WP.31
ENGLISH ONLY

UNITED NATIONS
ECONOMIC COMMISSION FOR EUROPE

CONFERENCE OF EUROPEAN STATISTICIANS

Work Session on Statistical Data Editing
(Oslo, Norway, 24-26 September 2012)

Topic (v): Software & tools for data editing and imputation

INTERACTIVE ADJUSTMENT AND OUTLIER DETECTION OF TIME DEPENDENT DATA IN R

Submitted by Statistics Austria & TU WIEN [1]

I. THE X12 TOOLS

A. X12-ARIMA for Seasonal Adjustment

1. X12-ARIMA is a widely used seasonal adjustment software for time series analysis and forecast prediction, developed, distributed and maintained by the United States Census Bureau [see, e.g., Time Series Research Staff Statistical Research Division, 2011]. It is the state-of-the-art technology for seasonal adjustment and used in various statistical offices.

2. The X12-ARIMA Seasonal Adjustment program of the U.S. Census Bureau extracts the different components (mainly: seasonal component, trend component, outlier component and irregular component) of a monthly or quarterly time series. The procedure makes additive or multiplicative adjustments and creates an output data set which contains the adjusted time series and intermediate calculations.

B. R Packages x12 and x12GUI

3. The output from X12-ARIMA is somehow static. Users have to extract the required information for further processing. The **x12** R package [Kowarik and Meraner, 2012] provides wrapper functions and an abstraction layer for batch processing X12-ARIMA calculations. It allows to summarize, modify and store the output from X12-ARIMA in a more advanced and user-friendly manner within a well-defined class-oriented implementation. A history of results and parameters is provided, so that it is an easy task for the user to restore or review previous settings.

4. Having the powerful X12-ARIMA Seasonal Adjustment program available directly from within R, as well as containing the new facilities for marking outliers, batch processing and change tracking, make the package a potent and functional tool for daily use in statistical offices.

[1] Prepared by Alexander Kowarik (alexander.kowarik@statistik.gv.at), Angelika Meraner (angelika.meraner@statistik.gv.at), Daniel Schopfhauser (e0925704@student.tuwien.ac.at), Matthias Templ (matthias.templ@gmail.com).

5. On top of the class-oriented (command line) implementation, the graphical user interface (GUI) grants the less experienced R user access to **x12** via the R package **x12GUI** [Schopfhauser et al., 2012]. By simply clicking on a specific time point in a plot window inside the GUI, users can interactively select additive ouliers, level shifts and temporary changes and get feedback immediately.

6. Outlier detection is one issue in seasonal adjustment of time dependent data. Outliers influence the processes even though the corresponding values are true, and it is an obligation to eliminate the effects of outliers, otherwise they diminish the reliability of parameter estimates and results.
The identification of sudden changes in the level of a time series is of high priority and well covered with the **x12** R package. The implemented plot function can depict

Additive Outliers (AO): single and temporary abnormal observations in a time series.
Level Shifts (LS): abrupt and permanent drops or rises in the level of a time series, affecting all observations as of a certain time point.
Temporary Changes (TC): temporary level shifts, returning to their previous level at an exponentially rapid pace.

However, swamping or masking problems make the identification of outliers difficult. To possible overcome these problems, visualisation and interaction with the data is one strength of the R package **x12**. For example, the annual comparison of a selected (possible) outlier can be visualized, illustrating the trend corresponding to a certain month or quarter.

C. Outline for the Rest of the Paper

In Section II, the general structure of X12-ARIMA is discussed while Section III presents the respective R package **x12**. An example can be found in Section IV. The corresponding graphical user interface which is implemented in the **x12GUI** package is described in Section V and Section VI concludes.

II. DATA MANAGEMENT IN X12

7. To run X12-ARIMA, an application dependent input file has to be provided by the user, i.e. the so called specification file with the file extension **.spc** [see e.g. Hood and Monsell, 2010]. It stores the pathname and other general information about the analyzed data, the way it should be processed and the kind of output generated by X12-ARIMA. There are several possible formats to store the inspected data (like *datevalue* format, free format, X-11 format and X12save format).

8. A very basic input specification file could look like Listing 1.

```
series{
  file="airpassengers.dat"
  format="datevalue"
}
```

Listing 1. Specification File for X12-ARIMA

It states that the data should be read from a file called "airpassengers.dat" which contains a dataset in the so called *datevalue* format. Calling X12-ARIMA in the terminal would display the following:

```
C:\work\x12a minimal

 X-12-ARIMA Seasonal Adjustment Program
 Version Number 0.3  Build 188
 PSP = 24
 Execution began Jul 16, 2012  17.06.12
```

```
  Reading input spec file from minimal.spc
  Storing any program output into minimal.out
  Storing any program error messages into minimal.err


 WARNING: At least one visually significant seasonal peak has been
          found in one or more of the estimated spectra.

 Execution complete for minimal.spc

C:\work\
```

LISTING 2. Command line interface for X12-ARIMA

More detail on the possible specs can be found in the X12-ARIMA reference manual [Time Series Research Staff Statistical Research Division, 2011].

9. In every X12-ARIMA run, several files are created including the output, error and log files (see e.g. [Time Series Research Staff Statistical Research Division, 2011] and [Hood and Monsell, 2010]). The actual selection of diagnostic output files can be specified by the user in advance.

Output file (*.out): contains the results of the calculations performed by X12-ARIMA as well as additional diagnostics if that was specified and clarifications about the settings used.

Error file (*.err): contains all the error and warning messages as shown in the terminal during the execution.

Log file (*.log): contains additional diagnostis if requested in the specification file with the savelog-argument.

Diagnostics summary file (*.udg): contains the most important diagnostics and will only be generated if specifically requested in the terminal.

Other files: Depending on the save argument in the specification file, a wide variety of output files can be created containing diagnostics in tabular or tabular-like format.

III. R X12

A. Class Structure of R Package x12

10. The **x12** package for R contains a range of different wrapper functions for the X12-ARIMA binaries [Kowarik and Meraner, 2012]. The R function `X12()` prepares the input data and specification file before calling the X12-ARIMA program. Afterwards it reads the generated output. It consists of classes for different applications which serve as data structure for several X12-ARIMA related input and output tables, as well as methods for manipulating, retrieving, plotting and summarizing parameters or data.

The classes *x12Single* and *x12Batch* form the core of the package. An object of class *x12Single* contains all the information for one time series including the parameters for the X12-ARIMA run. The concatenation of several objects of class *x12Single* is an object of class *x12Batch*, which is intended for batch processing. The most important range of functions are implemented as methods, e.g. `summary()` and `plot()`. If the X12-ARIMA parameters are changed iteratively, all the outputs can be compared. Moreover, it is possible to restore X12-ARIMA parameters from a previous run. To run X12-ARIMA with the data and settings specified in an object of class *x12Single*, *x12Batch* or *ts*, the package provides the `X12()` method. The following short example creates an object of class *x12Single* and performs an `X12()` run with it:

```
s <- new("x12Single", ts=AirPassengers, tsName="air")
s <- X12(s)
```

Hidden from the user, the function `X12()` creates an input specification file and calls X12-ARIMA with this specification. The output is then imported back into R, saved into the corresponding class and returned.

11. As mentioned above, objects of class *x12Single* contain all the information necessary to call X12-ARIMA using the R function `X12()`.

12. Setting, changing and retrieving X12-ARIMA parameters from *x12Single*-, *x12Parameter*- or *x12Batch*-objects can be done by means of the `getP()` and `setP()` methods. In case of setting parameters for *x12Batch* objects, an additional element number vector allows the specification of a respective *x12Single* object:

```
#sObject is of class x12Single
sObject <- setP(sObject, ArgumentList)
ValueList <- getP(sObject, ArgumentList)

#bObject is of class x12Batch
bObject <- setP(bObject, ArgumentList, c(elements))
```

13. Output generated by X12-ARIMA will be stored in objects of class *x12Output*, most slots of which correspond to the tables returned by the program. These tables can be accessed through the `summary()` function and several plot functions or directly from the corresponding object, as can be seen in the following example:

```
#s is an x12Single object
#retrieving the original time series
orgts <- s@x12Output@a1

#retrieving the calculated forecasts
forecast <- s@x12Output@forecast
```

The same applies to the list of important diagnostics results which are stored in the `dg` slot of an *x12Output* object.

B. Plot Methods

14. Beside the access of all parameters and results in an interactive manner, one of the biggest advantages of having X12-ARIMA directly available from within the R environment is the possible usage of plotting functionality. The **x12** package presents the user with its own implementation of the generic plot function for *x12Single* and *x12Output* objects but also with more specialized plot functions in the context of seasonal adjustment.

15. The integrated `plot()` function is implemented in a flexible manner and can be used for a variety of different objectives. As an example, Figure 1(a) shows the original time series as well as the seasonally adjusted series, the trend and forecasts with the respective prediction intervals. Figure 3 on the other hand shows the original time series, the trend and outliers detected by X12-ARIMA. The corresponding help file for the plot function that comes with the package lists all plot parameters. These plot parameters specify, for example, if the trend should be plotted, if transformations should be applied, if outliers should be highlighted and if forecasts or backcasts should be plotted. They also control the appearance of the plot.

16. Function `plotRsdAcf()` allows the plotting of the (partial) autocorrelations of the (squared) residuals which are included in *x12Output* and therefore also in *x12Single* objects. As with most

functions that accept both of these types, the *x12Single* version is merely a wrapper for its consort. This is also valid for the remaining plot functions mentioned below. The most important parameter of the `plotRsdAcf()` function is `which`. It expects one of the characters `"acf"`, `"pacf"` and `"acf2"` standing for "autocorrelation of the residuals", "partial autocorrelation of the residuals" and "autocorrelation of the squared residuals". An example is shown in Figure 1(b) and discussed in the example section IV.

17. The `plotSeasFac()` function graphically represents the seasonal factors which are contained in the *x12Output* slots `d10` and `d8`. The `SI_Ratios` parameter specifies if the SI ratios should be displayed while the rest of the parameters define the appearance of the plot. See Figure 1(d) for an example.

18. The `plotSpec()` function allows the plotting of spectral plots. Spectra concerning this function can be found in the *x12Output* slots `sp0` (original series), `sp1` (differenced seasonally adjusted series), `sp2` (modified irregular series) and `spr` (regARIMA model residuals) and are of class *spectrum*. The parameter `which` defines the type of spectrum to be plotted, i.e. one of the characters `"sa"`, `"original"`, `"irregular"` and `"residuals"` has to be selected. Should the `plotSpec()` function be called for an object of class *spectrum*, an additional `frequency` parameter has to be passed to the method. For an example that is discussed later, see Figure 1(c).

C. Other Methods

19. The **x12** package contains an implementation of the generic summary method for *x12Output*, *x12Single* and *x12Batch* classes. By calling the `summary()` function, a diagnostics summary for X12-ARIMA output is printed which can also be returned in the form of a data frame for further processing. Note that various parameters can be set in the function `summary()`, see Kowarik and Meraner [2012].

20. With the history functionality, parameter settings and output stored in the *x12Parameter* and *x12Output* slots of *x12Single* and *x12Batch* objects can be reverted to a previous state. To this end, settings and output of every `X12()` run are stored in the respective *x12OldParameter* and *x12OldOutput* slots before being modified in a new run. Returning to previous settings is easily accomplished by means of the `prev()` function , i.e. `prev(object, n=NULL)` for *x12Single* objects and `prev(object, index=NULL, n=NULL)` for *x12Batch* objects, where `n` corresponds to the chosen index of a previous run. If old settings and output are no longer required, the `cleanHistory()` function can reset the *x12OldParameter* and *x12OldOutput* slots to the empty default state.

21. Function `saveP(object, file)` serves the purpose of saving the X12-ARIMA parameters stored in the *x12Parameter* slots of *x12Single* and *x12Batch* objects to a file for later use. They can be retrieved using `loadP(object, file)`. An example can be found in the following listing:

```
#Create an object of class x12Single
s <- new("x12Single",ts=AirPassengers)
s <- setP(s,list(arima=c(2,1,1),sarima=c(2,1,1)))

#Save the parameters
saveP(s,file="xyz1.RData")

#Load a saved parameter set to an x12Single object
s <- new("x12Single",ts=AirPassengers)
s <- loadP(s,file="xyz1.RData")
```

22. The `read.spc()` function reads input specification files and returns objects of class *x12Single* or *x12Batch*. Only X12-ARIMA parameters represented by slots of *x12Parameter* objects are supported. This function is still an early beta and will not work in specific situations.

IV. AN ILLUSTRATIVE EXAMPLE

23. The following example briefly illustrates the R **x12** functionality, i.e. an *x12Batch* object is created for which several X12-ARIMA parameters are set and the findings are shown by means of a textual summary and several plot functions. The following listing shows an example of how to work with R **x12**. First the data are loaded and the path for the X12-ARIMA binaries is set. A new *x12Batch* object is generated (consisting of four time series) and different parameters are set for each time series. Subsequently, `X12()` is run on this *x12Batch* object.

```
library(x12)      ## call the package within R
data(AirPassengers)  ## load data
x12path <- ".../x12a.exe"  ## set the path where x12 is installed

## create an x12Batch object with four time series:
xb <- new("x12Batch",list(AirPassengers,AirPassengers,AirPassengers,AirPassengers))

## set the same X12-ARIMA parameters in all four elements:
xb <- setP(xb,list(estimate=TRUE,outlier="all"))

## set a different parameter in each element:
xb <- setP(xb,list(critical=list(LS=3.5,TC=2.5)),1)
xb <- setP(xb,list(arima=c(0,1,1),sarima=c(0,1,1)),2)
xb <- setP(xb,list(arima=c(0,1,1),sarima=c(1,1,1)),3)
xb <- setP(xb,list(arima=c(1,1,1),sarima=c(1,1,1)),4)

## run X12():
xb <- X12(xb)

## show the summary of all four series:
summary(xb)

## extract an x12Single object from the x12Batch object, e.g. the first series:
s <- xb@x12List[[1]]

## show the summary of this series
summary(s)
```

The last line of the previous code listing calls the diagnostics summary for the *x12Single* object `s` taken from the *x12Batch* object `xb`. It summarizes the output and settings, see:

```
--------------------------   Series_1   ------------------------------
------------------------------------------------------------------------

  Model Definition

ARIMA Model: (0 1 1)(0 1 1) (Automatic Model Choice)
Transformation: Automatic selection : Log(y)
Regression Model: Automatically Identified Outliers

  Outlier Detection
Critical |t| for outliers:
aocrit1 aocrit2  lscrit  tccrit
 "3.89"     "*"   "3.5"   "2.5"
Total Number of Outliers: 6
Automatically Identified Outliers: 6
```

```
  Regression Model
                variable    coef stderr    tval
1 autooutlier_TC1951.May   0.078  0.023   3.321
2 autooutlier_TC1951.Jun  -0.099  0.024  -4.174
3 autooutlier_TC1952.Mar  -0.083  0.023  -3.610
4 autooutlier_LS1953.Jun  -0.090  0.023  -3.851
5 autooutlier_TC1954.Feb  -0.075  0.023  -3.243
6 autooutlier_AO1960.Mar  -0.104  0.024  -4.270

  Seasonal Adjustment

Identifiable Seasonality: yes
Seasonal Peaks: rsd
Trading Day Peaks: rsd
Overall Index of Quality of SA
(Acceptance Region from 0 to 1)
Q: 0.2
Number of M statistics outside the limits: 0

SA decomposition: multiplicative
Moving average used to estimate the seasonal factors: 3x3
Moving average used to estimate the final trend-cycle: 9-term Henderson filter
```

The plot functions mentioned in Section III.B. can be used as shown below. The results can be viewed in Figures 1(a), 1(b), 1(c) and 1(d).

```
plot(s, trend=TRUE, sa=TRUE, forecast=TRUE)
plotRsdAcf(s, which="acf2")
plotSpec(s)
plotSeasFac(s)
```

Including outliers in the regression model works as follows:

```
s <- setP(s,list(regvariables=c("AO1950.1","AO1953.5","LS1955.1","lpyear")))
s <- X12(s)
plot(s, trend=TRUE, showAllout=TRUE)
```

The corresponding plot showing all automatically detected and manually selected outliers can be found in Figure 3.

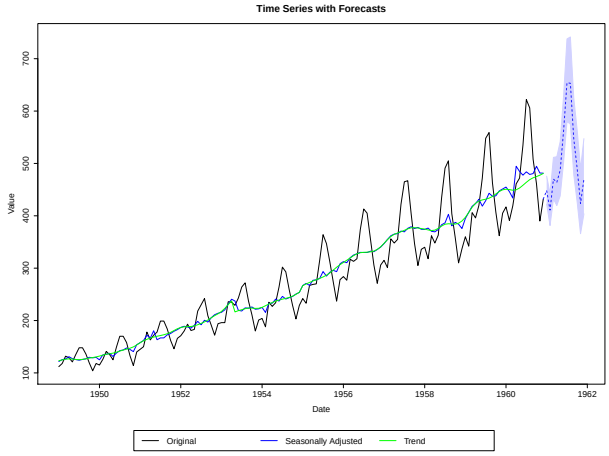
V. THE R X12 POINT AND CLICK GRAPHICAL USER INTERFACE

A. Overview

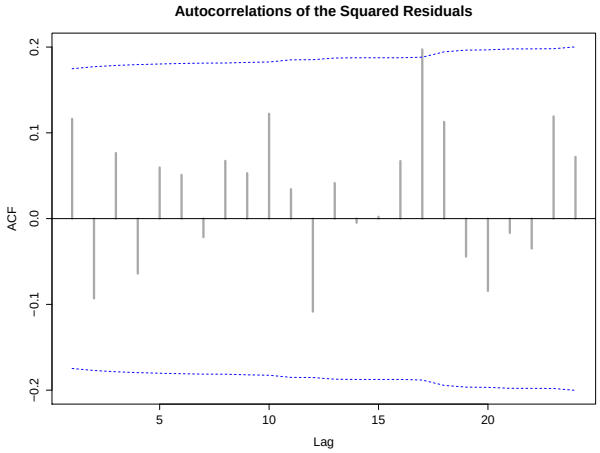
24. The **GTK+** toolkit [Team, 2012] is used for constructing the graphical user interface. It is implemented in R via the **RGtk2** package [Lawrence and Temple Lang, 2012] which supports **GTK+** versions 2.0 and higher (see [Lawrence and Lang, 2010] for more detail).
R graphics are embedded in the GUI with the **cairoDevice** package [Lawrence, 2011], an R wrapper for the vector based, antialiased 2D graphics library **Cairo** (http://www.cairographics.org). **GTK+** and **Cairo** are both licenced under the GNU Lesser General Public License (LGPL).

25. In the previous sections the **x12** R package was handled via the command line interface, making its usage a challenging task for people without prior knowledge of R. In this context, the R package **x12GUI** provides a **G**raphical **U**ser **I**nterface (GUI) which was built with **RGtk2**. The GUI offers the following advantages:

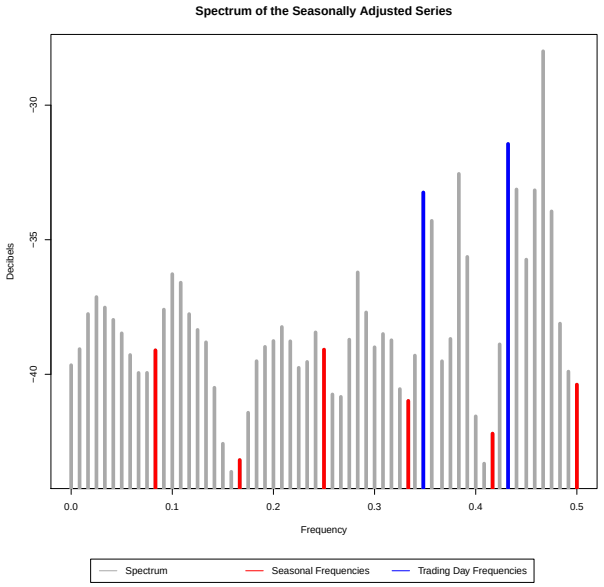
- no prior knowledge of R or the R package **x12** is required
- convenient management of the displayed X12-ARIMA parameters



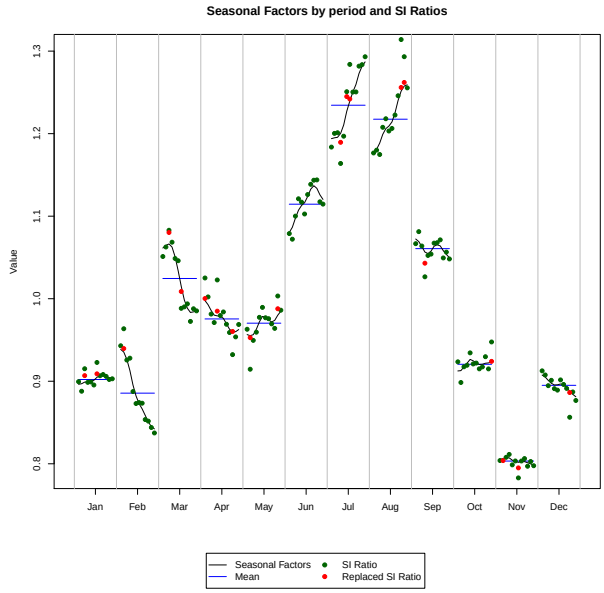
(a) Output of the `plot()` method showing trend and forecasts with prediction intervals as well as the seasonally adjusted series.



(b) Output of the `plotRsdAcf()` function from the R package **x12**, showing the autocorrelations of the squared residuals from the regARIMA model.



(c) Output of the `plotSpec()` function, showing the spectrum of the seasonally adjusted series.



(d) Output of the seasonal factor plot (`plotSeasFac()`).

FIGURE 1. Plots obtained with different plot functions called in the example using the AirPassengers data.

- dynamic editing, i.e. adding and highlighting of outliers from within the plot window
- comparable summaries are provided via point and click
- interactive manipulation of the plots
- on the fly feedback and results

26. Basically the whole **x12** functionality can be controlled with the GUI. After initializing an *x12Single* or an *x12Batch* object, it can be started with:

```
1 library(x12GUI)
```

```
x12path <- ".../x12a.exe"
data(AirPassengersX12Batch)
x12GUI(AirPassengersX12Batch)
```

The interface works with a copy of the original data thus not changing the respective object itself but returning the modified version.

B. User Interaction within the GUI

27. Figure 2 shows the **x12** GUI after startup. It consists of six major areas as well as a menu and a status bar. The left half is considered the input region while the right half displays the output.

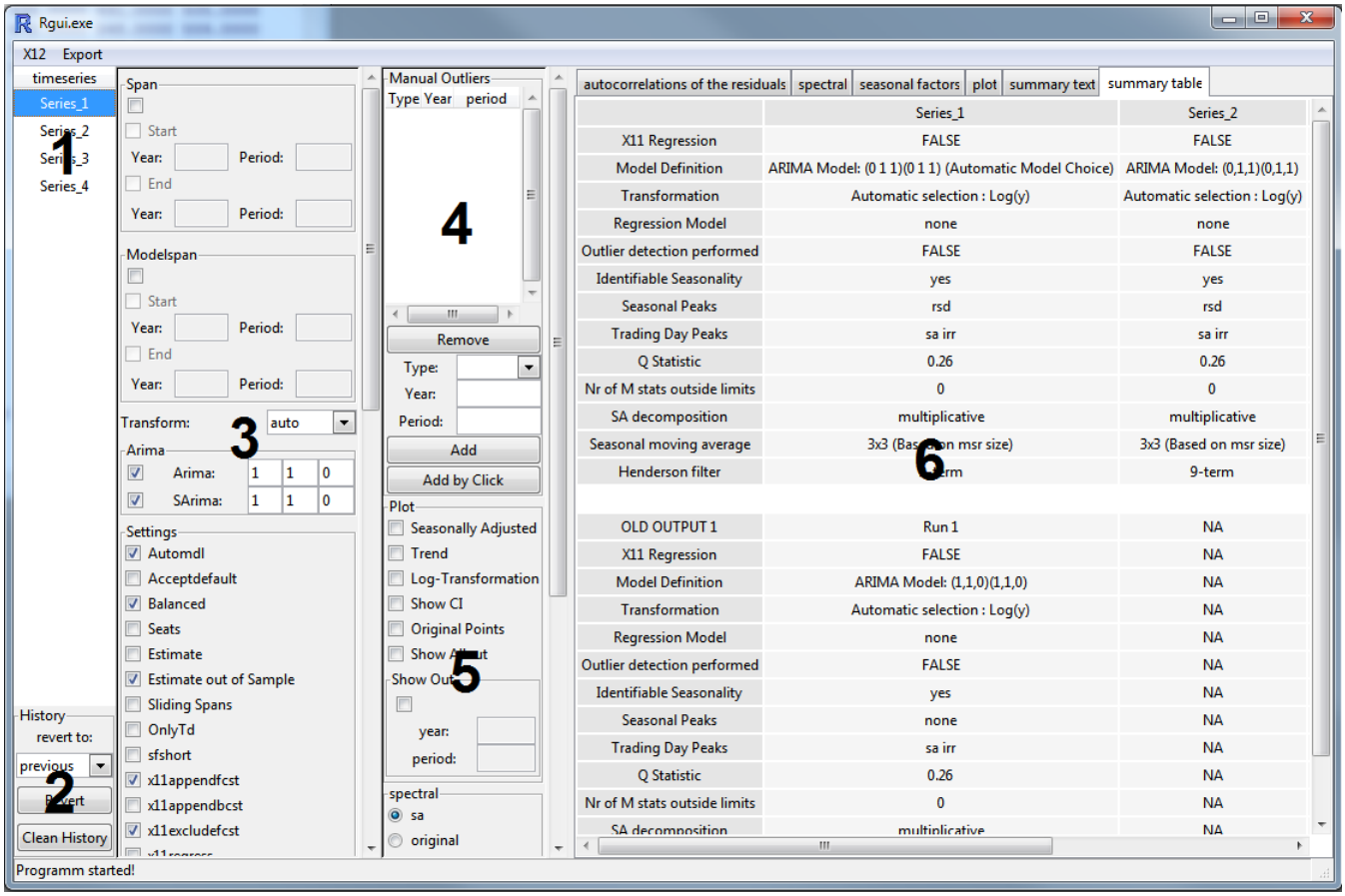


FIGURE 2. Screenshot of the **x12** GUI after startup.

28. A description of the six areas tagged in Figure 2 is given in the following:

(1) Time series selection table: lists the time series belonging to the *x12Single* or *x12Batch* object passed to the **x12** GUI. One or more of them can be selected for parameter manipulation tasks. Deselecting everything is equivalent to selecting everything. Plots can only be viewed for one object at a time as do the textual summaries.

(2) History panel: gives the user access to the history features of the **x12** package like `prev()` and `clearHistory()` (see Section III.C.). The combobox contains all possible runs the settings corresponding to the selected time series can be reverted to.

(3) X12-ARIMA parameter list: contains components for editing the X12-ARIMA parameters. If more than one time series was selected in the time series selection table and some parameter settings differ for these series, said parameters are indicated with an asterix or something similar.

(4) Manual outlier editing: provides the means to manually add and remove outliers from the regression model. Formally, they are values of the `regvariables` argument in the form of "TypeYear.Periode" characters but the GUI distinguishes them from the remaining variables. They can be added "by click" as well, refering to the possibility of flagging an outlier in the plot (see Figure 3). These interactions provide a strong tool for practitioners who have to seasonally adjust their time series.

(5) Plot and summary parameter list: controls the parameter settings of plots and summaries. Changes here have an immediate effect on the output area **(6)**.

(6) Output region: contains all possible output, i.e. the plots mentioned in Section III.B. and the diagnostics summaries introduced in Section III.C. As an additional **x12GUI** feature, outliers can be added to the regression model by right clicking on the respective data point in the time series depicted in the plot window. Each plot can also be saved to a file as **.pdf**.

The menu also includes possibilites to save and load parameters as well as to export objects, plots and summaries.

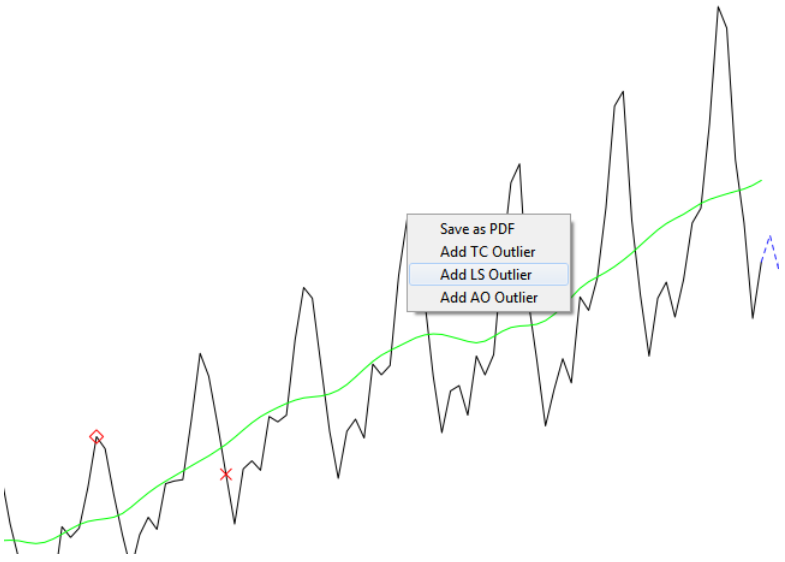


FIGURE 3. Context menu for manually selecting outliers from within the time series plot. This plot is displayed directly in the GUI - here we provide the snapshot of the corresponding area that displays the plot.

29. The seasonal adjustment becomes interactive by the dynamic selection of outliers, the straightforward changing of parameters within the GUI and the possibility to load any results that have been fitted.

VI. CONCLUSION

The analysis of time dependent data plays a substantial role in application fields like business, economics, natural sciences, environmetrics and official statistics. Improving the quality of the findings and reducing the expenditure of time seems very worthwhile.
The X12-ARIMA seasonal adjustment software of the U.S. Census Bureau is profoundly capable in this regard but it comes with the disadvantage of the slightly complicated and inconvenient usage. The input specification files for example demand deep knowledge from the user and the output generated is rather difficult to administer and to modify.
Using the **x12** R package overcomes these difficulties. It offers the possibility to employ R for preprocessing time series data, for managing X12-ARIMA parameters and output and for presenting diagnostics in a more approachable and accessible manner.
In addition to that, the graphical user interface - provided by the **x12GUI** package - enables the manual selection of outliers directly from within the time series plot. Without this GUI, this would normally require several work steps in the R **x12** package and still a lot of more steps using the original software tool X12-ARIMA.
Furthermore, very little prior knowledge of R or the respective packages is required for using the GUI.

References

Catherine C. H. Hood and Brian Monsell. *Getting Started with X-12-ARIMA, Using the Command Prompt on Your PC*, 2010. URL http://www.catherinechhood.net/papers/gsx12input.pdf.

Alexander Kowarik and Angelika Meraner. *x12: X12 - wrapper function and structure for batch processing*, 2012. URL http://CRAN.R-project.org/package=x12. R package version 1.0-3.

Michael Lawrence. *cairoDevice: Cairo-based cross-platform antialiased graphics device driver.*, 2011. URL http://CRAN.R-project.org/package=cairoDevice. R package version 2.19.

Michael Lawrence and Duncan Temple Lang. Rgtk2: A graphical user interface toolkit for R. *Journal of Statistical Software*, 37:52, 2010. URL http://www.jstatsoft.org/v37/i08/paper.

Michael Lawrence and Duncan Temple Lang. Rgtk2: R bindings for gtk 2.8.0 and above, 2012. URL http://CRAN.R-project.org/package=RGtk2. R package version 2.20.24.

Daniel Schopfhauser, Alexander Kowarik, and Angelika Meraner. *x12GUI: X12 - Graphical User Interface*, 2012. URL http://CRAN.R-project.org/package=x12GUI. R package version 0.2-0.

GTK+ Development Team. Gtk+: The gimp toolkit, 2012. URL http://www.gtk.org/.

Time Series Research Staff Statistical Research Division. *X-12-ARIMA Reference Manual*. U.S. Census Bureau, 2011. URL http://www.census.gov/ts/x12a/v03/x12adocV03.pdf.