Discussing X-13ARIMA-SEATS in R

James Livsey and Christoph Sax

Abstract

This vignette shows how R can be used to simplify discussions on seasonal adjustment. Thanks to the R packages *seasonal* and *x13binary*, installing X-13ARIMA-SEATS is as easy as installing any other R package. Which paves the path to a simple workflow of discussing and exchanging seasonal adjustment problems. Additionally, our new *x13story* package allows to document and publish discussions both as a traditional documentation or as an interactive online *story*.

Introduction

For a variety of reasons, many statistical agencies perform seasonal adjustment. Of the multiples different choices for seasonal adjustment methodology and software, a large number of agencies, including the U.S. Census Bureau, choose to use X-13ARIMA-SEATS. The *big picture* ideas behind seasonal adjustment are well understood and easily explained.

However, once it comes time to perform a seasonal adjustment, an inexperienced user can become easily overwhelmed with the options and diagnostics of most seasonal adjustment software. This, coupled with the associated learning costs, yields many people who could benefit from a better understanding of the X -13ARIMA-SEATS process.

The spread of open source R in statistical agencies makes it possible to simplify discussions on X-13 and to flatten the learning curve. Thanks to the R packages *seasonal* and *x13binary*, installing X -13ARIMA-SEATS is as easy as installing any other R package. Running X-13 in R is mostly like running any built-in R function. These allows for a simple way of discussing and exchanging seasonal adjustment problems, as we will show in Section 2.

While R, *seasonal* and *x13binary* offer a useful way to discuss and exchange seasonal adjustment problems, our new *x13story* package allows you to document and publish discussions both as a traditional documentation or as an interactive online lesson, or *story*. A few examples stories are integrated on this website. The transformation of an R script discussing a seasonal adjustment problem to a PDF or an interactive story will be described in the Section 3 and 4.

Summarizing Seasonal Adjustment Problems in an R Script

With the spread of R in statistical offices, R scripts are becoming one of the best ways to talk about statistical problems. This is because R scripts can include code (of course), but also comments and, most importantly, data.

The last point is under-appreciated in practice, as it brings large benefits during exchange. The `dput()` function writes a text representation of an R object to the console, from where it can be copied and inserted into your R script.

Here is an example on how to use `dput` on the built in example series `AirPassengers`:

```
dput(AirPassengers)
```

This generates the code representing the `AirPassengers` time series. Copy-pasted, the equivalent of `AirPassengers` is:

```
myAirPassengers <- structure(c(112, 118, 132, 129, 121, 135, 148, 148, 136, 119,
104, 118, 115, 126, 141, 135, 125, 149, 170, 170, 158, 133, 114,
140, 145, 150, 178, 163, 172, 178, 199, 199, 184, 162, 146, 166,
171, 180, 193, 181, 183, 218, 230, 242, 209, 191, 172, 194, 196,
196, 236, 235, 229, 243, 264, 272, 237, 211, 180, 201, 204, 188,
235, 227, 234, 264, 302, 293, 259, 229, 203, 229, 242, 233, 267,
269, 270, 315, 364, 347, 312, 274, 237, 278, 284, 277, 317, 313,
318, 374, 413, 405, 355, 306, 271, 306, 315, 301, 356, 348, 355,
422, 465, 467, 404, 347, 305, 336, 340, 318, 362, 348, 363, 435,
491, 505, 404, 359, 310, 337, 360, 342, 406, 396, 420, 472, 548,
559, 463, 407, 362, 405, 417, 391, 419, 461, 472, 535, 622, 606,
508, 461, 390, 432), .Tsp = c(1949, 1960.91666666667, 12), class = "ts")
```

Why is this great? Because there is no need to accompany your script with a data file! There is no need for the receiver of your script to adjust the file path that points to your datafile. All you need is an installation of R to run the script, even without understanding your code.

Putting your data, your comments and your data into the same R script has many nice applications:

1. You can copy-paste a discussion in an email.
2. You can put it to GitHub, or GitHub Gist, or any other code sharing platform (Example).
3. You can put it on your website.
4. You can use it in mailing lists or question/answer sites such as CrossValidated or StackOverflow (Example).
5. You can transform it to a PDF documentation or an interactive online story (see the next section).

While this approach is useful for many statistical problems, it is especially useful for seasonal adjustment problems. Thanks to *x13binary* and *seasonal*, getting access to X-13ARIMA-SEATS in R is as easy as:

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

This automatically installs the *seasonal* package with the latest build of X-13 on any platform (Linux, Mac or Windows). If you want to make the installation of *seasonal* conditional on not being installed already, you can start your R script with the following line:

```
if (!require("seasonal")) install.packages("seasonal"); library(seasonal)
```

which ensures there is a working version of X-13 and *seasonal*. *seasonal* is an easy-to-use R-interface to X-13ARIMA-SEATS, which allows you to use (almost) all features of X-13. For a more detailed description, consider the vignette.

`seas` is the core function of the seasonal package. By default, `seas` calls the automatic procedures of X-13ARIMA-SEATS to perform a seasonal adjustment that works well in many circumstances:

```
m <- seas(AirPassengers)
```

By default, `seas` calls the SEATS adjustment procedure. If you prefer X-11, you can use:

```
seas(AirPassengers, x11 = "")
```

As we have seen, it is handy to put the data directly into the script. This is especially true if we have external regressors or transformation variables, which can be used as follows. Here, we are modeling a temporary level shift, by manually constructing an R time series:

```
tls <- ts(0, start = 1949, end = 1965, freq = 12)
window(tls, start = c(1955, 1), end = c(1957, 12)) <- 1
seas(AirPassengers, xreg = tls, outlier = NULL, x11 = "")
```

If you need specific holiday regressors, the built in `genhol` function may be useful. It is also possible to import an existing X-13ARIMA-SEATS model into R. For details, see the corresponding section in the vignette.

Putting your data, your comments and your code into the same R script is crucial, but there are a few other things that facilitate the exchange and the discussion of statistical problems:

1. *Do not read from local files*. If you cannot make your data part of the script, e.g., because it is too large, put it as a `.csv` file on a fixed URL (e.g., on GitHub) and use the R function `read.csv` to directly read from there. `read.csv` can take an URL as an argument; there is no need to download and read in separate steps.
2. *Use as few external packages as possible*. Try to use R base and avoid external packages such as *ggplot2*, *dplyr*, *data.table* etc. Using external packages makes it harder to get the same environment on somebody else's machine. Also, packages that seem familiar to you may not be familiar to somebody else. Basic R is the common language that you can expect the other person to understand. If you really need packages (e.g., seasonal), install them conditionally, as shown above.

From Script to Document

Having kept a statistical discussion in a single R script, it is easy to transform it into a reproducible research note or an interactive *story*. The *x13story* package includes several tools to facilitate this task.

x13story is still under development and not yet on CRAN. To install from GitHub, use:

```
devtools::install_github("christophsax/x13story")
```

x13story is based on the R Markdown package, which allows you to easily author reproducible documents in R, by combining *chunks* or R code with text written in *Markdown*. Markdown is a very lightweight markup language that allows you to structure and format your text.

The *x13story* package contains an R Markdown template with the same name, which is specifically optimized for time series problems. It uses wider figure margins and a unified font for graphs and text (Palatino). However, you are not restricted to the use of *x13story*, and most utility functions in the *x13story* package also work with the popular `tufte_handout` style, which is part of the *R Markdown* package.

The *R Markdown* package is nicely integrated into RStudio, but it is also easy to use it in other R environments. To create a fresh X-13 story template in any R environment, use:

```
rmarkdown::draft("MyArticle.Rmd", template = "x13story", package = "x13story")
```

In the latest versions of RStudio, you can select the template from the menu:

```
New Document Symbol > R Markdown ... > From Template > X-13 Handout
```

The *x13story* package comes with a bunch of functions designed to make the production of pretty PDFs easy. A more extensive description with examples is given in the template, which explains its use.

From Document to Story

While PDF documents are great to summarize research on seasonal adjustment, interactive learning stories offer an even better opportunity to share your knowledge with a broader audience. From an R Markdown *x13story*, it is only a minor step to create an interactive lesson, which we call a *story*.

A story contains one or several *views*. A view visualizes an R object of class `"seas"` – essentially an X-13 adjustment. It shows a particular aspect of an adjustment, which is specified by the `series` argument. A view is shown in the interactive online tool, as it is currently available here. As all seasonal adjustments in the online tool, views can be interactively manipulated by the user.

To initialize a view, use the `x13page` function in an R chunk:

```
m <- seas(AirPassengers, x11 = "")
x13page(m, 'x11.seasonal')
```

The *x13story* template described above has more examples on how to use the `x13page` function.

To generate an **interactive story** from your Markdown document, use the `view` function from the seasonalview package. The function accepts a local or a remote file path, so the following downloads an X-13 story from the Internet and interactively displays it in the browser:

```
view(story = system.file("stories", "outlier.Rmd", package="x13story"))
```

If you want to make your story publicly available, you can put it to a [GitHub Gist] (as described above). The `view` function accepts remote URLs, so you can share the corresponding one line R command with your friends.

You can also contribute your story directly to the `x13story` package and the website. To do so, you need to fork the package on GitHub, add your new story and send a pull request to the authors. Your story will be automatically checked for technical validity. After confirmation by one of the authors, it appears on the website. (If that appears too complicated, just send your file to one of the authors. It is greatly appreachiated.)