Capitalized Title Here

true

Abstract

An abstract of less than 150 words.

Discuss, Document, Popularize: An Ideal Workflow

Here, we sketch an ideal workflow to deal with seasonal adjustment problems. As R has become nearly ubiquitous in statistical agencies, and the deployment of X-13 has become a no-brainer, an R based workflow is probably the easiest way to go. It involves three major steps:

1. For discussion, a good starting point is to use a simple R script that includes data, code and comments.
2. For documentation, using markdown and knitr, an R script is easily transfered in a reproducible research document, which can be shared and published as a PDF.
3. Finally, with just a few additional tweaks, we can transform such a document into an interactive lesson, which makes it easy to learn about seasonal adjustment, even if you are not using R.

Discuss: X-13, R and seasonal

As R has become nearly ubiquitous in statistical agencies, R scripts are one of the best ways to talk about any statistical problems. R scripts can include code (of course), but also comments and data.

The last point is underused in practice, but brings enormous benefits in terms of sharing any statistical problem. Using the `dput()` function from R base on any object, it writes a text representation to the console, which can be copy- pasted into your script. For example,

```
dput(AirPassengers)
```

generates the code that represents the `AirPassengers` time series. Copy- pasted, the equivalent would be:

```
myseries <- 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")
```

Putting your data, your code and your comments into the same R script brings enormous benefit in terms of sharing your problem.

1. You can copy-paste your Problem to an email.

2. You can put it to github, or github gist, or any other code sharing platform
3. You can put it on your own website.
4. You can use it in mailing lists or Question/Answer sites such as Stackoverflow or Crossvalidated.

The great thing about having data, code and comments in the same document is that everybody with a working R installation can run it, even without having understood your code.

While this is true for any statistical problem, it is especially true for problems with seasonal adjustment. Thanks to `x13binary` and `seasonal`, installing the X-13 binaries and a powerful R interface is as easy as:

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

which automatically installs the latest build of X-13 on any platform (Linux, Mac or Windows). If you want to make installation of seasonal optional, you can start your R script with the following line:

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

which ensures you have a working version of X-13 and seasonal on your R installation.

Putting your data, your code and your comments into the same R script is crucial, but there are a few more things to be aware of:

1. *Never read from local files*. If you really cannot include it in your R script (e.g., because it is too large), put a `.csv` file on a fixed URL (Github is good for that) and use `read.table` or `read.csv` from R base to directly read from there. These functions can take URLs as an arguments, so you can directly download from there.
2. *Use as few external packages as possible*. Use R base functions, avoid ggplot, dplyr, data.table etc. if possible. Using more packages makes it more difficult to get the same environment on somebody else's computer. 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 be familiar with. If you really need packages (e.g., seasonal), install them conditionally, as shown above.

A single R script containing code, data and comments is the easiest way to share any statistical problem, and it comes especially handy if you are talking about any seasonal adjustment problem.

Document: A reproducible research note

Having kept your discussion in a single R script, it is very easy to transform it into a reproducible research note. RMarkdown is great to use without RStudio. But since it is made by the same people as RStudio, the integration in RStudio is even greater. For other platforms, please refer to the the RMarkdown vignette.

In order to transform your R script into an RMarkdown document, transform your comments to markdown text and put the R code into R chunks.

(All the function names are still constantly changing and may well be wrong)

The *x13tools* package offers several tools to make life easier.

It contains an RMarkdown template, `x13_handout` *(not done yet)*, specifically optimized for time series problems, by using wider figure margins and a unified font for graphs and text (palatino). However, you are not restricted to it, and most functions form the *x13tools* package also work well with the popular `tufte_handout` style, which is included in RStudio.

The *x13tools* package comes with a bunch of functions designed to make the outputs of X-13 and seasonal printer friendly. These functions start with `pretty`:

	Estimate	Std. Error	z value	Pr(>|z|)
Weekday	-0.0029	0.0005	-5.64	0.0000
Easter[1]	0.0178	0.0072	2.48	0.0131
AO1951.May	0.1002	0.0204	4.90	0.0000
MA-Nonseasonal-01	0.1156	0.0859	1.35	0.1781
MA-Seasonal-12	0.4974	0.0775	6.42	0.0000
method: X11 adj. – *ARIMA*: (0 1 1)(0 1 1) – *Obs.*: 144 – *Transform*: log				
nAICc: 947.3 – *BIC*: 963.9 – *QS*: 0 – *Box-Ljung*: 26.65 – *Shapiro*: 0.9908				

Table 1: An example of the prettysummary function

`prettyplot` like `plot`, but with a more Tufte-like look, a lower ink/information ratio and a font that is adjusted to the text font (palatino, by default). `prettyplot` has a method for the `seas` objects and for `ts` objects. If multiple series are supplied, it will also automatically add a legend, which is suited for black and white printing.

`prettymonthplot` like `plot`, but again more Tufte-like. To be implemented.

`prettysummary` like `summary`, but returning a nice Latex table.

`prettysummary2` like `prettysummary`, but can be used for multiple models. Gives a nice model comparison table, but needs some more work (and another name) ....

Here are some examples:

```
library(x13story)
m <- seas(AirPassengers, x11 = "")
x11.seasonal <- series(m, 'x11.seasonal')
```

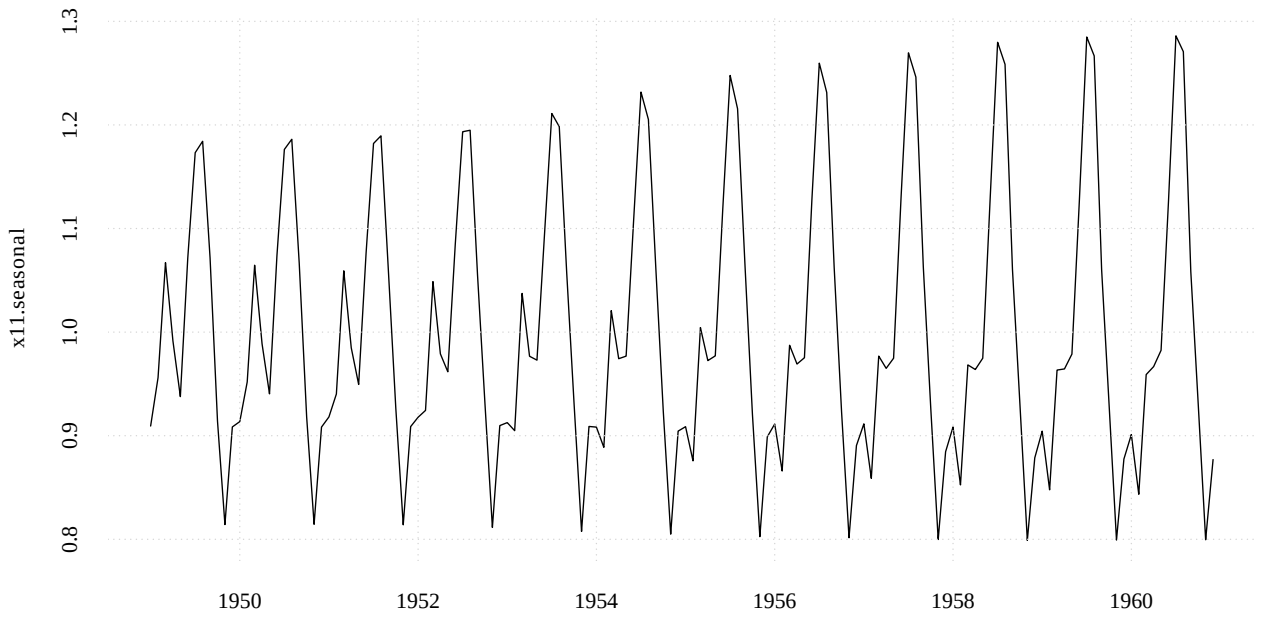

Figure 1: An example of the prettyplot function

`prettyplot` like `plot`, but with a more Tufte-like look, a lower ink/information ratio and a font that is adjusted to the text font (palatino, by default). `prettyplot` has a method for the `seas` objects and for

`ts` objects. If multiple series are supplied, it will also automatically add a legend, which is suited for black and white printing.

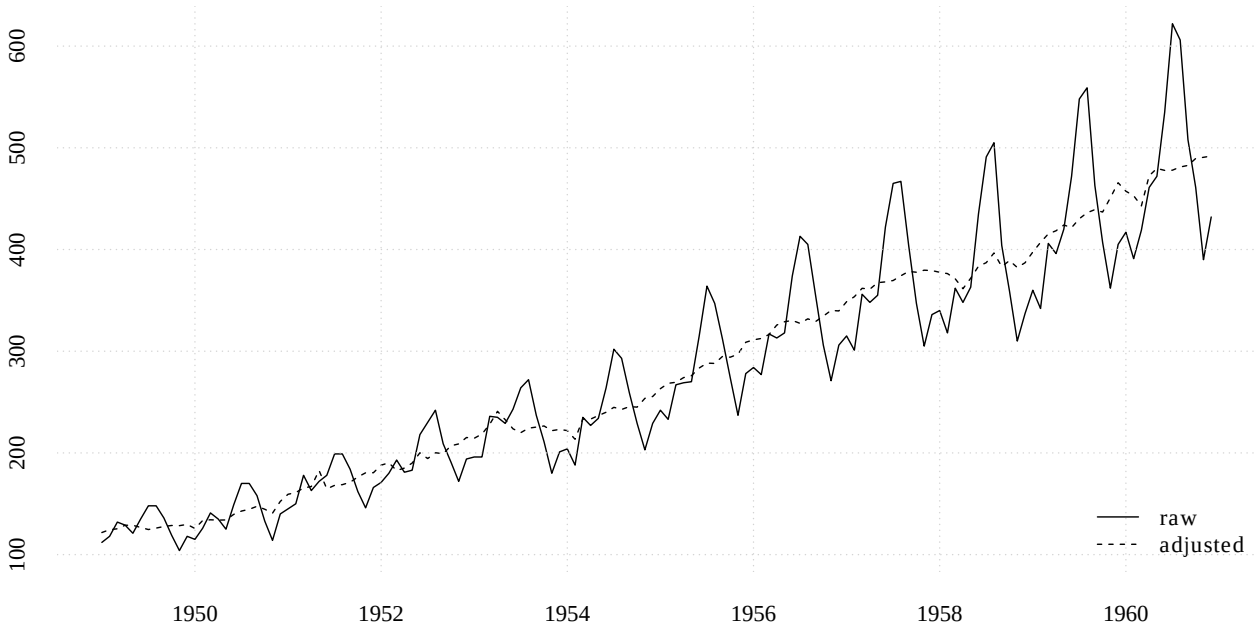

Figure 2: An example of the snapshot function

Popularize: Creating an interactive lesson

Having set up a reproducible research not, it is only a minor step to create an interactive lesson. For this, use the `snapshot` function to generate a snapshot of a seasonal adjustment process. The snapshot model needs a `seas` object as an argument, together with a view.

Both the call of the `seas` object and the view will be shown in the interactive learning tool.

The markdown text that follows after the snapshot function will be included as a description of the lesson in HTML. The R chunk you were using to produce pdf output is ignored. If you want to include code in the HTML description, use non R codes in markdown.