02.

Ladiray D, Quenneville B (2012). *Seasonal Adjustment with the X-11 Method*, volume 158. Springer Science & Business Media.

Monsell B (2007). "The X-13A-S Seasonal Adjustment Program." In *Proceedings of the 2007 Federal Committee On Statistical Methodology Research Conference*. URL http://www.fcsm.gov/07papers/Monsell.II-B.pdf.

National Bank of Belgium, Deutsche Bundesbank, Eurostat (2017). *JDemetra+: Econometric Software for Seasonal Adjustment and Other Time Series Methods*. Eurostat. URL https://ec.europa.eu/eurostat/cros/content/download.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Sax C (2017). **seasonalview**: *Graphical User Interface for Seasonal Adjustment*. R package version 0.3, URL https://CRAN.R-project.org/package=seasonalview.

Sax C (2018). **seasonal**: *R Interface to X-13-ARIMA-SEATS*. R package version 1.7, URL https://CRAN.R-project.org/package=seasonal.

Sax C, Eddelbuettel D (2018). "Seasonal Adjustment by X-13ARIMA-SEATS in R." *Journal of Statistical Software*, **87**(11), 1–17. doi:10.18637/jss.v087.i11.

UK Office for National Statistics (2007). *Guide to Seasonal Adjustment with X-12-ARIMA*. URL http://www.ons.gov.uk/ons/guide-method/method-quality/general-methodology/time-series-analysis/guide-to-seasonal-adjustment.pdf.

US Bureau of Labor Statistics (2016). "Current Employment Statistics." http://www.bls.gov/web/empsit/cesseasadj.htm. Accessed: 2016-12-12.

US Census Bureau (2017). *X-13ARIMA-SEATS Reference Manual*. Time Series Research Staff, Center for Statistical Research and Methodology, US Census Bureau, Washington, DC, USA, version 1.1 edition. URL http://www.census.gov/ts/x13as/docX13ASHTML.pdf.

Affiliation:

Christoph Sax
University of Basel
Peter Merian-Weg 6, Basel, Switzerland
ORCID: 0000-0002-7192-7044
E-mail: christoph.sax@gmail.com
URL: http://www.christophsax.com

Dirk Eddelbuettel
University of Illinois at Urbana-Champaign
725 S Wright St, Chicago, USA
ORCID: 0000-0001-6419-907X
E-mail: edd@debian.org
URL: http://dirk.eddelbuettel.com