Usando o SGS da SPE através do R usando a biblioteca APIA

Rogério Porto

18 de maio de 2015

1 Introdução

Apresentamos um breve tutorial de como usar o SGS da SPE através do R para consultar e analisar as séries que estão no SGS. Não pretendemos inserir, excluir nem alterar dados do SGS através do R. Além disso, escrevemos este tutorial com o objetivo de documentar e testar as funções que existem, enquanto esse conjunto de funções não se transforma em um pacote. Ou seja, este tutorial deverá sofrer alterações futuras.

Desse modo, se os exemplos deste texto não funcionarem exatamente como apresentados, então há defasagem entre a biblioteca de funções, a versão do tutorial, a base do SGS ou combinações dessas.

Esperamos que o leitor tenha um conhecimento mínimo do ambiente R e disposição para usar uma interface do tipo linha de comando (que pode ser programável, como o Stata) em detrimento de interfaces do tipo *point-and-click*, como o Eviews.

2 Acessando o SGS

Para ter acesso ao SGS através do R, desenvolvemos uma biblioteca de funções chamada APIA (pronuncia-se "ápia"). Antes de usar a APIA, precisamos ter o arquivo `apia.R` e o diretório `apia` no endereço de trabalho que, em geral, deve ser a pasta `Documentos`, que fica em `C:\Users\numerodoCPF\Documents`.

Com isso, inicie uma sessão do R (clicando no ícone do R que está no *desktop* do computador). Iremos precisar do pacote `RMySQL` que pode ser ins-

talado digitando `install.packages("RMySQL")` e escolhendo o repositório mais próximo.

Agora basta digitar

```
> source("apia.R")
```

e já temos acesso a funções que acessam os dados e metadados do SGS.

Vamos começar vendo quais séries estão previstas para serem divulgadas em um determinado dia, por exemplo no dia 18, digitando

```
> agendasgs(18)
```

```
    defasagem fonte       codigo dia
33          1  ACSP          SPC  18
225         1  ACSP       SPC_sa  18
34          1  ACSP    USECHEQUE  18
226         1  ACSP USECHEQUE_sa  18
101         1   CNC     Duraveis  18
```

ou

```
> agendasgs(dias = 18)
```

```
    defasagem fonte       codigo dia
33          1  ACSP          SPC  18
225         1  ACSP       SPC_sa  18
34          1  ACSP    USECHEQUE  18
226         1  ACSP USECHEQUE_sa  18
101         1   CNC     Duraveis  18
```

Podem aparecer algumas mensagens de aviso que devem simplesmente ser ignoradas. Para essas mensagens não aparecerem, podemos armazenar o resultado da agenda em um objeto `a`, por exemplo,

```
> a <- agendasgs(18)
```

e imprimindo o objeto `a` na tela

```
> a
```

```
    defasagem fonte       codigo dia
33          1  ACSP          SPC  18
225         1  ACSP       SPC_sa  18
34          1  ACSP    USECHEQUE  18
226         1  ACSP USECHEQUE_sa  18
101         1   CNC     Duraveis  18
```

Para saber a agenda dos dias 18 e 20, digite

```
> a <- agendasgs(c(18, 20))
> a

    defasagem   fonte       codigo dia
33          1    ACSP          SPC  18
225         1    ACSP       SPC_sa  18
34          1    ACSP    USECHEQUE  18
226         1    ACSP USECHEQUE_sa  18
101         1     CNC     Duraveis  18
281         2   BACEN      Cons_PF  20
282         2   BACEN   ConsLiv_PF  20
286         2   BACEN   Juros_Veic  20
284         2 Tesouro   Ben_Assist  20
285         2 Tesouro     Ben_Prev  20
283         2 Tesouro   Seg_Desemp  20
```

Para saber a agenda do dia 16 até o dia 18, digite

```
> a <- agendasgs(16:18)
> a

    defasagem    fonte       codigo dia
553         0    BACEN     SelicMes  16
33          1     ACSP          SPC  18
225         1     ACSP       SPC_sa  18
34          1     ACSP    USECHEQUE  18
226         1     ACSP USECHEQUE_sa  18
101         1      CNC     Duraveis  18
118         2 IBGE PMC     PMC_A_sa  16
117         2 IBGE PMC       PMC_sa  16
```

Note que o resultado aparece ordenado por defasagem, fonte, código e dia. Uma vez que o resultado está armazenado em um objeto, podem ser feitas manipulações usuais como, por exemplo, selecionar apenas uma das fontes que possuem séries previstas para serem divulgadas nesses dias

```
> subset(a, fonte == "ACSP")
```

```
    defasagem fonte       codigo dia
33          1  ACSP          SPC  18
225         1  ACSP       SPC_sa  18
34          1  ACSP    USECHEQUE  18
226         1  ACSP USECHEQUE_sa  18
```

Uma outra função relacionada à anterior é a que seleciona todas as séries que deveriam estar atualizadas segundo o calendário do SGS. Para isso digite o comando seguinte que armazena dados dessas séris no objeto `b`:

```
> b <- atualizarsgs()
```

Em seguida, podemos usar as funções usuais do R para, por exemplo, criar uma tabela com a quantidade de séries a atualizar segundo o código do responsável e a fonte, conforme a seguir,

```
> table(b$fonte, b$responsavel)
```

```
                  2 16 17
  CNI             0  2  0
  HSBC            0  1  0
  IBGE PIM        5  0 14
  IBRE FGV        0  7  0
  IBRE FGV        0  1  0
  ONS             0  3  0
  Sec. Energia-SP 0  1  0
```

ou simplesmente a lista dos códigos das séries a atualizar em ordem alfabética:

```
> sort(b[, "codigo"])
```

```
 [1] "BCD_PIM"    "BCD_PIM_sa" "BCN_PIM"    "BCN_PIM_sa" "BCT_PIM"
 [6] "BCT_PIM_sa" "BI_PIM"     "BI_PIM_sa"  "BK_PIM"     "BK_PIM_sa"
[11] "EnergSP"    "ICC_PIM"    "ICC_PIM_10" "ICI_pl"     "ICI_sa_pl"
[16] "IE_pl"      "IE_sa_pl"   "Ind_Ext"    "Ind_Ext_sa" "IndTran"
[21] "IndTran_sa" "ISA_pl"     "ISA_sa_pl"  "Nuci_pl"    "Nuci_sa_pl"
[26] "ONS_hid"    "ONS_Nuc"    "ONS_term"   "PIM"        "PIM_10"
[31] "PIM_sa"     "PMI_M_sa"   "UCI"        "UCI_sa"
```

Podemos também consultar os metadados de uma série específica no SGS com

```
> b <- metasgs("SPC")
> b[c("idserie", "nome", "frequencia")]

$idserie
[1] 39

$nome
[1] "Consultas de crÃ©dito para vendas no comercio varejista ACSP"

$frequencia
[1] "m"
```

ou, simplesmente,

```
> metasgs("SPC")

$idserie
[1] 39

$codigo
[1] "SPC"

$nome
[1] "Consultas de crÃ©dito para vendas no comercio varejista ACSP"

$descricao
[1] "Consultas de credito para vendas no comercio varejista da ACSP; entra no site
```

```
$frequencia
[1] "m"

$idgrupo
[1] 3

$idunidade
[1] 2

$seriemae
[1] ""

$ajustada
[1] "S"

$ajustesazonal
[1] "IBC-Br 01/12/2000"

$bacen
[1] "1453"

$bloomberg
[1] ""

$fonte
[1] "ACSP"

$formula
[1] ""

$dia
[1] "18"

$datainicio
[1] "2001-01-01"

$datafim
[1] NA
```

```
$atualizada
[1] "2001-01-01"

$ultimovalor
[1] 10

$referenciavalor
[1] 2e+06

$tolerancia
[1] 7e+05

$responsavel
[1] 17

$defasagem
[1] 1
```

Para consultar os metadados de várias séries de uma só vez basta digitar, por exemplo,

```
> b <- metasgs(codigo = c("SPC", "SPC_sa"))
> b[c("idserie", "nome", "frequencia")]

$idserie
[1]  39 239

$nome
[1] "Consultas de crÃ©dito para vendas no comercio varejista ACSP"
[2] "Consultas de crÃ©dito p/ vendas no comercio varejista _sa "

$frequencia
[1] "m" "m"
```

Como existem muitas séries no SGS, podemos armazenar todos os metadados em um objeto para consultarmos sempre que necessário e fazermos as manipulações que desejarmos:

```
> b <- metasgs()
```

Também podemos consultar todas as unidades de medida cadastradas no SGS digitando `unidadesgs()` e todos os usuários cadastrados digitando `usuariosgs()`. Nestes dois casos as funções não têm parâmetros, isto é, não deve ter nada entre os parênteses.

3 Acessando dados do SGS

Até agora acessamos apenas metadados do SGS. No entanto, para começarmos fazendo análises simples, precisamos acessar os dados do SGS.

Por exemplo, vamos acessar e carregar para a memória do R duas séries e armazená-las em um objeto chamado `x` com

```
> x1 <- sgs(freq = "t", codigos = c("PIB", "PIB_sa"))
> head(x1)
```

```
        data       PIB   PIB_sa
1 1995-03-01  97.71036       NA
2 1995-06-01 101.19811       NA
3 1995-09-01 108.71250       NA
4 1995-12-01 105.22439       NA
5 1996-03-01  96.71861 100.0209
6 1996-06-01 100.18704 100.3977
```

Se nada for informado em `codigos`, então todas as séries trimestrais (`freq = "t"`) serão armazenados no objeto `x`, o que pode demorar muito e tomar muito espaço na memória, principalmente com as séries mensais. Portanto, use essa função com cautela! Assim, por exemplo, podemos carregar na memória duas séries mensais:

```
> x2 <- sgs(freq = "m", codigos = c("SPC", "SPC_sa"))
> x2[721:749,]
```

```
          data     SPC  SPC_sa
721 2013-01-01 1998315 2048812
722 2013-02-01 1654399 2033735
723 2013-03-01 2026770 2058834
724 2013-04-01 1981335 2086805
```

```
725 2013-05-01 1986934 2001765
726 2013-06-01 1923743 2080297
727 2013-07-01 2031026 2057920
728 2013-08-01 2121497 2106654
729 2013-09-01 2008862 2052662
730 2013-10-01 2289600 2093775
731 2013-11-01 2175726 2093384
732 2013-12-01 2641710 2099550
733 2014-01-01 2048327 2106051
734 2014-02-01 1802043 2112663
735 2014-03-01 2041653 2090129
736 2014-04-01 1883237 2086138
737 2014-05-01 2116943 2129460
738 2014-06-01 1951507 2080476
739 2014-07-01 1957057 2013368
740 2014-08-01 2056062 2094839
741 2014-09-01 2133303 2122774
742 2014-10-01 2395967 2166614
743 2014-11-01 2153628 2108157
744 2014-12-01 2616096 2052744
745 2015-01-01 1968756 2041809
746 2015-02-01 1653168 2035807
747 2015-03-01 2115052      NA
748 2015-04-01 1858320      NA
749 2015-05-01 2010717      NA
```

Como existem muitas séries mensais, isso pode levar um bom tempo. Assim, é melhor carregar de uma só vez todas as séries que pretendemos analisar.

4 Realizando cálculos simples

ATENÇÃO. As funções a seguir realizam cálculos em vetores numéricos gerais, sem preocupação com o significado prático dos resultados. Ou seja, se o vetor contiver uma série temporal, iremos supor que a série já está "arrumada"com início e fim corretos para os cálculos e, se necessário, com o devido ajuste sazonal. O ônus de um eventual uso incorreto cabe apenas ao usuário!

As funções `tt1(x)`, `tt4(x)` e `tt12(x)` calculam as variações percentuais na margem, interanual trimestral e interanual mensal a partir de uma série `x`. Ou seja, se o vetor `x` contiver a série $x = \{x_1, x_2, \ldots, x_n\}$, então, digitando `tt1(x)` aparecerá na tela o resultado da variação $t/(t-1)$ da série x, isto é, os valores

$$\left\{100\frac{x_2}{x_1}, 100\frac{x_3}{x_2}, \ldots, 100\frac{x_n}{x_{n-1}}.\right\}$$

Por exemplo, a série da variação na margem de $x = (1, 2, \ldots, 10)$ pode ser obtida digitando

```
> x <- 1:10
> tt1(x)

 [1]        NA 100.00000  50.00000  33.33333  25.00000  20.00000  16.66667
 [8]  14.28571  12.50000  11.11111
```

Note que o primeiro elemento do vetor que é mostrado na tela é `NA`, indicando que não é possível calcular a variação na margem para o primeiro elemento da série pois não existe um valor anterior ao primeiro.

Similarmente, se o vetor `x` contiver a série $x = \{x_1, x_2, \ldots, x_n\}$, então, digitando `tt4(x)` e `tt12(x)` irão mostrar na tela o resultado da variação $t/(t-4)$ e $t/(t-12)$ da série x, respectivamente. Voltando ao exemplo, temos

```
> x <- 1:20
> tt4(x)

 [1]        NA        NA        NA        NA 400.00000 200.00000 133.33333
 [8] 100.00000  80.00000  66.66667  57.14286  50.00000  44.44444  40.00000
[15]  36.36364  33.33333  30.76923  28.57143  26.66667  25.00000

> tt12(x)

 [1]        NA        NA        NA        NA        NA        NA        NA
 [8]        NA        NA        NA        NA        NA 1200.0000  600.0000
[15]  400.0000  300.0000  240.0000  200.0000  171.4286  150.0000
```

Na prática usual, a função `tt1(x)` é usada com séries mensais ou trimestrais, sempre com ajuste sazonal. Por sua vez, as funções `tt4(x)` e `tt12(x)`

são usualmente aplicadas a séries trimestrais e mensais, respectivamente, sempre sem ajuste sazonal.

Por exemplo, a variação na margem de uma série trimestral sazonalmente ajustada pode ser obtida com

```
> calc1 <- tt1(x1$PIB_sa)
> calc1
```

```
 [1]           NA           NA           NA           NA           NA
 [6]  0.376711184  4.372908735 -1.326018325  0.412218523  0.996044492
[11]  1.711608340  0.512020921 -2.646556916  2.373497325  0.317474245
[16] -1.353428325 -0.210220352  0.436110522  0.705592295  1.239110578
[21]  1.412563276  1.026105935  0.914295383  1.085432366  0.257622162
[26] -0.146530811 -0.811080729  0.128040626  1.817901807  0.612694976
[31]  1.567080878  1.205498562 -1.291159504 -0.001685626  0.842938692
[36]  1.223349447  1.791262196  1.960660965  1.383997476  0.942839794
[41]  0.015379721  1.515155237 -0.269301405  0.955245907  1.246427198
[46]  1.025582555  1.339503792  1.116058463  1.818529890  1.789623108
[51]  1.026892646  1.665289376  1.707968297  1.554787609  1.800514606
[56] -4.093953615 -2.158749066  2.757445222  2.394129601  2.218460044
[61]  1.865996899  1.521582031  1.126020422  1.171645236  0.833721523
[66]  1.879206943 -0.511121189  0.331493916  0.236188291  0.439981091
[71]  1.253530216  0.382324920  0.757314377  1.172261004  0.103926766
[76]  0.035538211  0.713094345 -1.409212305  0.156808922  0.268806947
[81] -0.160605255
```

Já a variação interanual mensal de uma série mensal pode ser obtida com

```
> calc2 <- tt12(x2[721:749, "SPC"])
> calc2
```

```
 [1]         NA         NA         NA         NA         NA         NA
 [7]         NA         NA         NA         NA         NA         NA
[13]  2.5027085  8.9243284  0.7343211 -4.9511062  6.5431967  1.4432281
[19] -3.6419524 -3.0843786  6.1946017  4.6456586 -1.0156610 -0.9695992
[25] -3.8846825 -8.2614566  3.5950771 -1.3230942 -5.0178961
```

Note que essas funções permitem fazer coisas não usuais como, por exemplo, calcular a variação na margem de uma série sem ajuste sazonal ou a variação interanual trimestral de uma série mensal (isto é, a variação percentual entre valores de uma série separados por 4 marcações de tempo):

```
> calc3 <- tt4(x2[721:749, "SPC"])
> calc3
```

```
 [1]          NA          NA          NA          NA  -0.5695298  16.2804741
 [7]   0.2099893   7.0741192   1.1036099  19.0179769   7.1244780  24.5210340
[13]   1.9645451 -21.2944182  -6.1622190 -28.7114407   3.3498558   8.2941417
[19]  -4.1435053   9.1770181   0.7728125  22.7752194  10.0442143  27.2381864
[25]  -7.7132503 -31.0020547  -1.7912100 -28.9659095   2.1313459
```

Na realidade, essas funções são casos particulares da função `ttk(x, k)` que calcula a variação $t/(t-k)$ de uma série x qualquer, onde se espera que k seja um número inteiro positivo. Por exemplo, podemos obter calcular a variação $t/(t-5)$ de x com

```
> ttk(x, 5)
```

```
 [1]        NA        NA        NA        NA        NA 500.00000 250.00000
 [8] 166.66667 125.00000 100.00000  83.33333  71.42857  62.50000  55.55556
[15]  50.00000  45.45455  41.66667  38.46154  35.71429  33.33333
```

Outros cálculos simples que são muito utilizados na prática são as variações acumuladas em quatro trimestres e em 12 meses. Por exemplo a variação percentual acumulada em quatro trimetres do PIB pode ser obtida com

```
> ac4t(x1$PIB)
```

```
 [1]          NA          NA          NA          NA          NA          NA
 [7]          NA -0.98975097  0.08001757  1.48568695  2.20356949  3.38925759
[13]  2.77137615  1.98402692  1.62848223  0.35528329  0.31719273 -0.15331132
[19] -0.40745946  0.48940948  1.36229747  2.45966122  3.78971549  4.38214328
[25]  4.10757492  3.64073112  2.56054637  1.27600568  0.58197952  0.61685475
[31]  1.60091349  3.07621263  3.63144153  3.26277915  2.36571007  1.22350700
[37]  1.49306402  2.81666071  4.27755632  5.65978387  5.71469844  5.25667215
[43]  4.13605284  3.14916007  3.21359345  2.70156519  3.32974829  4.00028844
[49]  4.22140458  5.26219685  5.57557970  6.00596243  6.22392962  6.17381417
[55]  6.45534282  5.01799427  2.87415010  0.71675176 -1.31013022 -0.23435633
[61]  2.55225514  5.29848693  7.47088696  7.57063901  6.60242251  5.63422113
[67]  4.74033594  3.91547364  3.03516576  2.08569262  1.80819175  1.76354616
[73]  2.00151112  2.77745984  2.79548482  2.74210458  2.77704890  1.48029976
[79]  0.72781037  0.14640886 -0.88822673
```

e a variação percentual acumulada em 12 meses da série do SPC com

```
> ac12m(x2[721:749, "SPC"])
```

```
 [1]          NA          NA          NA          NA          NA          NA
 [7]          NA          NA          NA          NA          NA          NA
[13]          NA          NA          NA          NA          NA          NA
[19]          NA          NA          NA          NA          NA   1.2717675
[25]   0.7485879 -0.4401225 -0.2062872  0.0861613 -0.8560471
```

Também estes são casos particulares de variações acumuladas em k períodos, onde se espera que k seja um número inteiro positivo e que podem ser obtidas com a fução `ackt(x, k)`. Por exemplo, podemos fazer cálculos pouco usuais como

```
> ackt(x, 2)
```

```
 [1]        NA        NA        NA 133.33333  80.00000  57.14286  44.44444
 [8]  36.36364  30.76923  26.66667  23.52941  21.05263  19.04762  17.39130
[15]  16.00000  14.81481  13.79310  12.90323  12.12121  11.42857
```

ou verificar curiosidades, como a de que a variação na margem é igual à variação acumulada em um período:

```
> cbind(tt1(x), ackt(x, 1))
```

```
            [,1]       [,2]
 [1,]         NA         NA
 [2,] 100.000000 100.000000
 [3,]  50.000000  50.000000
 [4,]  33.333333  33.333333
 [5,]  25.000000  25.000000
 [6,]  20.000000  20.000000
 [7,]  16.666667  16.666667
 [8,]  14.285714  14.285714
 [9,]  12.500000  12.500000
[10,]  11.111111  11.111111
[11,]  10.000000  10.000000
[12,]   9.090909   9.090909
[13,]   8.333333   8.333333
```

```
[14,]   7.692308   7.692308
[15,]   7.142857   7.142857
[16,]   6.666667   6.666667
[17,]   6.250000   6.250000
[18,]   5.882353   5.882353
[19,]   5.555556   5.555556
[20,]   5.263158   5.263158
```

Antes de realizar esses cálculos, pode ser necessário fazer transformações nas séries como, por exemplo, diminuir a frequência de mensal para trimestral ou anual ou de trimestral para anual, que pode ser feito com as funções `trimestralizam`, `anualizam` e `anualizat`, respectivamente. Para ilustrar, vamos obter o total de consultas ao SCP em cada trimestre de janeiro de 2013 a março de 2015, isto é, vamos trimestralizar uma série mensal. Essa série já está armazenada no objeto `x2`, nas linhas 721 a 747:

```
> x2[721:747, c(1, 2)]

          data     SPC
721 2013-01-01 1998315
722 2013-02-01 1654399
723 2013-03-01 2026770
724 2013-04-01 1981335
725 2013-05-01 1986934
726 2013-06-01 1923743
727 2013-07-01 2031026
728 2013-08-01 2121497
729 2013-09-01 2008862
730 2013-10-01 2289600
731 2013-11-01 2175726
732 2013-12-01 2641710
733 2014-01-01 2048327
734 2014-02-01 1802043
735 2014-03-01 2041653
736 2014-04-01 1883237
737 2014-05-01 2116943
738 2014-06-01 1951507
739 2014-07-01 1957057
740 2014-08-01 2056062
```

```
741 2014-09-01 2133303
742 2014-10-01 2395967
743 2014-11-01 2153628
744 2014-12-01 2616096
745 2015-01-01 1968756
746 2015-02-01 1653168
747 2015-03-01 2115052
```

A série mensal pode ser trimestralizada com

```
> trimestralizam(x2[721:747, 2], FUN = sum)

[1] 5679484 5892012 6161385 7107036 5892023 5951687 6146422 7165691 5736976
```

Se, em vez do total em cada trimestre, quisermos saber a quantidade média de consultas em cada trimestre, podemos usar a mesma função:

```
> trimestralizam(x2[721:747, 2], FUN = mean)

[1] 1893161 1964004 2053795 2369012 1964008 1983896 2048807 2388564 1912325
```

Outro exemplo é o de anualizar uma série. Tomando como exemplo essa série mensal, temos duas opções:

1. anualizar a série mensal;
2. trimestralizar a série mensal e anualizar a série trimestral resultante.

Se estivermos somando os valores, essas duas opções devem dar o mesmo resultado, veja a seguir. Primeiro anualizamos os dados mensais em um passo apenas com

```
> x <- x2[721:747, 2]
> anualizam(x, FUN = sum)

[1] 24839917 25155823
```

Agora, vamos anualizar os dados mensais em dois passos. O primeiro passo transforma a série mensal em trimestral com

```
> x1 <- trimestralizam(x, FUN = sum)
```

para, em seguida, anualizar a série trimestral resultante com

```
> anualizat(x1, FUN = sum)

[1] 24839917 25155823
```

Os resultados não são iguais se estivermos usando mediana em vez de soma:

```
> x <- x2[721:747, 2]
> anualizam(x, FUN = median)

[1] 2017816 2052195
```

e

```
> x1 <- trimestralizam(x, FUN = median)
> anualizat(x1, FUN = median)

[1] 2014671 2048858
```

Mas quem vai querer saber da quantidade mediana de consultas anuais ao SPC, não é mesmo?

Do mesmo jeito que nas funções anteriores, estes são casos particulares de se transformar uma série de n observações em m grupos aplicando uma função `FUN` em cada grupo de k observações. Tomando um exemplo "maluco", vamos calcular o desvio-padrão de cada grupo de 5 observações de uma série de números de 1 a 20, fazendo:

```
> x <- 1:20
> n2mk(x, k = 5, FUN = sd)

[1] 1.581139 1.581139 1.581139 1.581139
```