

Séries Temporais do BACEN – Ajuste Sazonal

Objetivo

- Obter dados diretamente do site do **Sistema Gerenciador de Séries Temporais (SGS)** do Banco Central do Brasil (www.bacen.gov.br).
- Criar/Atualizar planilhas e arquivos com o conteúdo da série
- Utilizar arquivos de configuração como listas: *list*
- Aplicar automaticamente ajustes sazonais na Série

Descrição do Problema

Automatização da obtenção de dados e aplicação de ajustes sazonais na série (X13), através WebService de acesso ao Sistema de Séries Temporais do BACEM.

Package: *seasonal*

seasonal é um package que implementa um interface R para o X13-ARIMA-SEATS do United States Census Bureau.

Principais funções:

seas(<TimeSeries>...*list*=<Especificações>): executa o ajuste sazonal

final (<Seasonal TimeSeries>) : extrai a série ajustada

predict(<Seasonal TimeSeries>): extrai a série ajustada

ortiginal(...): Série original

plot(<Seasonal TimeSeries>): gráfico com a série ajustada

view(<Seasonal TimeSeries>): interface gráfica para o ajuste e definição de parâmetros

Packages e Sources Auxiliares

Específicas da Disciplina:

source("../lib/BACEN-TSMS-WebService.r"): Acesso ao webservice do BACEN

```
TSMSGetSeries = function(codes, startDate = as.Date("2016-01-01"), endDate = Sys.Date(),
  output = c("data.frame", "list", "xml"),
  parseDates = TRUE,
  url = "https://www3.bcb.gov.br/wssgs/services/FachadaWSSGS?method=getValoresSeriesXML",
  ssl.verifypeer = FALSE, config=list()) {
```

Packages padrão do R:

library(XLConnect): interface com o Excel: leitura e escrita

library(lubridate) : funções para trabalhar com datas

library(seasonalview): : interface gráfica do *seasonal*

R: Time-Series Objects

Objeto básico do R:

```
ts(data = NA, start = 1, end = numeric(), frequency = 1,
    deltat = 1, ts.eps = getOption("ts.eps"), class = , names = )
as.ts(x, ...)
is.ts(x)
```

Arguments

- data** a vector or matrix of the observed time-series values. A data frame will be coerced to a numeric matrix via `data.matrix`. (See also 'Details'.)
- start** the time of the first observation. Either a single number or a vector of two integers, which specify a natural time unit and a (1-based) number of samples into the time unit. See the examples for the use of the second form.
- end** the time of the last observation, specified in the same way as `start`.
- frequency** the number of observations per unit of time.
- deltat** the fraction of the sampling period between successive observations; e.g., 1/12 for monthly data. Only one of `frequency` or `deltat` should be provided.
- ts.eps** time series comparison tolerance. Frequencies are considered equal if their absolute difference is less than `ts.eps`.
- class** class to be given to the result, or none if `NULL` or `"none"`. The default is `"ts"` for a single series, `c("mts", "ts", "matrix")` for multiple series.
- names** a character vector of names for the series in a multiple series: defaults to the colnames of `data`, or `Series 1, Series 2, ...`.
- x** an arbitrary R object.
- ...** arguments passed to methods (unused for the default method).

Exemplo:

```
data.ts = ts(data, frequency = 12, start = c(2012,1));
```

Séries Temporais do BACEN

Exemplo de Script

```
source("../lib/BACEN-TSMS-WebService.r");
library(XLConnect);
library(seasonal);
library(lubridate);
library(seasonalview);

## Configuração de Proxy
config = use_proxy(url=" ", port = 80, username = " ", password = " " auth = "ntlm");

## Códigos de séries de inflação
# codes = c(188,189,190,433);

codes = 433;

## Baixa as séries
data = TSMSGetSeries(codes, config = config, startDate="01/01/2005");

data.ts = ts(data, frequency = 12, start = c(2012,1));

startDate = min(data[["DATA"]]);
data.ts = ts(data[["VALOR"]], frequency = 12, start = c(year(startDate),month(startDate)));

data.seas = seas(data.ts);
data.seas = seas(data.ts, arima.model = "(0 1 1)(0 1 1)");

view(data.seas)

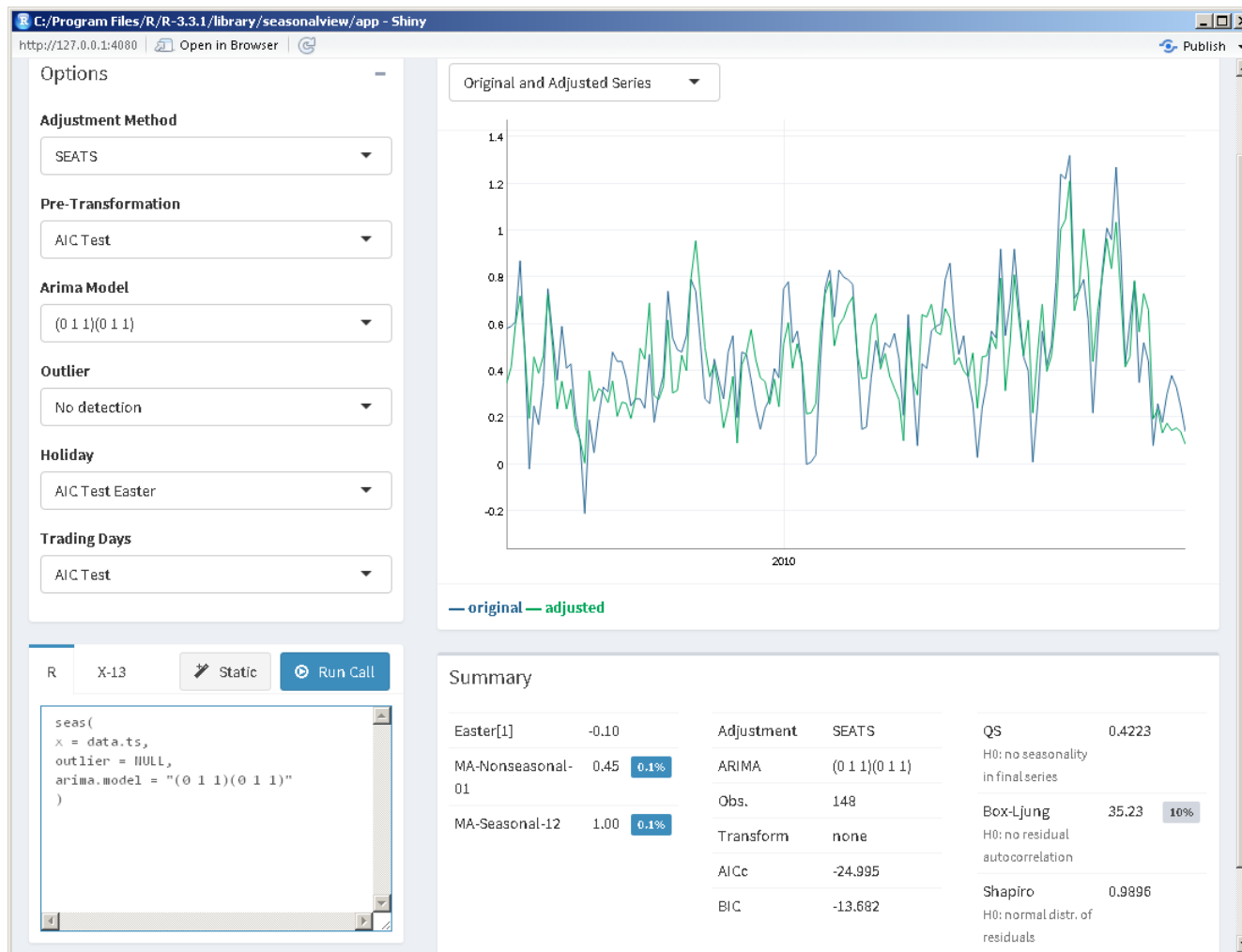
## Especificando
spec = list("arima.model" = "(0 1 1)(0 1 1)", "outlier"=NULL, "regression.aictest" = "td");
data.seas = seas(data.ts, list = spec);
print(final(m));
print(original(m));

## NA handling
# data.seas = seas(data.ts, na.action = na.omit)      # no NA in final series
# data.seas = seas(data.ts, na.action = na.exclude) # NA in final series
# data.seas = seas(data.ts, na.action = na.fail)    # fails

## Funções de extração de dados
final(m)
predict(m)
original(m)
```

Package: *seasonalview* – Interface Gráfica

Interface Gráfica: *view(<ts>)* `view(data.seas)`



Séries Temporais do BACEN

Exercício

Desenvolver um script em R leia um arquivo de configuração com uma lista de séries para serem baixadas, aplicado o ajuste sazonal, e a séries original e a série ajustada grava em uma planilha.

Objetivo: Baixar as séries do Banco Central e atualizar planilhas Excel com os dados

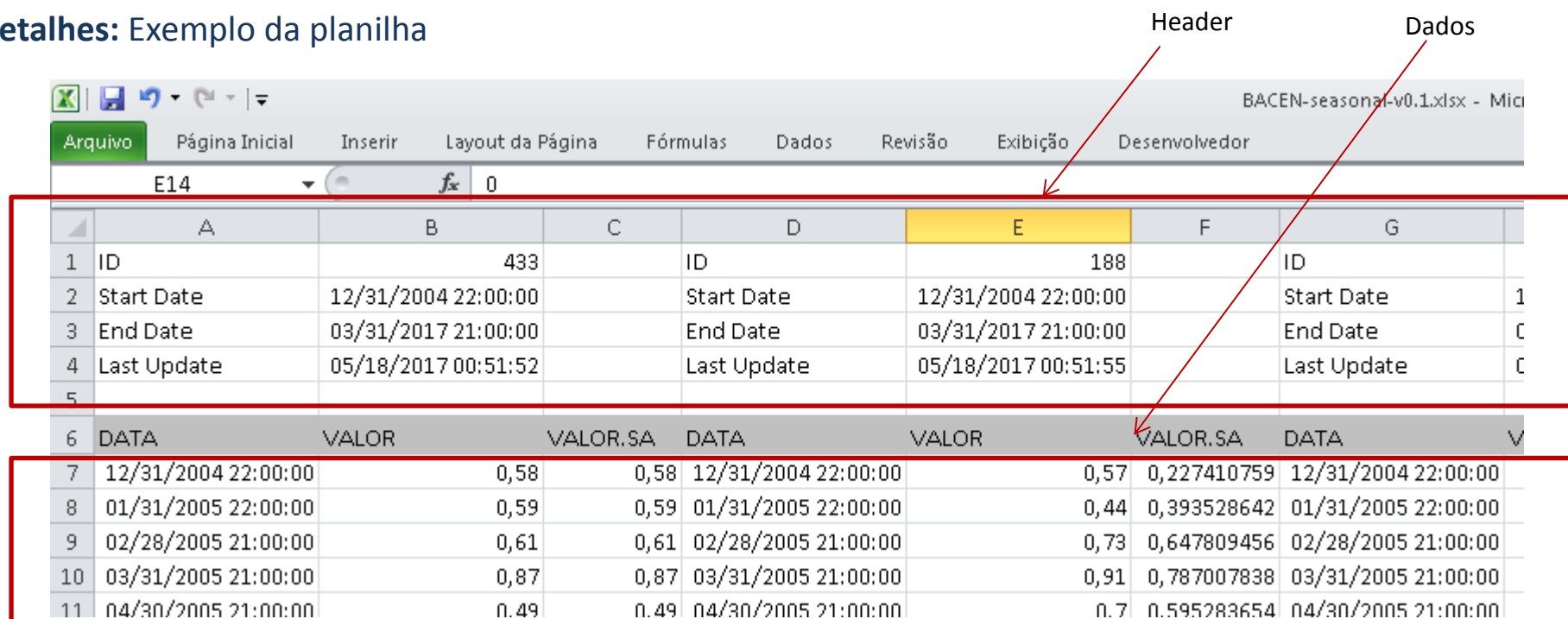
Parâmetros:

`workbook`: path do arquivo Excel que será atualizado com os dados baixados

`url` (opcional) = <https://www3.bcb.gov.br/sgspub/JSP/sgsgeral/FachadaWSSGS.wsd>

Retorno: Número de séries atualizadas

Detalhes: Exemplo da planilha



BACEN-seasonal-v0.1.xlsx - Microsoft Excel							
Arquivo Página Inicial Inserir Layout da Página Fórmulas Dados Revisão Exibição Desenvolvedor							
E14 fx 0							
	A	B	C	D	E	F	G
1	ID	433		ID	188		ID
2	Start Date	12/31/2004 22:00:00		Start Date	12/31/2004 22:00:00		Start Date
3	End Date	03/31/2017 21:00:00		End Date	03/31/2017 21:00:00		End Date
4	Last Update	05/18/2017 00:51:52		Last Update	05/18/2017 00:51:55		Last Update
5							
6	DATA	VALOR	VALOR.SA	DATA	VALOR	VALOR.SA	DATA
7	12/31/2004 22:00:00	0,58	0,58	12/31/2004 22:00:00	0,57	0,227410759	12/31/2004 22:00:00
8	01/31/2005 22:00:00	0,59	0,59	01/31/2005 22:00:00	0,44	0,393528642	01/31/2005 22:00:00
9	02/28/2005 21:00:00	0,61	0,61	02/28/2005 21:00:00	0,73	0,647809456	02/28/2005 21:00:00
10	03/31/2005 21:00:00	0,87	0,87	03/31/2005 21:00:00	0,91	0,787007838	03/31/2005 21:00:00
11	04/30/2005 21:00:00	0,49	0,49	04/30/2005 21:00:00	0,7	0,595283654	04/30/2005 21:00:00