

Ministério da Saúde



Volume 3 - Caderno R

ASIS - Análise de Situação de Saúde



Brasília, 2014



© 2013 Ministério da Saúde. Universidade Federal de Goiás.

Todos os direitos reservados. É permitida a reprodução parcial ou total desta obra, desde que citada a fonte e que não seja para venda ou qualquer fim comercial. Venda proibida. Distribuição gratuita. A responsabilidade pelos direitos autorais de textos e imagens desta obra é da área técnica. A coleção institucional do Ministério da Saúde pode ser acessada, na íntegra, na Biblioteca Virtual em Saúde do Ministério da Saúde: <www.saude.gov.br/bvs>.

Tiragem: 1ª edição – 2013 – 500 exemplares

Elaboração, distribuição e informações:

MINISTÉRIO DE SAÚDE

Secretaria de Vigilância em Saúde

Coordenação-Geral de Informações e Análise
Epidemiológica
SAF Sul, Trecho 2, lotes 05/06, bloco F,
Edifício Premium, Torre I, sala 14
CEP: 70070-600 – Brasília/DF
Telefone: (61) 3315 7708

UNIVERSIDADE FEDERAL DE GOIÁS

Instituto de Patologia Tropical e Saúde Pública

Departamento de Saúde Coletiva
Rua 235, S/N, Esq. 1ª avenida, sala 404, Setor Leste
Universitário
CEP: 74605-050 – Goiânia/GO
Telefone: (62) 3209-6109 / 3209-6115

Coordenação:

Ana Lúcia Sampaio Sgambatti de Andrade – UFG, IPTSP,
Departamento de Saúde Coletiva

Elaboração de texto:

Alessandra Corrêa Tomé Teixeira de Oliveira – PUC-GO
Ana Lúcia Sampaio Sgambatti de Andrade – UFG, IPTSP
Celina Maria Turchi Martelli – UFPE, Professora visitante
Elier Broche Cristo – CGIAE, SVS, Ministério da Saúde
Elisabeth Barboza França – UFMG
Elisabeth Carmen Duarte – UnB, OPAS
José Leopoldo Ferreira Antunes – USP, FSP
Marta Rovey de Souza – UFG, IPTSP
Noêmia Teixeira de Siqueira Filha – CPqAM – Fiocruz
Otaliba Libânio de Moraes Neto – UFG, IPTSP
Ricardo Arraes de Alencar Ximenes – UFPE, UPE
Ruth Minamisava – UFG, FEN
Tatiana Haruka Sugita – UFG
Walter Massa Ramalho – UnB, Faculdade da Ceilândia
Wayner Vieira de Souza – CCPqAM – Fiocruz

Capa, projeto gráfico e diagramação:

Silvestre Linhares da Silva

Normalização:

Delano de Aquino Silva – CGDI/Editora MS

Impresso no Brasil / Printed in Brazil

Ficha Catalográfica

Brasil. Ministério da Saúde.

Análise de situação de saúde : Caderno R / Ministério da Saúde, Secretaria de Vigilância em Saúde, Universidade Federal de Goiás – Brasília : Editora do Ministério da Saúde, 2013.
xx p. : il.

ISBN

1. Diagnóstico da Situação de Saúde. 2. Análise de situação. 3. Epidemiologia. I. Título.

CDU 616-036.22

Catálogo na fonte – Coordenação-Geral de Documentação e Informação – Editora MS – OS 2013/0338

Títulos para indexação:

Em inglês: Health situation analysis: textbook

Em espanhol: Análisis de situación de salud: libro texto

Prefácio

O R [1] é um programa gratuito, disponível online, utilizado para análise estatística. Aprender a utilizar a ferramenta R é um investimento que requer treino e dedicação. No Brasil o R vem sendo muito utilizado para análise de dados na área da saúde, especialmente em anos recentes. Pretende-se, com este curso, incentivar o uso do R pelos gestores para Análise de Situação de Saúde (ASIS) nos diferentes cenários epidemiológicos do país. Para tal, preparamos uma coletânea de exercícios, que incluem situações vivenciadas pelo gestor em sua rotina de trabalho. Estes exercícios integram o material instrucional do Curso EAD em Análise de Situação de Saúde, elaborado para capacitação de gestores. Os exercícios do Curso ASIS utilizam diferentes softwares para sua resolução. No entanto, neste treinamento do R, pretende-se solucionar os mesmos exercícios empregando-se o R. Na primeira parte deste volume, apresentamos um tutorial com uma introdução ao desenvolvimento com a ferramenta para análise Estatística R. São apresentados os passos para baixar o programa, instalar no microcomputador e executar comandos. E nas demais seções, apresentamos a resolução dos exercícios do Curso para os Módulos: Sistema de Informação em Saúde, Análise de Inquéritos Populacionais, Série Temporal e Dados Demográficos.

Bom trabalho a todos!

Sumário

Introdução / Treinamento R 6

Introdução	6
Treinamento para ferramenta de análise Estatística	6
Download e Instalação do R	7
Executando Comandos no R	11
Executando alguns exemplos	16
Tipos de variáveis e objetos no R	21
Geração de Gráficos	25
Operações de ordenação	28
Outras opções de gráficos	29
Opções de cores, paletas e legendas	32
Construção de Clustering	35
Variáveis aleatórias, geração de amostras	38

Resoluções do Módulo 2 42

Sistema de Informação em Saúde

Atividade 1	42
Questão A	42
Questão B	44
Questão D	46
Questão E	47
Atividade 2	48
Questão A	48
Atividade 3	48
Questão A	48
Atividade 5	51
Questão B	51
Questão C	51
Questão D	60

Questão E	63
Questão G	71
Questão H	82

Resoluções do Módulo 5 83

Análise de Inquéritos Populacionais

Atividade 1	83
Questão A	83
Questão B	86
Exemplo VIGITEL	93
Questão A	93

Resoluções do Módulo 6 98

Análise de Série Temporal

Atividade 1	98
Atividade 2	105
Atividade 3	110

Resoluções do Módulo 3 118

Análise de Dados Demográficos

Atividade 1	118
Atividade 2	121
Atividade 3	128

Considerações Gerais	132
Referências Bibliográficas	132

Introdução / Treinamento R

1. Introdução

O R [1] é uma ferramenta software-livre sobre a qual é possível estudar e entender melhor o funcionamento de suas funções, inclusive de fazer alterações, caso seja necessário, no código fonte. Teve origem no ano 1991 a partir da versão 3 do S-Plus no laboratório Bell, pelos desenvolvedores: Rich Becker, John Chamber e Allan Wilks. Em 1996 o R teve sua primeira distribuição, *Data Analysts Captivated by R's Power*: <http://www.nytimes.com/2009/01/07/technology/business-computing/07program.html>

Outra grande vantagem é a de ser gratuito, podendo ser usado para desenvolver ferramentas de uso cotidiano, sem se preocupar com futuros custos de patente. Existe um grande número (mais de 4.000) de bibliotecas desenvolvidas para o R, para as quais estão disponíveis os códigos fonte, as funções e a documentação com as referências bibliográficas usadas para desenvolver as metodologias estatísticas. Estas bibliotecas têm finalidades específicas e apresentam conjuntos de comandos que não estão presente no pacote que vem com a instalação básica.

Há na internet listas de discussão para o R, as quais podem ser consultadas para tirar dúvidas e até propor novas formas de solução nas bibliotecas. Inclusive o desenvolvedor pode juntar as funções que fez com determinado objetivo, criando um novo pacote específico para determinada área e enviá-lo para o site do R, seguindo os padrões da ferramenta R.

2. Treinamento para ferramenta de análise Estatística R

Este é um aplicativo que apresenta linguagem por comandos, permitindo ao usuário desenvolver *scripts* (programas, sequências de comandos) que depois podem ser encapsulados em funções com determinada finalidade. É possível construir uma função no R para fazer vários cálculos e depois gerar um gráfico. Suponha que seja um gráfico para cada Unidade Federativa (UF) do Brasil. Depois o usuário pode gerar um *loop* que em cada interação é chamada a fun-

ção desenvolvida e o gráfico é gerado automaticamente. De forma análoga, podemos percorrer uma base de dados nacional e gerar um determinado resultado para cada município. Ou seja, com o uso do R abrimos muitas opções e facilidades para realizar análises e gerarmos resultados em grande escala, existindo também, a possibilidade de usarmos o R para cálculos específicos e resultados individuais.

Os *scripts* desenvolvidos no R podem ser chamados por outros aplicativos. Isto nos permite uma grande capacidade de análise e processamento. Podemos colocar um *script* desenvolvido no R dentro de uma determinada rotina em um aplicativo *web*. Desta forma, podemos agendar tarefas no servidor para que essas rotinas sejam executadas em determinados dias e horários que o servidor esteja com menor nível de processamento. Os resultados gerados destes processos podem ser cálculos estatísticos ou até a geração de gráficos, em formato PDF, por exemplo. Depois o aplicativo apresentará uma grande rapidez para apresentar os resultados solicitados no site, pois está tudo previamente calculado.

2.1 Download e Instalação do R

O programa R pode ser baixado e instalado diretamente em microcomputadores, conectados à intranet de trabalho com acesso a internet, ou em notebook pessoais. Esta ferramenta pode ser utilizada em sistemas operacionais Windows ou Linux. Esta operação pode ser realizada diretamente do site oficial do R [1]: <http://www.r-project.org/>

2.1.1 Download do R

Nesta seção apresentamos como pode ser feito o *download* e instalação da ferramenta R. Segue o passo a passo para *download* do programa que instala o R no sistema operacional Windows:

1. Acesse o site do R: <http://www.r-project.org/>
2. Entre no *link* CRAN (Download, Packages)
3. Escolha um determinado servidor, exemplo: Brazil (University of Sao Paulo, Sao Paulo), <http://www.vps.fmvz.usp.br/CRAN/>
4. Escolha a opção: Download R for Windows
5. Entre no *Subdirectories*: **base**
6. Baixe o arquivo de instalação do R, a versão mais recente disponível, exemplo: Down-

load R 3.0.2 for Windows (52 megabytes, 32/64 bit). Salvar o arquivo EXE no micro que esteja sendo usado o arquivo disponível, exemplo: R-3.0.2-win.exe

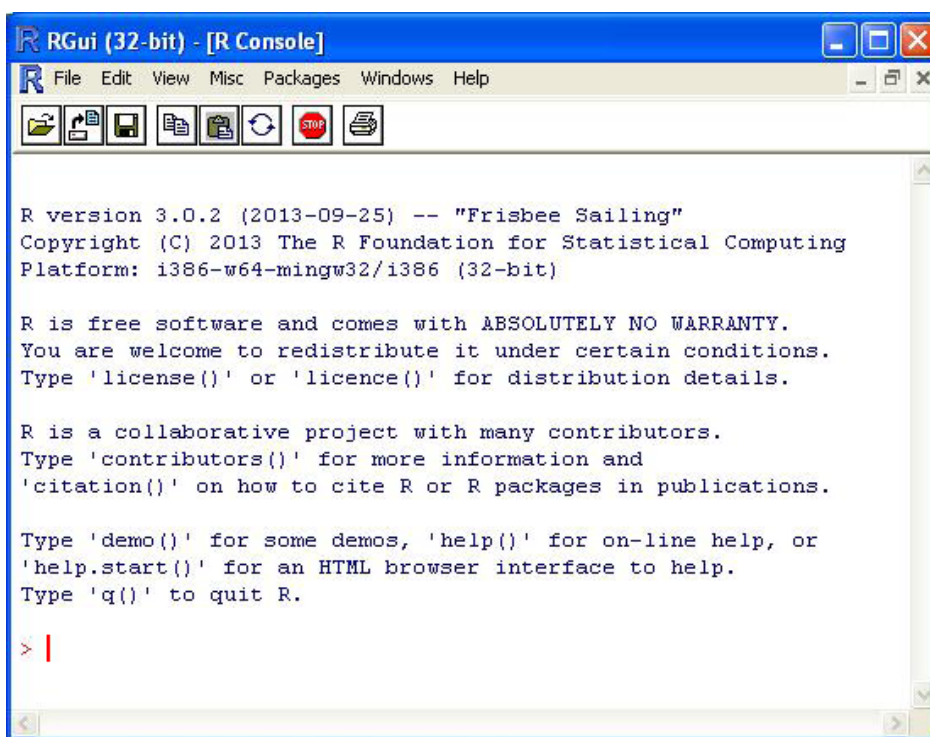
Após a realização dos passos anteriores o programa para instalação do R está no seu microcomputador, pronto para o R ser instalado.

2.1.2 Instalação do R

Após o *download* (seção Download do R) com sucesso do instalador do R, o mesmo pode ser instalado e configurado no seu microcomputador. Segue um passo a passo para a instalação do R:

1. Execute o arquivo “.exe” baixado, e confirme a opção Run ou Executar
2. Selecione o idioma que será utilizado na instalação, exemplo: English, e clique no botão OK
3. Na tela Welcome, clique no botão Next >
4. Na tela Information, clique no botão Next >
5. Mantenha a pasta proposta: C:\Program Files\R\R-3.0.2, ou caso seja de interesse altere o destino da instalação e clique no botão Next >. A pasta que não existia será criada.
6. Na tela seguinte aparecem opções para a versão do sistema operacional usado: 32 ou 64 bits. Selecione a opção correta dependendo do seu sistema operacional. Clique no botão Next >
7. Nas opções de instalação recomenda-se o padrão: No (accept defaults). Clique no botão Next >
8. É sugerida a Pasta “R” para ser criada no menu “Iniciar”. Clicar no botão Next >
9. Na próxima tela outras opções podem ser configuradas como a de criar um ícone no desktop. Clique no botão Next >
10. Neste ponto o R será instalado, uma barra de progresso indicará o andamento desta tarefa. Quando concluído, uma última tela informa a conclusão do processo. Clique no botão Finish
11. No desktop aparecerá o seguinte ícone: R i386 3.0.2
Pelo ícone criado no desktop ou pelo menu “Iniciar” pode ser executada a ferramenta R.

Quando este programa é executado aparece a tela abaixo:



```
R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

No círculo destacado em verde aparece o *prompt* de comando do R: **> |**

É neste cursor que os comandos do R podem ser digitados, e executados com a tecla “Enter”.

2.1.3 Instalando pacotes no R

Com a instalação do módulo básico do R (2.2. Instalação do R) teremos disponível um grande número de comandos e funções. A instalação básica é equipada de um conjunto básico de bibliotecas, por exemplo: **stat**. Nesta biblioteca encontramos testes estatísticos bastante usados como: Student’s t-Test () e Wilcoxon Rank Sum and Signed Rank Tests (). Para carregar a mesma no R, execute no R o comando: `(stats)`. Para acessar a documentação de um deles use no R o comando: **t.test**. Para mais detalhe em relação ao uso da ajuda no R consulte a seção Acessando a Ajuda do R.

Com facilidade podemos nos encontrar com um determinado exercício específico, para o qual seja mais conveniente usarmos alguma biblioteca específica já desenvolvida. Desta forma, fazemos uma grande economia de tempo ao não ter que desenvolver *scripts* que já estão pron-

tos e validados. Podemos citar o exemplo de amostra complexa, bastante usado nos inquéritos populacionais. Existe uma biblioteca desenvolvida para o R, **survey**, mas ela não vem com a instalação básica. Para verificar, execute o seguinte comando no R: `(survey)`. Aparecerá a seguinte mensagem:

```
Erro em library(survey) : there is no package called 'survey'.
```

Veja o passo a passo para localizar, baixar e instalar no seu micro outros pacotes, neste caso específico **survey**:

1. Certifique-se que já tenha o R instalado no seu microcomputador, seção Instalação do R
2. Acesse o site do R: <http://www.r-project.org/>
3. Entre no link CRAN (Download, Packages)
4. Escolha um determinado servidor, exemplo: Brazil (University of Sao Paulo, Sao Paulo), <http://www.vps.fmvz.usp.br/CRAN/>
5. Escolha a opção: Download R for Windows
6. Entre no *Subdirectories*: **contrib**
7. Selecione a pasta que indica a versão do R que foi instalada, exemplo: **3.0.2/**
8. Nesta pasta encontrará um arquivo chamado [survey_3.29.zip](#), o qual pode ser baixado e salvo no seu microcomputador.
9. Execute o R, e no menu “Pacotes” escolha a opção “Instalar pacote(s) a partir de arquivos zip locais ...”
10. Selecione a pasta e o arquivo (.zip) com o pacote que deseja instalar. Neste caso [survey_3.29.zip](#). É bastante comum que um determinado pacote dependa de outros, neste caso o R informa qual o outro(s) pacote necessário que precisa ser instalado. Nesse caso siga o mesmo passo a passo apresentado aqui.
11. Para acessar a documentação completa de um determinado pacote, acesse o link abaixo, o qual contém, ao final, o nome do pacote de interesse:
<http://cran.r-project.org/web/packages/survey/>

Após a realização dos passos anteriores o novo pacote (**survey**) está instalado no R e pode ser usado. É necessário carregar o pacote, para seu uso, e para tal execute o seguinte comando no R: `(survey)`. Desta vez não aparecerá a mensagem que o pacote não é encontrado. Agora que o novo pacote está carregado e pronto para ser usado no R, a documentação de suas funções pode ser consultada. Execute os comandos abaixo no R:

Survey sample analysis: ? [svydesign](#)

Confidence intervals for proportions: ? [svyciprop](#)

Outros pacotes de interesse podem ser instalados no R, como por exemplo: **foreign**. Este pacote inclui comandos para abrir e operar arquivos de bases de dados, como formato **DBF**. Para carregar arquivos DBF no R, é necessária a utilização do comando `foreign`, o qual foi desenvolvido na biblioteca **foreign**. O pacote aparece no site do R na pasta [contrib](#) como “[foreign_o.8-57.zip](#)”. Após sua instalação, o mesmo precisa ser colocado em uso. Utilize o seguinte comando no R:

```
(foreign) # open DBF
```

Para acessar a documentação (manual) deste pacote acesse o link:

<http://cran.r-project.org/web/packages/foreign/>

no qual pode ser baixado o arquivo PDF com o manual do mesmo.

2.2 Executando comandos no R

No R não é necessário nenhum caracter de fim de linha, exemplo: “,”, “;” ou “.”. Existem três formas para executar os comandos:

1. Digitar diretamente no *prompt* (>) de comando do R e dar “Enter” no final da linha
2. Primeiro digitar os comandos em um arquivo texto, com a intenção de, ao final, armazená-lo no microcomputador e ficar disponível para outras rodadas. Depois, copiar uma ou mais linhas do arquivo digitado e colar no *prompt* (>) de comando do R. Quando esta alternativa for usada, sempre pressionar o botão “Enter” na última linha (comando) do bloco, caso contrário o programa poderá ficar incompleto.
3. Digitar os comandos em um arquivo texto, com um determinado editor, exemplo: “**Bloco de notas++**” ou “**Tinn-R**”. Após este arquivo estar salvo no nosso microcomputador o mesmo pode ser executado pelo comando:

```
> source('C:/pasta/arquivo1.txt') # executa comandos no arquivo
```

Este comando irá localizar o arquivo indicado e executar seu conteúdo; cada linha no arquivo pode conter um comando diferente. Para escrever mais do que um comando na mesma linha é necessário separá-los com “;”, exemplo: ; Qualquer uma das opções

anteriores executa comandos no R. Dependendo do *script* construído a saída pode ser um resultado que será apresentado na própria tela do R, ou para um arquivo (pdf, txt, csv, png, tiff ou jpeg entre outros) que será gerado e armazenado no microcomputador, por exemplo: uma figura ou tabela com valores numéricos.

Para executar comandos dentro de um arquivo texto previamente criado faça o seguinte exemplo: **comando1; comando2**

1. Crie um arquivo de texto na pasta de dados (seção Criando uma pasta de trabalho para o R) com nome: “print.txt”;
2. Dentro deste arquivo texto escreva a linha:
print ('Executando comandos desde um arquivo TXT ...')
2. Salve e feche o arquivo criado;
3. No R execute o comando: > **source** (“C:/Curso_EAD_ASA/print.txt”)

No exemplo anterior o R irá acessar o arquivo “print.txt”, executar seu conteúdo, resultando o seguinte texto no *prompt* (>) de comando do R:
Executando comandos desde um arquivo TXT...

2.2.1 Criando uma pasta de trabalho para o R

Para desenvolver os exercícios propostos sugerimos criar uma **pasta de dados**, com nome: “**C:/Curso_EAD_ASA/**”. Caso a pasta anterior não possa ser criada, utilize outro caminho, onde tenha permissão.

Nos *scripts* com as soluções dos exercícios se utiliza a variável **pasta_dados** para definir o caminho da pasta de dados. Caso a pasta de dados (pasta de trabalho) não tenha sido criada no caminho e com o nome indicado, será necessário alterar os *scripts* trocando o conteúdo desta variável. Quando for necessário usar a variável **pasta_dados** em um determinado programa, ela sempre aparecerá nas primeiras 5 linhas do *script*:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

Considere que a pasta de dados foi criada em outro caminho ou em outro nome indicado, por exemplo: Z:/outra_pasta/. Neste caso a linha anterior no *script* deve ser trocada por:

`pasta_dados = 'Z:/outra_pasta/'`

Caso a alteração não seja realizada, o R apresentará a seguinte mensagem de erro:

Erro em `setwd(pasta_dados)` :

não é possível mudar o diretório de trabalho

no comando:

```
> setwd (pasta_dados)
```

Caso a pasta de dados tenha sido criada em “**C:/Curso_EAD_ASA/**” nenhuma mudança será necessária nos programas.

2.2.2 Acessando a Ajuda do R

Quando executamos o R e estamos no *prompt* de comando, podemos consultar a ajuda de qualquer comando com “**?**”, para isso execute no R o seguinte exemplo: `> ? mean`. O comando anterior “**?**” irá ativar o seu navegador e mostrar a ajuda para este comando “**mean**”, que no caso específico calcula a média aritmética de um determinado vetor.

A ajuda do R apresenta uma grande quantidade de informação dos comandos. Sugerimos que seja consultada para todos os comandos envolvidos na solução dos exercícios deste caderno. Mesmo para usuários com experiência em R, a ajuda pode lembrar parâmetros e funcionalidades do comando que sejam de interesse. Quando a ajuda do R é solicitada para consultar um determinado comando, as seguintes informações são fornecidas:

- ✓ Nome e descrição do comando;
- ✓ Funcionalidade e uso;
- ✓ O nome do pacote onde o comando se encontra no R, no caso do comando **mean**: {base}, que é o pacote básico do R, instalado na seção Instalação do R;
- ✓ Sintaxe completa e detalhada do comando com todos seus parâmetros (argumentos);
- ✓ Listagem dos argumentos, especificando o nome, tipo de variável e função do seu conteúdo;
- ✓ Valores que devolve: nome, descrição e conteúdo;
- ✓ Exemplos de uso do comando;
- ✓ O “**See Also**” apresenta comandos relacionados, de grande utilidade para conhecer outros comandos com determinada associação com o atual. Por exemplo, no tipo de variável de entrada que trabalha;

- ✓ Referências Bibliográficas, que foram usadas no desenvolvimento do R para implementar este comando.

Segue a saída do R, quando a ajuda do comando *mean* é consultada:

```
> ? mean
```

```
(http://127.0.0.1:15790/library/base/html/mean.html)
```

```
mean {base}
```

R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

```
## Default S3 method:
```

```
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

- `x` An R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects, and for data frames all of whose columns have a method. Complex vectors are allowed for `trim = 0`, only.
- `trim` the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.
- `na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.
- `...` further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.

Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

[Package *base* version 2.15.0 [Index](#)]

No site do R encontramos um grande volume de documentação, manuais desenvolvidos para o estudo e uso desta ferramenta. Para acessar esta documentação, siga os passos abaixo:

1. Acesse o site do R: <http://www.r-project.org/>
2. Entre no link [Manuals](#) em **Documentation**

Neste local se encontra a documentação em Inglês. Para acessar a documentação em Português ou Espanhol, estando no local anterior:

1. Entre em “*Translations of manuals into other languages than English are available from the [contributed documentation](#) section (only a few translations are available).*”
2. E na seção **Non-English Documents: Portuguese**

Especificamente recomendamos os seguintes manuais, nesta ordem:

Portuguese

1. “Introdução ao uso do programa R” Victor Lemes Landeiro, Ago/2011.
<http://cran.r-project.org/doc/contrib/Landeiro-Introducao.pdf>
2. “Introdução à Programação em R” Luis Torgo, Out/2006.
<http://cran.r-project.org/doc/contrib/Torgo-ProgrammingIntro.pdf>
3. “Tópicos de Estatística utilizando R” Fernando Itano, Ago/2007.
<http://cran.r-project.org/doc/contrib/Itano-descriptive-stats.pdf>
4. “Bioestatística usando R” Colin Robert Beasley, 2004.
<http://cran.r-project.org/doc/contrib/Beasley-BioestatisticaUsandoR.pdf>

Os manuais anteriormente referenciados e outros estão disponíveis no site do R em formato PDF. No site também podem ser encontrados os conjuntos de dados necessários para determinados exercícios, *script* em R, referências bibliográficas e documentação. Nestes manuais do R podem ser encontrados os comandos usados nos exercícios resolvidos neste caderno, com outros exemplos e explicações. O aluno pode fazer testes em novos exercícios empregando estes e outros comandos no R.

2.3 Executando alguns exemplos

Após a instalação com êxito do R e seus pacotes de interesse, está pronto para executar comando no R. Primeiro carregue o R e espere aparecer o *prompt* de comando “> |” deste aplicativo. Neste ponto o R pode ser usado, faça os testes abaixo:

Criar um vetor (nome da variável: **va**) com valores numéricos inteiros:

```
> va = c (3, 2, 8, 13, 7, 1, 25, 12, 18, 9, 1, 0, 5, 3, 5, 2, 13, 21, 23, 23, 23, 5)
```


Neste caso não há saída, somente se cria a variável. O comando “c” no R, cria vetores (coluna), pode ser interpretado como: c = **concatenar**.

Para visualizar o conteúdo do vetor **va** criado:

```
> va
```

Neste caso a saída será: [1] 3 2 8 13 7 1 25 12 18 9 1 0 5 3 5 2 13 21 23 23 23 5

A notação “[1]” indica que na primeira posição se encontra o elemento “3” e os outros lhe seguem. Desta forma, usamos apenas o nome da variável no *prompt* para visualizar seu conteúdo.

Visualizar o tamanho do vetor **va**:

```
> length (va)
```

O R apresentará o número de elementos (tamanho =20) armazenados no vetor **va**.

Para acessar a estrutura de uma determinada variável:

```
> str (va)
```

O R apresentará: num [1:22] 3 2 8 13 7 1 25 12 18 9 ...

Indicando que o vetor “**va**” é numérico e tem 22 elementos.

Gerar as medidas resumo do vetor **va**:

```
> summary (va)
```

Saída no R, uma tabela com as estatísticas resumo para a variável **va**:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.00	3.00	7.50	10.09	16.75	25.00

Para calcular sua média:

```
> mean (va)
```

Para gerar um gráfico com os pontos do vetor “**va**”:

```
>plot (va, pch = 15) # pch é usado para definir o tipo de ponto usado
```

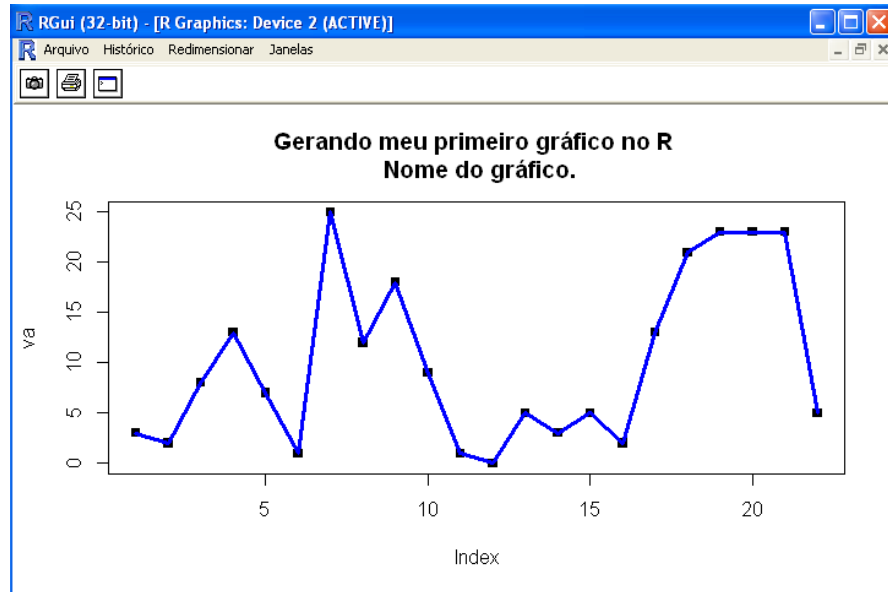
Para gerar a linha que une os pontos do vetor “**va**”:

```
> points (va, type = 'l', col = 'blue', lwd = 3) # l=linha, lwd= largura da linha
```

Para colocar nome na figura, continuando com a janela do gráfico aberta:

```
> title ('Gerando meu primeiro gráfico no R \n Nome do gráfico.')
```

O R apresentará uma nova janela (*R Graphics*). Esta janela é a saída padrão do R para apresentação dos gráficos, e neste curso identificamos como “**janela gráfica do R**”. Segue o gráfico gerado no R, com os últimos três comandos executados:



Gerando um boxplot:

```
> boxplot (va)
```

Gerando um histograma:

```
> hist (va, nclass=12) # com o parâmetro nclass alteramos a largura das  
barras no histograma
```

2.3.1 Comandos e símbolos mais usados nos exercícios

Segue uma lista com os comandos mais usados na solução dos exercícios:

1. `c (1,2)` # define um vetor com os valores 1 e 2, c=concatenar
2. `round (x, digits=2)` # devolve o valor de x com 2 casas decimais
3. `data.frame (var1, var2)` # cria um data frame (matriz) com duas variáveis (colunas)

4. `names (variavel)` # acessa os nomes das colunas de uma matriz ou data frame
5. `matrix (y, nrow = 3, ncol = 5, byrow = TRUE)` # cria uma matriz a partir do vetor "y", com 3 linhas, e 5 colunas. Neste caso a distribuição dos valores do vetor para a matriz é realizada por linhas.
6. `listaa = list ()` # cria a lista "listaa" vazia
7. `listaa [[1]]` # acessa o primeiro elemento da lista "listaa"
8. `sum (vetorb)` # soma os elementos do vetor 'vetorb', retornando um único valor
9. `function(a,b){...}` # define uma função em R, com os parâmetros a e b.
10. `length (vetorb)` # devolve o tamanho (número de elementos) do vetor "vetorb"
11. `as.integer (strings)` # converte de string (strings) para numérico
12. `substring (strings, 2, 3)` # devolve uma substring da variável "strings" da posição 2 a 3
13. `apply (x,1,sum)` # executa a soma das colunas da matriz "x"
14. `abline` # gera linhas no gráfico ativo
15. `title (tit)` # coloca o conteúdo da variável tipo string "tit" como nome do gráfico
16. `legend` # define a legenda do gráfico
17. `axis` # escreve a legenda dos eixos (X ou Y) de um gráfico
18. `seq (0,10,2)` # gera um vetor com a sequência que inicia em 0, tem intervalos de 2 e termina em 10, ou seja: `c(0,2,4,6,8,10)`
19. `par` # comando no R que permite alterar a configuração padrão da geração de gráficos
20. `which` # comando que permite saber quais índices são TRUE, ou seja, cumprem determinada condição (verdadeiros)
21. `library (pacotep)` # carrega o pacote "pacotep" no R, que já foi instalado previamente. Consulte a seção Instalando pacotes no R
22. `setwd (pastan)` # troca a pasta de trabalho do R para a nova pasta "pastan"
23. `dir ()` # visualiza os arquivos na pasta de trabalho, ou outra pasta se for especificada
24. `str (varb)` # mostra a estrutura da variável "varb"
25. `read.dbf (arquivoa, s.is = TRUE)` # carrega um arquivo em formato DBF (que o nome está no conteúdo da variável string "varb"), o parâmetro `as.is=TRUE` garante que colunas de texto não sejam convertidas para factors automaticamente.

Factor é um tipo de dados no R bastante usado quando o conteúdo de uma determinada coluna toma poucos valores diferentes, neste caso o R define os valores diferentes e depois os acessa com um índice.

```
26.read.table (file ='arquivoa.CSV', sep = ';', header = TRUE, stringsAsFactors = FALSE, dec='.')  
# carregao arquivo de texto "arquivoa.CSV", o qual tem suas colunas separadas por ";",  
tem cabeçalho (nome nas colunas), e o separador decimal é ".". Caso o Excel esteja con-  
figurado no padrão brasileiro (português), toque no comando R para dec=','
```

Segue uma lista de símbolos usados na solução dos exercícios:

1. = símbolo de atribuição
2. <- símbolo de atribuição, outra alternativa
3. == símbolo de comparação, diferente de atribuição
4. { símbolo para iniciar função
5. } símbolo para fechar função
6. ! símbolo da negação (*not*) o R devolverá TRUE se a condição que segue "!" for falsa
7. ? Somando no R para chamar a ajuda de um determinado comando,
exemplo: `?mean`

2.3.2 Conversão de arquivos XLSX para CSV

Com frequência precisamos carregar dados no R, e uma opção bastante usada é mediante arquivos texto em formato CSV. Nesta seção apresentamos como converter arquivos XLSX (Excel) para texto (CSV). Quando esteja no Excel, com a aba de interesse ativa, entre no menu principal na opção "Salvar como":

- Salvar como tipo: CSV (separado por vírgulas) (*.csv)
- Nome do arquivo: nomeArquivo.csv
- Salvar em: **C:/Curso_EAD_ASA** (pasta de dados, seção Criando uma pasta de trabalho para o R)

Nas próximas seções é apresentado o uso do R com maior detalhamento, assim como as soluções dos exercícios propostos no Curso EAD em Análise de Situação de Saúde (ASIS). Os comandos apresentados, para a solução dos exercícios, podem ser digitados no R para obter os resultados. Como visto na seção Executando comandos no R, estes comandos também podem ser: a) copiados e colados no R, ou b) armazenados em um arquivo texto e depois serem executado pelo comando: `source ('caminho/arquivo.txt')`.

2.4 Tipos de variáveis e objetos no R

Os objetos no R podem ser criados diretamente com valores específicos (seção Executando alguns exemplos) ou desde arquivo de dados. Entre as fontes usadas temos as bases de dados do DATASUS (.dbf), planilhas Excel, arquivos texto (.csv), entre outros. Para os exemplos iniciais escolhemos CSV, um formato muito usado e presente como uma das saídas no site do DATASUS na hora de fazer a exportação dos dados. Consulte a atividade nº1 do Módulo de Séries Temporais do Curso ASIS.

Tendência da mortalidade infantil em dois Estados brasileiros: São Paulo (SP) e Maranhão (MA). Segue uma tabela com os valores apresentados no caderno de exercícios:

Ano	Obitos Menor 1 ano SP	NV SP	TMI x 1000 SP	Obitos Menor 1 ano MA	NV MA	TMI x 1000 MA
1996	15710	699013	22,47	1069	61056	17,51
1997	15159	701947	21,60	1213	75392	16,09
1998	13756	693413	19,84	1501	79272	18,93
1999	12796	714428	17,91	1543	96587	15,98
2000	11922	687779	17,33	1898	100811	18,83
2001	10437	632483	16,50	2188	108527	20,16
2002	9534	623302	15,30	2425	117917	20,57
2003	9273	610555	15,19	2473	127920	19,33
2004	8959	618080	14,49	2213	126518	17,49
2005	8353	618880	13,50	2465	130266	18,92
2006	8078	603368	13,39	2240	127724	17,54
2007	7774	595408	13,06	2164	127307	17,00
2008	7585	601795	12,60	2110	128302	16,45
2009	7482	598473	12,50	2051	123635	16,59
2010	7163	601352	11,91	1860	119566	15,56

Para trabalhar com estes valores, é necessário converter o arquivo para CSV. Consulte a seção Conversão de arquivos XLSX para CSV. Sugerimos o seguinte nome para o arquivo de dados: **Modulo_SerieTemporal_Atividade1.csv**

Segue programa em R:

```
### Dados da Atividade nº1, Módulo Séries Temporais, curso ASIS  
# define a pasta de trabalho, que precisa ser previamente criada
```

```

pasta_dados = `C:/Curso_EAD_ASA/`
# o R se posiciona na pasta definida
setwd (pasta_dados)

# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()

# Carrega o arquivo texto CSV com os dados
dad = read.table (
file = `Modulo_SerieTemporal_Atividade1.csv`,
sep = ';',
header = TRUE,
stringsAsFactors = FALSE,
dec = ',')

```

Para consultar o tipo de objeto construído no R, com a importação do arquivo CSV:

```
> str(dad)
```

O comando “**str**” apresenta a estrutura do dado (data.frame) que é um container contendo uma variável por cada coluna presente no arquivo. A notação apresentada: *15 obs. of 7 variables*, significa que temos 15 linhas e 7 colunas. A saída: `$ Ano: int 1996 1997 ...`, representa que a variável (coluna) Ano está formada somente por valores inteiros, os anos (1996 a 2010).

Para acessar a tabela completa:

```
> dad
```

Para acessar uma determinada coluna da tabela, por exemplo: as taxas anuais da mortalidade infantil no Maranhão:

```
> dad $ TMI.x.1000.MA
```

O comando anterior primeiro especifica a tabela (data.frame), e após “\$” o nome da coluna.

Para obter o tamanho da coluna (número de elementos):

```
> length(dad $ TMI.x.1000.MA)
```

Para entender melhor a nova estrutura consultada:

```
> str (dad $ TMI.x.1000.MA)
```

Especifica que a coluna foi definida como um vetor numérico com 15 elementos.

Estatísticas para a variável TMI (Taxa da Mortalidade Infantil) de Maranhão:

```
> min(dad $ TMI.x.1000.MA) # mínimo
> mean(dad $ TMI.x.1000.MA) # média
> round(mean(dad $ TMI.x.1000.MA), digits=2) # média com 2 casas decimais
> ? round # consultando a ajuda
> median( dad $ TMI.x.1000.MA) # mediana = II Q = 2do quartil
> max( dad $ TMI.x.1000.MA) # máximo
> var( dad $ TMI.x.1000.MA) # variância
> sd( dad $ TMI.x.1000.MA) # desvio padrão
> dad $ Ano [which.max( dad $ TMI.x.1000.MA)] # ano com maior taxa
> range( dad $ TMI.x.1000.MA) # vetor com min e max
```

Medidas Resumo para as TMI do Maranhão:

```
> summary( dad $ TMI.x.1000.MA)
```

Cálculo com fórmula, exemplo “**média**”:

```
> mean( dad $ TMI.x.1000.MA)
> sum( dad $ TMI.x.1000.MA ) / length( dad $ TMI.x.1000.MA) # somatório/n
```

Acessando valores internos de um vetor, diretamente pela sua posição (índice):

```
> dad $ Ano [15] # ano 2010, se dúvida consulte dad
```

Acessando a TMI para UF MA em 2010:

```
> dad $ TMI.x.1000.MA [15] # tmi 2010
```

Geração de vetores e sequências:

```
> ind_Anos05a10 = 10:15 # índice dos anos de 2005 a 2010
> ind_Anos05a10
```

```
> dad $ Ano [ind_Anos05a10] # anos de 2005 a 2010
> dad $ TMI.x.1000.MA [ind_Anos05a10] # tmi de 2005 a 2010
```

Lembrando a estrutura de “dad”:

```
> str(dad)
> dad
```

Criando um segundo data-frame, selecionando as colunas 1ª (anos) e 7ª (TMI - MA) e as linhas correspondentes aos anos de 2005 a 2010:

```
> dad_05a10 = dad [ind_Anos05a10, c(1,7)] # c = concatenar
> dad_05a10
```

Gerando uma sequência com índices contendo os anos pares:

```
> anos.pares = seq(from = 1, to = 15, by = 2 ) # posição em "dad" dos anos pares
> anos.pares
> dad $ Ano [anos.pares] # anos pares de 96 a 2010
> dad $ TMI.x.1000.MA [anos.pares] # tmi dos anos pares
> dad.aPares = dad [anos.pares, c(1,7)]
> dad.aPares
```

Atividade prática r-1. Empregando os comandos anteriores como exemplo, gere as TMI para os anos ímpares, com esta finalidade sugerimos criar as variáveis: `anos.impares` e `dad.aImpares`. Para a realização das atividades práticas sugerimos a seguinte sequência:

1. Leve as linhas com os comandos necessários para **Notepad++** ou **Tinn-R**, e crie um novo arquivo de texto
2. Retire os símbolos de *prompt* do R (>) nos inícios de linhas com comandos
3. As linhas que não contem comandos (só texto) devem ser comentadas, colocando o símbolo “#” no início da linha, ou eliminadas. Comentar é a melhor prática, pois assim os scripts ficam documentados.
4. Faça as mudanças que forem necessárias no novo arquivo texto que está sendo gerado
5. Quando concluído o *script* com os comandos em R, salve o arquivo texto com os comandos na pasta de trabalho do Curso ASIS (C:\Curso_EAD_ASA) com o seguinte padrão de nome: **Curso_ASIS_R_ativp_1.r**
6. Copiar e colar o *script* completo no *prompt* do R

7. Caso ocorra algum erro, ou ainda não obtenha o resultado desejado: repita os passos anteriores: 4 a 7.

2.5 Geração de Gráficos

O R apresenta um grande conjunto de opções para a geração de gráficos. Desde as margens, limites, eixos, orientação, tipo de gráficos e cores, quase todos os parâmetros da saída dos gráficos podem ser alterados e modificados ao nosso interesse. Nesta seção o principal objetivo é transferir ao aluno algumas destas opções no R.

Segue um exemplo para as TMI do Maranhão, neste caso identificando os anos de interesse com a geração de índice que resultam de filtros construídos com condicionais:

```
> plot ( y= dad $ TMI.x.1000.MA
        , x= dad $ Ano
        , pch = 15
        , type = 'l'
        , col = 'blue'
        , lwd = 3
        , las = 1
        , xlab = "Anos"
        , ylab = "TMI MA"
        , main = "Taxa da Mortalidade Infantil (x1000) de Maranhão \n Anos
de 1996 a 2010" )
```

O R apresentará uma nova janela (*R Graphics*). Esta janela é a saída padrão do R para apresentação dos gráficos, e neste curso identificamos como “**janela gráfica do R**”. Não feche a janela gráfica criada anteriormente, mantenha a mesma aberta durante a execução de todos os exemplos. Caso a janela gráfica esteja maximizada, para voltar a visualizar o *prompt* () de comandos, faça *restore down*, mas não feche nem minimize.

Para consultar a ajuda dos comandos usados anteriormente:

```
> ? plot
> ? par
```

Gerando o *grid* horizontal (taxas) e vertical (anos) no gráfico anterior:

```
> abline (v = dad $ Ano) # grid vertical
> abline (h = 16,20) # grid horizontal
```

Identificando no gráfico os anos com TMI acima de 19:

```
> ind.sup19 = which (dad $ TMI.x.1000.MA > 19 ) # anos com tmi > 19
  ind.sup19
> dad $ Ano [ ind.sup19 ]
> dad $ TMI.x.1000.MA [ ind.sup19 ]
> dad [ind.sup19, c(1,7)]
> points ( y= dad $ TMI.x.1000.MA [ ind.sup19 ]
          , x= dad $ Ano[ ind.sup19 ]
          , pch = 15
          , col = 'red'
          )

> text ( x= dad $ Ano[ ind.sup19 ]
        , y= dad $ TMI.x.1000.MA[ind.sup19]
        , labels = dad$Ano[ind.sup19]
        , pos = 4 ) # pos=4 indica à direita

> ? points
> ? text
```

Para salvar o gráfico gerado, primeiro deixe a janela do gráfico como ativa e maximize a mesma. No menu *File* (Arquivo) escolha a opção “*Save as*” (Salvar como). Depois escolha o nome (**grafi_tmi19_sup.pdf**) na pasta de trabalho. Sugerimos o formato **PDF** por ser um formato vetorizado, ou seja, pode ser diminuído ou aumentado de tamanho sem perda de resolução.

Atividade prática r-2. A partir do gráfico gerado anteriormente, gere um novo destacando as TMI abaixo de 19 em cor verde. Com esta finalidade gere um novo índice: **ind.inf19**. Salve a nova figura gerada com nome: **graf2_ativ2_tmi19_inf.pdf**. A janela do gráfico anterior pode permanecer aberta. Sugerimos maximizar a janela gráfica antes de salvar, para que a figura armazenada ocupe a tela completa. Lembre-se de gerar, na pasta de trabalho, o arquivo texto **Curso_ASIS_R_ativp_2.r**, com o passo a passo detalhado na **Atividade prática r-1**.

Para gerar Boxplot:

```
> boxplot ( dad $ TMI.x.1000.MA
           , col = "#C8C8C8"
           , main = "TMI de Maranhão"
           , las = 1
           )
> ? boxplot
```

Para maior detalhamento e opções de cores, consulte a seção Opções de cores, paletas e legendas.

Gerar histograma com as TMI:

```
> hist ( dad $ TMI.x.1000.MA
        , col = "#C8C8C8"
        , main = "TMI de Maranhão"
        , las = 1
        , nclass = 12
        , xlab = "TMI"
        , ylab = "Frequência"
        )
> ? hist
```

Criar gráfico com 2 eixos Y (número de óbitos (x100) à esquerda e número de nascidos vivos (x1000) à direita):

```
> # mar = c(5, 4, 4, 2)+ 0.1 # padrão c(bottom, left, top, right)
par(mar = c(5, 4, 4, 4)+ 0.1 ) # para aumentar espaço no gráfico à direita
plot ( y = dad $ Obitos.Menor.1.ano.MA /100
      , x= dad $ Ano
      , pch = 15
      , type = 'l'
      , col = 'blue'
      , lwd = 3
      , las = 1
      , xlab = "Anos"
```

```

, ylab = "Número de Óbitos (x100) menor de 1 ano (linha azul)"
, main = "Maranhão 1996 a 2010 \n Número de óbitos (x100) menor de 1 ano (eixo
Y à esquerda - azul) \n Número de nascidos vivos (x1000) (eixo Y à direita - vermelho)" )
abline ( v= dad $ Ano ) # grid vertical nos anos
points(y= dad$Obitos.Menor.1.ano.MA/100, x= dad$Ano, pch =15, col ='blue')

```

Construindo o segundo eixo Y, à direita, para o número de nascidos vivos (x1000):

```

> par (new = TRUE)
plot(dad$Ano, dad$NV.MA/1000, type="l", axes=F, frame=T, ann=F, col='red', lwd=3)
points(dad $ Ano, dad $ NV.MA /1000, pch =16, col ='red')
axis(4, las=1)

```

Especifica a legenda do gráfico:

```

> legend (2002, 80, c('Número de Óbitos (x100) menor de 1 ano (eixo Y à
esquerda)', 'Número de nascidos vivos (x1000) (eixo Y à direita)'), lwd=3, col=-
c('blue', 'red'), lty=1, pch=c(15,16))

```

Salve o gráfico gerado na pasta de trabalho com nome: **graf3_2eixosY_ativ3_ob_nv.pdf**. Antes de salvar a figura, verifique que a tela gráfica foi maximizada.

Atividade prática r-3. Proponha um novo nome para o gráfico anterior e faça a alteração no R. Gere o gráfico mantendo a legenda mas sem o *grid*, maximize a figura e salve (**graf4_2eixosY_ativ3_ob_nv.pdf**) na pasta do curso. Depois abra o arquivo gerado. Salve o novo programa em R na pasta de trabalho: **Curso_ASIS_R_ativp_3.r**

2.6 Operações de ordenação

No processamento dos dados, com frequência são necessárias operações que envolvem ordenação. O R tem algumas funções prontas para esta finalidade, entre elas as principais são: **order**, **sort** e **rank**.

Considere que se deseja ordenar os anos pelas TMI, iniciando com as maiores taxas. Para entender melhor o funcionamento da função **order**, execute no *prompt* do R o seguinte comando:

```

> ordem_tmi = order (dad $ TMI.x.1000.MA, decreasing = TRUE)

```

```
> dad $ TMI.x.1000.MA [ ordem_tmi ] # tmi ordenadas
> dad $ Ano [ ordem_tmi ] # anos com ordem pelas maiores tmi
> dad [ ordem_tmi , c(1,7) ]
```

Interprete a tabela gerada anteriormente. Qual ordem seguem os anos, na primeira coluna?

Considere que deseja ordenar as TMI, de menor a maior, sem interessar os anos onde elas ocorreram:

```
> sort (dad $ TMI.x.1000.MA, decreasing = FALSE)
```

Outra função com finalidade similar é **rank**. Esta função também pode ser usada para trabalharmos com as estatística de ordem (posto = posição):

```
> rank (dad $ TMI.x.1000.MA)
```

Interprete o vetor saída gerado anteriormente. O que significam a primeira posição ser 8?

Para interpretar o resultado gerado pelo **rank**, podemos pensar que atribui uma nota para as TMI, iniciando pelas menores. Desta forma, o resultado do comando anterior termina em 1, por ser a última taxa (15.56) a menor de todas as TMI. De forma análoga, na posição 7 resultou 15, pelo fato da TMI no 7º ano (2002) ser a maior (20.57). Note que **rank** é diferente de **sort** (ordena os valores do vetor das TMI). E **rank** também é diferente de **order** (gera um vetor de posição, indicando se ocorrer a ordenação, onde se encontraria cada elemento no vetor original).

2.7 Outras opções de gráficos

O R tem diferentes tipos e opções de gráficos que podem ser gerados dependendo do conjunto de dados que esteja sendo trabalhado. Para visualizar todos os exemplos de um determinado comando no R para gerar gráficos digite: **example(comando)**. O R apresentará a janela gráfica, em ocasiões inicialmente vazia, para mudar o exemplo clique (ou Enter) na janela gráfica.

Segue opções para os gráficos de **barra**:

```
> example ( barplot )
> barplot (dad $ TMI.x.1000.MA, names.arg = dad $ Ano, main = "TMI Maranhão")
> ? barplot
```

Gráficos de **pizza**: estilos, opções e exemplos:

```
> example ( pie )
> pie (dad $ TMI.x.1000.MA, labels = dad $ Ano, main = "TMI Maranhão 1996 a 2010")
> ? pie
```

Outros gráficos, mais opções e exemplos:

```
> example ( plot )
> example ( points )
> example ( hist )
> example ( boxplot )
```

Podemos dividir a janela padrão de saída do R em “**nl**” linhas e “**nc**” colunas, com esta finalidade usamos o parâmetro **mfrow** dentro do comando **par** (configuração prévia dos gráficos). Este comando pode ser executado, por exemplo, antes de um **plot**:

```
> par(mfrow =c(1,2)) # indica que serão 2 gráficos na mesma tela na horizontal
> boxplot ( dad $ TMI.x.1000.MA
           , col = "#C8C8C8"
           , main = "TMI de Maranhão"
           , las = 1 )
> boxplot ( dad $ TMI.x.1000.SP
           , col = "#C8C8C8"
           , main = "TMI de São Paulo"
           , las = 1 )
```

Neste caso temos uma questão não desejada, note que as escalas no eixo Y dos dois boxplots não é a mesma. Ambos boxplots podem ser gerados juntos, desta forma a escala no eixo Y será a mesma. Para que a janela gráfica do R seja reiniciada, e não esteja dividida em duas partes, é necessário fecha-la (não o R, nem o *prompt* de comando, somente a janela gráfica). Para que o R abra uma nova janela gráfica:

```
> boxplot ( dad [ , c(4,7) ]
           , main = "TMI de São Paulo e Maranhão"
           , las = 1
           , col = c("blue","dark green"))
```

Salve o gráfico gerado com 2 boxplots, com nome **graf5_R_ST_boxplot2.pdf** na pasta de trabalho.

Da mesma forma que geramos e visualizamos gráficos no R na tela do micro (saída padrão), a ferramenta nos permite gera um arquivo PDF. Neste caso, cada gráfico gerado será uma nova pagina no arquivo .pdf que se encontra aberto. Segue um exemplo de como gerarmos um pdf, com 2 paginas por exemplo:

```
> pdf ( file = 'graf6_R_ST_bp2_pg2.pdf' , paper='a4r', width=11, height=8 )
> boxplot ( dad $ TMI.x.1000.MA
           , col = "#C8C8C8"
           , main = "TMI de Maranhão"
           , las = 1 )
> boxplot ( dad $ TMI.x.1000.SP
           , col = "#C8C8C8"
           , main = "TMI de São Paulo"
           , las = 1 )
> dev.off() # fecha o arquivo pdf que está aberto.
```

Neste caso os gráficos gerados não foram visualizados na tela. Para verificar a geração dos mesmos, confirme a existência do arquivo **graf6_R_ST_bp2_pg2.pdf** na pasta de trabalho do curso: **C:\Curso_EAD_ASA**. Note que antes de fechar o arquivo pdf que estava em aberto, cada gráfico gerado cria uma nova página no arquivo PDF. Esta ideia pode ser usada quando queremos gerar vários resultado e armazenar eles no mesmo arquivo.

Atividade prática r-4. Gere um novo arquivo na pasta de trabalho, com nome “**graf7_ativ4_pg4.pdf**”, contendo 4 páginas. As páginas devem ser preenchidas por diferentes gráficos da sua escolha, podendo selecionar entre os que foram gerados nas atividades anteriores. Salve o *script* de comandos R gerado na pasta de trabalho: **Curso_ASIS_R_ativp_4.r**

Além das opções de gráficos vistos anteriormente, o R nos permite outras opções, como o desenho de áreas de interesse. Isto pode ser interessante quando queremos destacar determinadas áreas nos gráficos que queremos apresentar para chamar a atenção em fatos específicos.

Para acessar exemplos na marcação de áreas específicas dentro dos gráficos:

```
> example ( polygon )
```

Após a visualização dos exemplos anteriores, feche a janela gráfica do R. Segue um exemplo para ilustrar o destaque de determinadas áreas:

```
> plot ( 0,0, xlim=c(0,1), ylim=c(0,1) , main="Brasil" )
> polygon ( x= c(0,1,1,0,0) , y= c(0,0,1,1,0) , col= "#009300" )
> polygon ( x= c(.5,.9,.5,.1,.5) , y= c(.1,.5,.9,.5,.1) , col= "yellow" )
> text ( .4, .5, 'ORDEM E PROGRESSO', pos ='4', col= "dark green", font=2 )
```

Para melhor visualização da imagem, maximizar a janela gráfica do R. Salvar esta figura como: **graf8_Brasil.PNG** na pasta de trabalho. Este exemplo mostra a grande variabilidade de opções e liberdade para estarmos gerando figuras e gráficos em diferentes formatos e formas diferentes. Na hora de salvar a figura escolha formato **PNG**.

Atividade prática r-5. Gere um novo gráfico, fazendo alterações na figura anterior:

- Legenda no eixo x: **Eixo X: Brasil**
- Legenda no eixo y: **Eixo Y: Ano 2014**
- Nome do gráfico: **Brasil – 2014.**
- Proponha e faça alguma outra alteração na figura construída.

Salve o programa gerado na pasta de trabalho: **Curso_ASIS_R_ativp_5.r**

Salve a nova figura na pasta de trabalho: **graf9_Brasil_2.PNG**

2.8 Opções de cores, paletas e legendas

No R há diferentes opções de especificar a cor desejada nos gráficos, entre elas, o próprio nome da cor em inglês, exemplo: “**red**”. Mas também é permitido especificar a cor pelo seu código de 6 dígitos (**#VVEEAA**). Cada par de dígitos indica uma intensidade das cores primárias: VV = vermelho, EE = verde e AA = azul. Os dígitos podem tomar valores nas letras e números, nesta ordem: F, E, D, C, B, A, 9, 8, 7, ..., 1, 0), de maior a menor intensidade. Desta forma, se formam as cores para os seguintes códigos:

- **# FF0000 – vermelho**
- **# 00FF00 – verde**
- **# 0000FF – azul**
- **# 000000 – preto**
- **# FFFFFFFF – branco**

2.8.1 Paleta de cores verde – preto – vermelho

A legenda de cores é o degradê de cores, o qual pode ser usado nos gráficos. Esta legenda de cores inicia no verde claro, que vai até o escuro, depois passa pelo preto, e por fim vai desde o vermelho escuro, até o mais claro. Sendo que o vermelho mais claro representa o mínimo, o preto o médio, e o vermelho claro o máximo. No total são 512 cores diferentes.

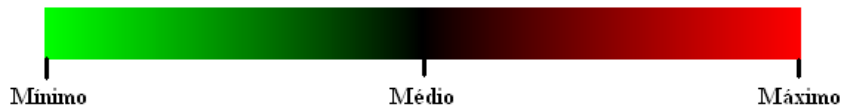
Segue programa em R para definir (gerar) esta paleta de cores:

```
> ## Escala Cor Verde-Preto-Vermelho
# 00FF00 (verde), 000000 (preto), FF0000 (vermelho)
cad = c("F","E","D","C","B","A",9:0)
ff00 = ""; k = 1
for (i in cad) {
  for (j in cad) { ff00[k] = paste(i,j,sep=""); k=k+1
  }}
VerdePreto = paste("#", "00", ff00, "00", sep="")
PretoVermelho = paste("#", ff00[256:1], "0000", sep="")
VerdPrVerm = c (VerdePreto, PretoVermelho)

> ### Legenda de Cor: verde - preto - vermelho
n.cor = length (VerdPrVerm)
plot(0,0, xlim=c(1,n.cor), ylim= c(0,1), col='white')
for (i in 1:n.cor) {
  abline(v=i, col=VerdPrVerm[i], lwd=2)
}
VerdPrVerm
```

Salve a nova figura gerada com a paleta de cores, verde – preto – vermelho na pasta de trabalho: **grafio_vvppee.PNG**. Neste caso específico, sugere-se não maximizar a janela gráfica, para melhor visualização da paleta de cores.

Segue figura com a escala de cores obtidas no R com o programa anterior:



2.8.2 Paleta de cores na escala cinza

De forma análoga à paleta de cores construída na seção anterior, outras podem ser construídas. Nesta seção apresentamos uma nova paleta de cores na escala cinza. Esta legenda de cores inicia no branco, vai até o preto, passando pelo cinza intermediária. Sendo que o branco representa o mínimo, o cinza o médio, e o preto o máximo. No total são 256 cores diferentes.

Segue programa em R para definir (gerar) esta paleta de cores:

```
## Escala Cor Cinza: branco - cinza - preto
escala.cinza = ''
for (i in 0:256) {
  escala.cinza[i] = gray(1-(i/256))
}
> ### Legenda de Cor: branco - cinza - preto
n.cor = length (escala.cinza)
plot(0,0, xlim= c(1,n.cor), ylim= c(0,1), col='white')
for (i in 1:n.cor) {
  abline(v=i,col= escala.cinza[i], lwd=2)
}
escala.cinza
```

Salve a nova figura gerada com a paleta de cores, na escala cinza: **graf11_cinza.PNG**. Neste caso especialmente, sugerimos não maximizar a janela gráfica, caso contrário não fica adequado a visualização da paleta de cores em escala cinza.

Segue figura com a escala de cores obtidas no R com o programa anterior:



Após conhecer várias opções de cores que podem ser usadas nos gráficos, aumenta o número de possibilidades que temos disponíveis na geração de figuras no R. A seguir outros exemplos usando paletas de cores disponíveis no R:

```
> barplot (1:25, col = gray.colors(25))
> pie (rep(1, 25), col = gray.colors(25))
> barplot (1:25, col = cm.colors(25))
> pie (rep(1, 25), col = topo.colors(25))
```

2.9 Construção de Clustering

Em análise estatística multivariada uma técnica muito usada é a de clustering [8,9,10,11]. Esta técnica não supervisionada permite fazer grupos de objetos, os quais podem ser discriminados e agrupados pelas características (variáveis ou propriedades). Várias metodologias existem com esta finalidade, entre elas encontramos: Hierárquica (dendograma), k-Médias (heatmaps) e SOM (Self-Organizing Maps, redes neurais).

Segue um exemplo de construção de um cluster hierárquicos, neste caso vamos usar um conjunto de dados (data-set) disponível no R (**IRIS**).

Para acessar exemplos deste tipo de clustering e sua representação (dendograma) em grafos:

```
> example ( hclust )
```

Exemplo com dados do IRIS:

```
> data(iris) # carrega o data-set iris
> iris # para visualizar seu conteúdo
> str(iris) # estrutura do data set
```

Para selecionar uma amostra aleatória, com **n=25**:

```
> iris.n = nrow (iris) # número de linhas
escolha = sample(1:iris.n, 25, replace = FALSE ) # amostra aleatórias, n=25
iris2 = iris[escolha, 1:4] # somente com 25, escolha aleatória
tipo2 = as.vector (iris $ Species [escolha]) # classificação dos escolhidos
dimnames(iris2)[[1]] = paste(tipo2, dimnames(iris2)[[1]], sep='_')
```

Gerando a matriz de distâncias e construção do dendograma:

```
> res.hc = hclust (dist(iris2) # matriz de distâncias
plot (res.hc) # cluster hierárquico
```

Note como subgrupos (cachos) diferentes são gerados, e os agrupamentos formados dividem de certa forma as espécies: **setosa**, **versicolor** e **virginica**. Salve a figura (dendograma) gerada correspondente ao clustering hierárquico com um subconjunto do banco de dados Iris. Após maximizar a janela gráfica, selecione a pasta de trabalho e nela nome para arquivo de saída: **graf12_hierarq_dendog_iris25.PNG**

Para acessar exemplos de heatmaps, usada para a representação do resultado do agrupamento por k-Médias em matrizes de cores:

```
> example ( heatmap )
```

Exemplo com dados do IRIS:

```
> iris3 = as.matrix(iris2)
> heatmap ( t(iris3) , margins = c(22, 5), col=escala.cinza, scale = 'none')
> heatmap ( t(iris3) , margins = c(22, 5), col= VerdPrVerm, scale = 'none')
```

Esta visualização nos permite uma visualização e interpretação rápida da distribuição dos dados, pois a matriz numérica é transformada em uma escala de cores. Desta forma, podemos analisar com rapidez em que variáveis e/ou observações encontram-se os maiores ou menores valores. Caso **scale = 'none'** não seja usado, as cores serão escaladas nas linhas (variáveis), e só poderão ser comparadas dentro delas e não entre elas.

Salve a figura (heatmap) gerada correspondente ao subconjunto aleatório (n=25) do ban-

co de dados Iris. Após maximizar a janela gráfica, selecione a pasta de trabalho e nela nome para arquivo de saída: **graf13_heatmap_iris25.PNG**

Segue um exemplo de cluster **k-Médias (k=3)** com dados do IRIS. O parâmetro **k**, na metodologia **k-Médias** precisa ser definido a priori, neste caso selecionamos **k=3** (número de species):

```
> nc = 3 # n de grupos
n.obj = 25 # n de objetos
dad.km = iris3
res.km = kmeans (dad.km, nc)
kmeans (dad.km, nc)
ir.rg = range(dad.km) # c(min, max)
dad_ctr = matrix(rep(c(ir.rg[1], mean(ir.rg), ir.rg[2]), n.obj), byrow=
TRUE, nrow=n.obj )
dad2.km = cbind( dad.km, dad_ctr)
n.col2 = length(dimnames(dad2.km) [[2]])
dimnames(dad2.km) [[2]] [n.col2-2] = "__Ctrl_Min__"
dimnames(dad2.km) [[2]] [n.col2-1] = "__Ctrl_Méd__"
dimnames(dad2.km) [[2]] [n.col2] = "__Ctrl_Máx__"
gkm = res.km $ cluster
pdf ( file = 'graf14_km3_iris_cinza.pdf', paper='a4r', width=11, height=8 )
for (ind.c in 1:nc) {
n.sub = sum(gkm == ind.c)
heatmap( dad2.km[ gkm == ind.c, ],
col = escala.cinza,
margins = c(10,25),
Colv = NA, # sem dendograma nas variáveis (vertical)
scale = 'none' # cc os valores são re-escalados nas linhas
)
}
dev.off()
```

Verifique na pasta de trabalho, o arquivo gerado: **graf14_km3_iris_cinza.pdf**

Interprete o resultado obtido na separação realizada

Atividade prática r-6. Gere um novo arquivo PDF com o resultado (heatmaps) do cluster k-médias ($k=3$), similar ao anterior, mas na escala de cor “VerdPrVerm”. O mesmo deve ser armazenado na pasta de trabalho com o nome: “**graf15_km3_iris_vvppee.pdf**”. Salve o programa gerado na pasta de trabalho: **Curso_ASIS_R_ativp_6.r**

2.10 Variáveis aleatórias, geração de amostras

Uma das distribuições [6,7] contínuas mais usadas é a normal (ou gaussiana) padrão, conhecida também pela sua notação: $X \sim N(0,1)$. Algumas das aplicações mais comumente usadas para ela são estudos populacionais de peso (Kg) e altura (cm), entre muitas outras. Na Estatística, é para a distribuição Normal que existem mais resultados e teoremas demonstrados. Muitas vezes, quando os dados estudados não seguem esta distribuição, os mesmos são transformados para este padrão.

Segue um exemplo de como gerar uma amostra no R da distribuição normal padrão, e como visualizar seu formato (histograma e curva de densidade):

```
> n = 1000 # tamanho da amostra
> padrao = rnorm (n, 0, 1)
> padrao
> summary (padrao)
> hist(padrao, nclass=50, freq = FALSE, col = gray.colors(25)[22] )
> points ( density(padrao), type ='l', col ='red', lwd =3)
```

Salve o gráfico gerado anteriormente na pasta de trabalho com o nome: **graf16_No1.jpeg**. Selecione formato **Jpeg** com e qualidade 100%.

Atividade prática r-7. Com base no exemplo anterior, gere o mesmo gráfico com histograma da normal padrão e densidade, mas mude para $n=10000$. Comente qual a principal mudança? Gere um terceiro gráfico que coloque os 2 anteriores, na mesma tela, um do lado do outro, identificando qual é cada um. Salve esse gráfico (com 2 figuras) na pasta de trabalho em formato **Jpeg** com nome: **graf17_No1_mil_10mil.jpeg** e qualidade 100%. Salve o programa gerado na pasta de trabalho: **Curso_ASIS_R_ativp_7.r**

O arquivo **Jpeg** também poder ser gerado e armazenado diretamente por comandos:

```
> jpeg (file= 'graf18_jpeg_cmd.jpeg' )
> hist(padrao, nclass=50, freq = FALSE, col = gray.colors(25)[22] )
> points ( density(padrao), type ='l', col ='red', lwd =3)
> dev.off ()
```

Verifique na pasta de trabalho o arquivo **Jpeg** gerado: **graf18_jpeg_cmd.jpeg**. Comente o gráfico construído, qual a sua principal crítica?

No arquivo **graf18_jpeg_cmd.jpeg**, a figura ficou em escala reduzida. Para melhorar o gráfico anterior, consulte a ajuda e verifique como pode ser melhorado, e a escala ser aumentada?

```
> ? jpeg
```

Segue novo *script*, para o mesmo gráfico (sobre-escrevendo o arquivo **graf18_...**), com escala aumentada:

```
> jpeg (file= 'graf18_jpeg_cmd.jpeg' , width = 1024, height = 768 )
> hist(padrao, nclass=50, freq = FALSE, col = gray.colors(25)[22] )
> points ( density(padrao), type ='l', col ='red', lwd =3)
> dev.off ()
```

A escala ficou melhor?

Para a amostra gerada anteriormente, podemos construir um gráfico de dispersão (scatterplot), destacando a média e os desvios padrão:

```
> plot( padrao )
> abline ( h= mean(padrao), col =' dark green', lwd =3)
> abline ( h= mean(padrao) +c(-1,1)*2*sd(padrao), col ='red', lwd =3)
```

Salve o gráfico gerado na pasta de trabalho em formato **Jpeg** com nome: **graf19_No1_dp.jpeg**

Uma das distribuições discretas mais usadas é a binomial, conhecida também pela sua

notação: $X \sim \text{Binomial}(n,p)$. Esta distribuição é usada em epidemiologia, por exemplo, em estudos de prevalência de determinada doença. Interpretação: número total de sucessos nos n experimentos, onde cada um teve probabilidade “ p ” de ocorrer.

Seguem comandos em R para gerar uma amostra da binomial, e como visualizar seu formato (histograma e curva de densidade):

```
> N = 1000 # tamanho da amostra
> n = 200 # exemplo em epidemiologia com 200 participantes
> p = .05 # Prevalência para uma determinada doença de 5%
> x = rbinom (N, size = n, p = p) # amostra aleatória
> x
> summary (x)
> hist(x, nclass=50, freq = FALSE, col = gray.colors(25) [22] )
> points ( density(x), type ='l', col ='red', lwd =3)
```

Atividade prática r-8. Interprete o resultado. Armazene o gráfico gerado na pasta de trabalho do curso, em formato **TIFF**, com nome: **graf2o_Binom_hist.tiff**

Após termos a amostra, podemos comparar as estimativas (amostra) vs parâmetros (população):

```
> print('media observada vs populacional'); print(mean(x)); print(n*p)
```

De forma análoga, podemos estudar a variância, amostral vs teórica (população):

```
> print('variancia amostral vs teorica'); print(var(x)); print(n*p*(1-p))
```

Atividade prática r-9. Caso queira que a média e variância observada (amostral) seja mais próxima da teórica (populacional) o que pode ser feito? Gere uma nova sequência de comandos e apresente o resultado. Para esta atividade faça uso da ferramenta **Tinn-R** ou **Notepad++**, criando um novo arquivo de texto (**Curso_ASIS_R_ativp_9.r**) na pasta de trabalho, contendo os novos comandos necessários. Dica: para fazer a comparação crie **x1** e **x2** com $n=100$ e 10000 , respectivamente. Para executar os comandos, após salvar o arquivo (**.r**), execute o seguinte comando no R:

```
> source ( "Curso_ASIS_R_ativp_9.r" ) # executa script no R
```


Caso queira repetir o teste anterior, rode novamente o comando anterior.

No R encontram-se desenvolvidos um grande conjunto de distribuições para variáveis aleatórias, tanto discretas como contínuas. Para cada uma delas é possível gerar amostras, consultar seus quantis, densidade e probabilidade acumulada. Para acessar outras distribuições de interesse:

- Uniforme discreta: `sample`
- Uniforme contínua: `runif`
- Poisson (discreta): `rpois`
- Exponencial: `rexp`
- Normal multivariada: `mvrnorm`
- Log Normal: `rlnorm`
- Weibull: `rweibull`

Resoluções do Módulo 2

Análise de Dados dos Sistemas de Informação em Saúde

Nesta seção apresentamos a resolução das atividades do Módulo de “Análise de Dados dos Sistemas de Informação em Saúde”.

Atividade 1

Questão A – Calcule a taxa geral ou bruta de mortalidade (**TBM**) para os Estados do Pará, Bahia, Espírito Santo e Rio Grande do Sul em 2010, utilizando os dados de óbitos disponíveis no SIM e dados de população do Censo 2010. O que essas taxas indicam em relação à situação de saúde dos Estados?

Após a extração dos dados indicada neste exercício, da fonte do Ministério da Saúde (DATASUS, <www.datasus.gov.br>), obtém-se os seguintes dados de óbito e população para o ano de 2010:

Período:2010

UF	Óbitos	População
Pará	31,600	7,581,051
Bahia	76,337	14,016,906
Espírito Santo	21,205	3,514,952
Rio Grande do Sul	77,985	10,693,929

Fonte: MS/SVS/DASIS - Sistema de Informações sobre Mortalidade - SIM, IBGE - Censos Demográficos.

Segue *script* em R para gerar a Taxa Geral ou Bruta de Mortalidade (TGM) por 1000, para as UF Pará (PA), Bahia (BA), Espírito Santo (ES) e Rio Grande do Sul (RS):

Questão A

```
# define um vetor com o nome das UF.
```

```

# Para string pode usar aspas simples ou duplas
uf = c( 'Pará', 'Bahia', "Espírito Santo", 'Rio Grande do Sul')
# define vetor com o nº de óbitos por uf
obitos = c( 31600, 76337, 21205, 77985)
# define vetor com população por uf
pop = c( 7581051, 14016906, 3514952, 10693929)

# calcula TGM (taxa geral ou bruta de mortalidade) por 1000
# com 1 dígito decimal
TGM = round( obitos / pop * 1000, digits=1)

# cria um data-frame (tabela) com as variáveis de interesse
resultado = data.frame (uf, obitos, pop, TGM)
# altera os nomes das colunas
names(resultado) = c("Região/UF", "Óbitos", "População", "TGM (por 100)")
# visualiza o conteúdo da variável "resultado"
resultado

```

Após executar os comandos anteriores no R, obtemos a saída abaixo:

```

> resultado
      Região/UF Óbitos População TGM (por 100)
1          Pará  31600   7581051         4.2
2          Bahia 76337  14016906         5.4
3  Espírito Santo 21205   3514952         6.0
4 Rio Grande do Sul 77985  10693929         7.3
>

```

Questão B – Calcule as **Taxas de Mortalidade Específicas por Idade (mx)** para o Espírito Santo e Rio Grande do Sul em 2010 e a razão entre as taxas (considere para isso os seguintes grupos etários: 0-14 anos, 15-39 anos, 40-64 anos, 65 e +). O que esses dados indicam? Compare com as TGM do exercício anterior e avalie porque ocorrem diferenças.

Seguem os dados de mortalidade para as UF ES e RS no ano de 2010:

Óbitos p/Residênc por Região/UF e Faixa Etária det - Período:2010

UF	0 a 14 anos	15 a 39 anos	40 a 64 anos	65 anos e mais	Total
Espírito Santo	935	3,299	6,056	10,872	21,205
Rio Grande do Sul	2,163	6,459	22,043	47,247	77,985

Fonte: MS/SVS/DASIS - Sistema de Informações sobre Mortalidade - SIM

Seguem os dados de população (n e %) para as UF ES e RS no ano de 2010:

População residente por UF e Faixa Etária detalhada - Período:2010

UF		0 a 14 anos	15 a 39 anos	40 a 64 anos	65 anos e mais	Total
Espírito Santo	n	811,642	1,505,836	947,857	249,617	3,514,952
	%	23.1%	42.8%	27.0%	7.1%	100.0%
Rio Grande do Sul	n	2,229,504	4,194,052	3,275,760	994,613	10,693,929
	%	20.8%	39.2%	30.6%	9.3%	100.0%

Fonte: IBGE - Censos Demográficos

Segue *script* em R para gerar as Taxas de Mortalidade Específicas por idade (mx) por 1000, para as UF ES e RS:

Questão B

```
# define um vetor com o nome das UF, e outro com os das faixas-etárias
```

```
uf = c( "Espírito Santo", "Rio Grande do Sul")
```

```
fxs_etarias = c( '0 a 14 anos', '15 a 39 anos', '40 a 64 anos', '65
```

```

anos e mais', 'Total')
# define vetor com o nº de óbitos por UF
obitos_ES = c(935,3299,6056,10872,21205)
obitos_RGS = c(2163,6459,22043,47247,77985)
# define vetor com população por uf
pop_ES = c(811642,1505836,947857,249617,3514952)
pop_RGS = c(2229504,4194052,3275760,994613,10693929)

# calcula TGM (taxa geral ou bruta de mortalidade) por 1000,
# com 2 dígito decimal
TGM_ES = round(obitos_ES / pop_ES * 1000, digits=2)
TGM_RGS = round(obitos_RGS / pop_RGS * 1000, digits=2)
# calcula a razão entre as 2 UF
razao_RGS_ES = round(TGM_RGS / TGM_ES, digits=2)

nome_col_lin = list() # cria uma lista vazia
# gera nomes para as linhas
nome_col_lin [[1]] = c(uf, 'Razão RGS/ES')
nome_col_lin [[2]] = fxs_etarias # nome para as colunas

# cria uma matriz(tabela) com as variáveis de interesse:
# linhas= UF e colunas= faixas-etárias
resultado = matrix ( c(TGM_ES, TGM_RGS, razao_RGS_ES),
nrow = 3, ncol = 5, byrow = TRUE,
dimnames = nome_col_lin )
resultado

```

Após executar os comandos anteriores no R, obtemos a saída abaixo:

```

> resultado
           0 a 14 anos 15 a 39 anos 40 a 64 anos 65 anos e mais Total
Espírito Santo      1.15      2.19      6.39      43.55 6.03
Rio Grande do Sul   0.97      1.54      6.73      47.50 7.29
Razão RGS/ES        0.84      0.70      1.05      1.09 1.21
>

```

Questão D – Calcule as **TMI (total e por componentes)** em 2010 utilizando os dados do SIM e SINASC para os mesmos Estados da questão A. O que essas taxas indicam? Quais as limitações principais de sua análise utilizando esses cálculos?

Seguem os dados de óbitos em menores de 1 ano e nascidos vivos para as UF do PA, ES e RS no ano de 2010:

Óbitos p/Residênc por UF e Faixa Etária det - Período:2010

UF	0 a 6 dias	7 a 27 dias	Neonatal	28 a 364 dias	Menor 1 ano (ign)	Total Menor 1 ano	NV
Pará	1,442	354	1,796	734	3	2,533	140,687
Espírito Santo	313	126	439	178	-	617	51,853
Rio Grande do Sul	718	276	994	498	-	1,492	133,243

Fonte: MS/SVS/DASIS - Sistema de Informações sobre Mortalidade - SIM e Sistema de Informações sobre Nascidos Vivos - SINASC

Seguem comandos em R para gerar as Taxas de Mortalidade: Infantil, Neonatal e Pós-Neonatal para as UF do Pará, Espírito Santo e Rio Grande do Sul:

Questão D

```
# define um vetor com o nome das UF
uf = c('Pará', 'Espírito Santo', 'Rio Grande do Sul')
tx_mort = c('TMI', 'Neonatal', 'Pós-Neontal')
# define vetor com o n° de óbitos por uf
obitos_PA = c(2533,1796,734)
obitos_ES = c(617,439,178)
obitos_RS = c (1492,994,498)
# define vetor com população por uf: nascidos vivos
pop_NV = c(140687,51853,133243)
nv_PA = pop_NV [1] # acessando o 1° elemento do vetor
nv_ES = pop_NV [2] # acessando o 2° elemento do vetor
```

```

nv_RS = pop_NV [3] # acessando o 3° elemento do vetor

# calcula Taxas de Mortalidade por 1000, com 1 dígito decimal
TM_PA = round(obitos_PA / nv_PA * 1000, digits=1)
TM_ES = round(obitos_ES / nv_ES * 1000, digits=1)
TM_RS = round(obitos_RS / nv_RS * 1000, digits=1)

nome_col_lin = list() # cria uma lista vazia
nome_col_lin [[1]] = uf # gera nomes para as linhas
nome_col_lin [[2]] = tx_mort # nome para as colunas

# cria uma matriz(tabela) com as variáveis de interesse:
# linhas= UF e colunas= taxas de mortalidade
resultado = matrix (c(TM_PA, TM_ES, TM_RS),
nrow = 3, ncol = 3, byrow = TRUE,
dimnames = nome_col_lin )
resultado

```

Após executar o programa anterior no R, obtemos a seguinte saída para as taxas de mortalidade:

```

> resultado
           TMI Neonatal Pós-Neontal
Pará      18.0      12.8      5.2
Espírito Santo  11.9      8.5      3.4
Rio Grande do Sul 11.2      7.5      3.7
>

```

Questão E – Calcule a **Taxa (Razão) de Mortalidade Materna (RMM)** para os Estados do Pará, Espírito Santo e Rio Grande do Sul em 2010. Nesse caso, não considere somente as causas maternas do Cap. XV, mas todos os óbitos maternos. O que essas RMM indicam? Compare e avalie as diferenças e limitações de sua análise. Compare com as RMM de 2010 disponíveis no IDB-2011 (Indicador Co3). Justifique possíveis diferenças.

Este exercício pode ser resolvido no R de forma análoga ao anterior, apresentado na questão D da Atividade 1.

Atividade 2

Questão A – Calcule a mortalidade proporcional por homicídios e as taxas de mortalidade por homicídios para os Estados do Pará, Espírito Santo e São Paulo em 2010. O que esses dados indicam?

Este exercício pode ser resolvido no R de forma análoga à questão D, apresentado na Atividade 1.

Atividade 3

Questão A - Calcule as taxas de mortalidade padronizadas por idade (TGMP) para o Espírito Santo e Rio Grande do Sul em 2010, utilizando como população-padrão a soma das duas populações. Para facilitar os cálculos, utilize o quadro abaixo.

Segue uma tabela com os dados para solucionar este exercício e resultados preliminares para as UF de ES e RS em 2010:

Idade	População padrão (a)	Taxa idade (mx) por 100.000 ES (b)	Óbitos esperados ES (ab/100000)	Taxa idade (mx) por 100000 RS (c)	Óbitos esperados RS (ac/100000)
0 a 14 anos	3,041,146	115.20	3,503	97.02	2,950
15 a 39 anos	5,699,888	219.08	12,487	154.00	8,778
40 a 64 anos	4,223,617	638.91	26,985	672.91	28,421
65 e +	1,244,230	4,355.47	54,192	4,750.29	59,105
Soma	14,208,881		97,168		99,254

Fonte: MS/SVS/DASIS - Sistema de Informações sobre Mortalidade - SIM, IBGE - Censos Demográficos

Neste exercício é empregado a definição e chamada a função no R. Segue comandos usados para gerar os resultados:

Questão A

```
# define cte que será usada neste exercício, cte=base
```



```

cte.ag = 100000
# define um vetor com o nome das UF
uf = c( "Espírito Santo", 'Rio Grande do Sul')
fxs_etarias = c( '0 a 14 anos', '15 a 39 anos', '40 a 64 anos', '65
anos e mais', 'Total')
# define vetor com o nº de óbitos por uf
obitos_ES = c( 935, 3299, 6056, 10872)
obitos_RS = c( 2163, 6459, 22043, 47247)
# define vetor com população por uf
pop_ES = c( 811642, 1505836, 947857, 249617)
pop_RS = c( 2229504, 4194052, 3275760, 994613 )
pop_ES_RS = pop_ES + pop_RS # soma de vetores

# define função para calculo da TGM
# calcula TGM (taxa geral ou bruta de mortalidade) por cte,
# com dig dígito decimal
# o padrão para cte=base=100,000, e dig=nº de digitos=2
tgm = function (ob, pop, cte=cte.ag, dig=2) {
  round( ob / pop * cte, dig)
}
# calcula TGM por chamada à função
TGM_ES = tgm( obitos_ES, pop_ES, cte.ag, 2)
# quando o parâmetro não é especificado, pega o valor padrão

TGM_RS = tgm( obitos_RS, pop_RS)

# óbitos esperados
# valor inteiro, sem casas decimais
ob.esp.ES = round (pop_ES_RS * TGM_ES / cte.ag, 0)
ob.esp.RS = round ( pop_ES_RS * TGM_RS / cte.ag, 0)

# população total
pop.tot = sum(pop_ES_RS) # soma o conteúdo de um vetor
# óbitos esperados totais por UF

```

```
ob.esp.tot_ES <- sum(ob.esp.ES) # o comando "<-" é equivalente com "="
ob.esp.tot_RS <- sum(ob.esp.RS)
```

```
nome_col_lin = list() # cria uma lista vazia
nome_col_lin [[1]] = fxs_etarias
# nome para as colunas
nome_col_lin [[2]] = c('População padrão',
'Taxa idade ES', 'Óbitos esperados ES',
'Taxa idadeRS', 'Óbitos esperados RS')
```

```
result = matrix ( c( c(pop_ES_RS, pop.tot),
c(TGM_ES,0 ),
c(ob.esp.ES,ob.esp.tot_ES ),
c(TGM_RS,0 ),
c(ob.esp.RS,ob.esp.tot_RS )) ,
nrow = 5, ncol = 5, byrow = FALSE,
dimnames = nome_col_lin )
```

```
# Taxa de mortalidade padronizada por idade
Tx.mort.pad.idade_ES = round(ob.esp.tot_ES / pop.tot * 1000, 1)
Tx.mort.pad.idade_RS = round(ob.esp.tot_RS / pop.tot * 1000, 1)
```

```
result
Tx.mort.pad.idade_ES
Tx.mort.pad.idade_RS
```

Após executar o programa anterior no R, obtemos a seguinte saída:

```
> result
      População padrão Taxa idade ES Óbitos esperados ES Taxa idadeRS Óbitos esperados RS
0 a 14 anos           3041146      115.20              3503          97.02           2951
15 a 39 anos          5699888      219.08             12487         154.00           8778
40 a 64 anos          4223617      638.91             26985         672.91          28421
65 anos e mais        1244230      4355.47            54192        4750.29          59105
Total                 14208881
      Tx.mort.pad.idade_ES
[1] 6.8
      Tx.mort.pad.idade_RS
[1] 7
```

Atividade 5

Questão B – Utilize seus dados das tabulações básicas (Exercícios 1) e calcule as taxas de mortalidade em 2010 por diabete melito para os mesmos estados do exercício 5.1. Compare o risco de morte por essas doenças nos Estados e interprete. Avalie as possíveis limitações de sua análise.

Este exercício pode ser resolvido no R de forma análoga ao exercício 1.1, apresentado na seção 2.1.

Questão C – Calcule as taxas de mortalidade específicas por idade e sexo de 2010 para Pará e Rio Grande do Sul. Prepare um gráfico do logaritmo das taxas específicas de mortalidade por idade para cada sexo. Analise e discuta seus resultados.

Certifique-se que os dados secundários necessários foram baixados do site do DATASUS <www.datasus.gov.br> e armazenados na pasta de dados (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

O passo a passo para obter os dados foi descrito antes do exercício 5.3 do caderno com os exercícios propostos no Módulo “Análise de Sistema de Informação em Saúde”. Para organizar o trabalho sugerimos usar os seguintes nomes para os arquivos CSV:

1. Ob_res_fxEtaria_det_UF_2010_sexM.csv

Para Óbitos por residência para Unidade Federativa (linha), faixa etária “det” (coluna), ano=2010, sexo Masculino.

2. Ob_res_fxEtaria_det_UF_2010_sexF.csv

Para Óbitos por Unidade Federativa (linha), faixa etária “det” (coluna), ano=2010, sexo Feminino.

3. Pop_res_FxEtaria_detalhada_UF_Sex-M_2010.csv

Para População residente para Unidade Federativa (linha), faixa etária “detalhada” (coluna), ano=2010, sexo Masculino.

4. Pop_res_FxEtaria_detalhada_UF_Sex-F_2010.csv

Para População residente para Unidade Federativa (linha), faixa etária “detalhada” (coluna), ano=2010, sexo Feminino.

Neste exercício é empregada a definição e chamada a função no R. Foi trabalhado com acesso a elementos de vetores e matrizes. Utilizou-se o comando **apply** para o trabalho com matrizes. E também se geraram gráficos.

Para solucionar este exercício será necessária a leitura de arquivos texto em formato CSV no R. Segue comandos usados para a configuração e geração de gráficos a partir dos arquivos CSV extraídos do DATASUS:

Questão C

```
# define a pasta de trabalho, que precisa ser previamente criada
pasta_dados = 'C:/Curso_EAD_ASA/'

# o R se posiciona na pasta definida
setwd (pasta_dados)

# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()

# define uma função que carrega arquivos em formato CSV
# com ela vamos poder carregar óbitos por sexo, e população por sexo
# são definidos valores padrão para os parâmetros
# menos para o nome do arquivo
read.ob.pop = function (arquivo, separadorCol=';',
cabecalho=TRUE, factors=FALSE,
sepDecimal='.', pulaLinhaInicio=4,
NoLinhaCarrega=27) {
  read.table (file = arquivo, sep=separadorCol, header=cabecalho,
stringsAsFactors=factors, dec=sepDecimal,
```

```

        skip=pulaLinhaInicio, nrows = NoLinhaCarrega)
    }

# para chamar a função "read.ob.pop"
# podemos ou não especificar os parâmetro
# pois eles estão pré-definidos na definição da função
# a leitura pula 4 linhas do cabeçalho do arquivo,
# antes dos nomes das colunas
# e depois dos nomes das colunas, carrega 27 linhas das UF (com DF)
Ob_M=read.ob.pop ( 'Ob_res_fxEtaria_det_UF_2010_sexM.csv', ';', TRUE,
FALSE, ', ', 4, 27)
Ob_F = read.ob.pop ( 'Ob_res_fxEtaria_det_UF_2010_sexF.csv')
Pop_M = read.ob.pop ( 'Pop_res_FxEtaria_detalhada_UF_Sex-M_2010.csv')
Pop_F = read.ob.pop ( 'Pop_res_FxEtaria_detalhada_UF_Sex-F_2010.csv')

# função para substituir '-' por '0' se houver
tira_trazo = function(dad, tra='-', troca=0) {
    ind = which (dad == tra)
    if ( length(ind) >0) { dad[ind]=troca }
    dad # resultado retornado pela função
}

# coluna "Menor.1.ano..ign."
# para determinadas UF é = '-'
# [,5] se refere aos elementos da 5ta coluna, todas as linhas
Ob_M[,5] = as.integer( tira_trazo(Ob_M[,5]) )
Ob_F[,5] = as.integer( tira_trazo(Ob_F[,5]) )

# índice das colunas de interesse
# 18 colunas, sem as 4 primeiras
# a 5ta coluna, que agora é a primeira, depois será alterada
ind_ob = (1:18) +4

# primeiro extrai os nomes das colunas

```

```

# e depois filtra os índice de interesse
fxs_etarias = names(Ob_M) [ind_ob]
fxs_etarias [1] = 'Menor.1.ano' # somente a 1ª precisa ser trocada
# tirando 'X' no início do nome
fxs_etarias [2:18] = substring(fxs_etarias [2:18], 2)
data.frame(fxs_etarias)

# vetor com lista dos nomes das UF
uf.nomes = Ob_M [,1]
data.frame(uf.nomes)

# verificando que as UF tenham a mesma ordenação nos 4 CSV
# por isso as 3 comparações abaixo precisam ser "TRUE"
sum(Ob_M [,1] == Ob_F [,1]) == 27 # óbitos masculinos e femininos
sum(Pop_M [,1] == Pop_F [,1]) == 27 # população de ambos sexos
sum(Ob_M [,1] == Pop_M [,1]) == 27 # óbitos vs população

# óbitos masculinos nas faixas etárias de interesse
dad.ob.m = Ob_M [ , ind_ob ]
# a função apply(x,1,sum) soma colunas de uma matriz
dad.ob.m[,1] = apply(Ob_M[,2:5], 1, sum) # óbitos Menor.1.ano
names(dad.ob.m) = fxs_etarias # nome das colunas

# é feito o mesmo procedimento para os óbitos femininos
dad.ob.f = Ob_F [ , ind_ob ]
dad.ob.f[,1] = apply(Ob_F[,2:5], 1, sum)
names(dad.ob.f) = fxs_etarias

# faixas etárias desejadas (dos óbitos)
data.frame(names( dad.ob.m))
# faixas etárias dos dados de população
data.frame(names( Pop_M))

# função que agupa a população residente em faixas etárias

```

```

# (idem aos óbitos)
pop.agrupa.idade = function (dad, modelo){
  # uso os mesmo nomes das colunas e inicializo a matriz em 0
  dad2 = modelo * 0
  # pega a 2ª coluna e a guarda na 1ª
  dad2[,1] = dad[,2] # Menor.1.ano
  # soma as col's 3 a 6 e o guarda na 2ª
  dad2[,2] = apply(dad[,3:6],1, sum) # 1.a.4.anos
  dad2[,3] = apply(dad[,7:11],1, sum) # 5.a.9.anos
  dad2[,4] = apply(dad[,12:16],1, sum) # 10.a.14.anos
  dad2[,5] = apply(dad[,17:21],1, sum) # 15.a.19.anos
  dad2[,6:18] = dad[,22:34] # de "20.a.24.anos" até "80.anos.e.mais"
  dad2 # resultado da função
}

# re define as faixas etárias, para iguais aos dos óbitos
dad.pop.m = pop.agrupa.idade (Pop_M, dad.ob.m)
dad.pop.f = pop.agrupa.idade (Pop_F, dad.ob.f)

# taxas de mortalidade específicas
# por UF, para todas as UF separadamente
# por sexo
tmi_m = dad.ob.m / dad.pop.m * 100000 # Masculino
tmi_f = dad.ob.f / dad.pop.f * 100000 # Feminino

# taxas de mortalidade específicas
tmi_tot = (dad.ob.m + dad.ob.f) / (dad.pop.m + dad.pop.f) * 100000

# procura a linha de uma determinada UF
ind.PA = which( uf.nomes == 'Pará' )
ind.RS = which( uf.nomes == 'Rio Grande do Sul' )

# UF PA, masculino e feminino
tmi_PA_m = as.matrix (tmi_m) [ind.PA ,] # pega uma linha da matriz

```

```

tmi_PA_f = as.matrix (tmi_f) [ind.PA ,]

# UF RS, masculino e feminino
tmi_RS_m = as.matrix (tmi_m) [ind.RS ,]
tmi_RS_f = as.matrix (tmi_f) [ind.RS ,]

# UF PA e RS, masculino "+" feminino
tmi_PA_tot = as.matrix (tmi_tot) [ind.PA ,]
tmi_RS_tot = as.matrix (tmi_tot) [ind.RS ,]

# configuração do gráfico
par.mar0 = c(5, 4, 4, 2) + 0.1 # padrão
# com margem inferior maior,
# para ter espaço para os nomes das faixas etárias
par.mar2 = c(8, 4, 4, 2) + 0.1

# função para gerar os gráficos com 2 linhas (UF)
gera.graf2 = function (
  dad1, dad2, y.lim, y.lab, tit, leg, leg.pos, ya )
# dad1 - vetor com dado para 1ª linha
# dad2 - vetor com dado para 2ª linha
# y.lim - vetor com mínimo e máximo para o eixo Y
# y.lab - string com nome para o eixo Y
# tit - string com o nome do gráfico
# leg - vetor com legenda do gráfico
# leg.pos - posição no gráfico onde será colocada a legenda
# ya - nome da legenda que será escrita no eixo X
{
  # configura gráfico
  par(col.axis='white', las=3, mar = par.mar2)
  # gera a primeira linha na cor azul, espessura n° 3
  plot(dad1, type='l', col='blue', lwd=3, ylim=c(0,y.lim),
    xlab='', ylab=y.lab,
    x.axis=NULL

```



```

)
#gera a segunda linha
points(dad2, type='l', col='red', lwd=3)
abline(h=ya) # gera o grid de linhas horizontais
title(tit) # nome do gráfico
# gera a caixa dentro do gráfico com a legenda
legend(leg.pos[1], leg.pos[2],leg,col=c('blue','red'), lwd=3,
lty=1 )

# faz a legenda no eixo "Y"
par(col.axis='black', las=3, mar = par.mar2 )
n.axis <- length(ya)
for (i in 1:n.axis) {
  axis(2, ya[i], ya[i])
}

# legenda no eixo "X"
x.axis <- fxs_etarias
n.axis <- length(x.axis)
for (i in 1:n.axis) {
  axis(1, i, x.axis[i])
}} # fim da função que gera o gráfico

desc.tx.mort = 'Taxa de Mortalidade Específica x 100.000'
legenda_uf2 = c('PA - Pará', 'RS - Rio Grande do Sul')
seq212 = seq(2,12,2)*1000 # grid horizontal

### Cada bloco abaixo gera um determinado gráfico ###
### Executar cada um deles separadamente ###

# TxMort PA vs RS, sexo Masculino
gera.graf2(tmi_PA_m, tmi_RS_m, 12000, desc.tx.mort,
paste(desc.tx.mort,'\n Sexo Masculino'),
legenda_uf2, c(5,11900), seq212 )

```

```

# TxMort PA vs RS, sexo Feminino
gera.graf2(tmi_PA_f, tmi_RS_f, 12000, desc.tx.mort,
  paste(desc.tx.mort, '\n Sexo Feminino'),
  legenda_uf2, c(5,11900), seq212)

# TMI PA vs RS, sexo (Masculino + Feminino)
gera.graf2(tmi_PA_tot, tmi_RS_tot, 12000, desc.tx.mort,
  paste(desc.tx.mort, '\n Sexo (Masculino + Feminino)'),
  legenda_uf2, c(5,11900), seq212)

# TMI PA vs RS, sexo Masculino, na escala LOG
gera.graf2(log(tmi_PA_m), log(tmi_RS_m), 10,
  paste('Log(', desc.tx.mort, ')'),
  paste('Log(', desc.tx.mort, ') \n Sexo Masculino '),
  legenda_uf2, c(2,9.8), seq(0,10,2) )

# TMI PA vs RS, sexo Feminino, na escala LOG
gera.graf2(log(tmi_PA_f), log(tmi_RS_f), 10,
  paste('Log(', desc.tx.mort, ')'),
  paste('Log(', desc.tx.mort, ') \n Sexo Feminino '),
  legenda_uf2, c(2,9.8), seq(0,10,2) )

# TMI PA vs RS, sexo (Masculino + Feminino), na escala LOG
gera.graf2(log(tmi_PA_tot), log(tmi_RS_tot), 10,
  paste('Log(', desc.tx.mort, ')'),
  paste('Log(', desc.tx.mort, ') \n Sexo (Masculino + Feminino) '),
  legenda_uf2, c(2,9.8), seq(0,10,2) )

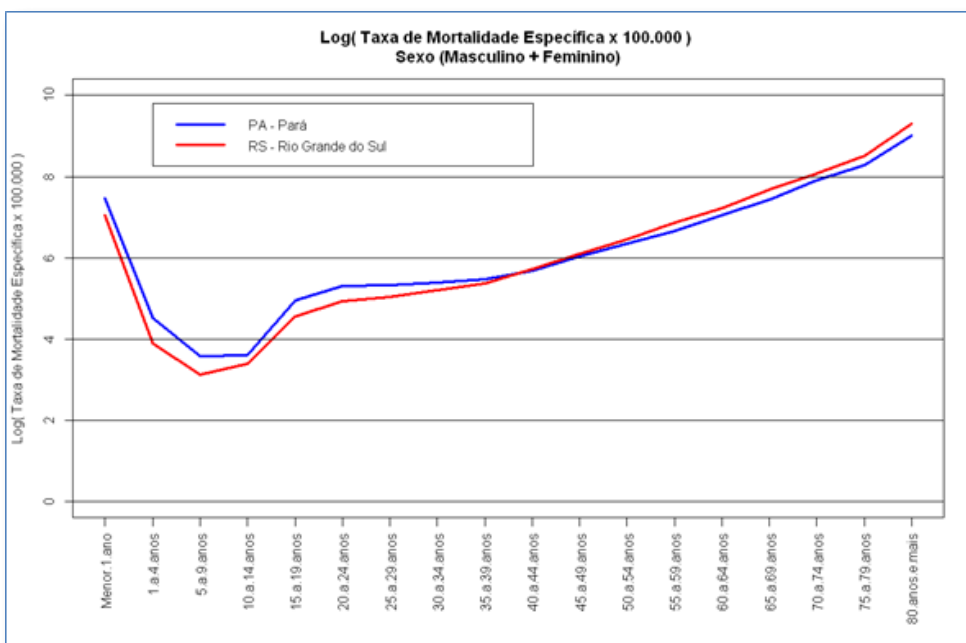
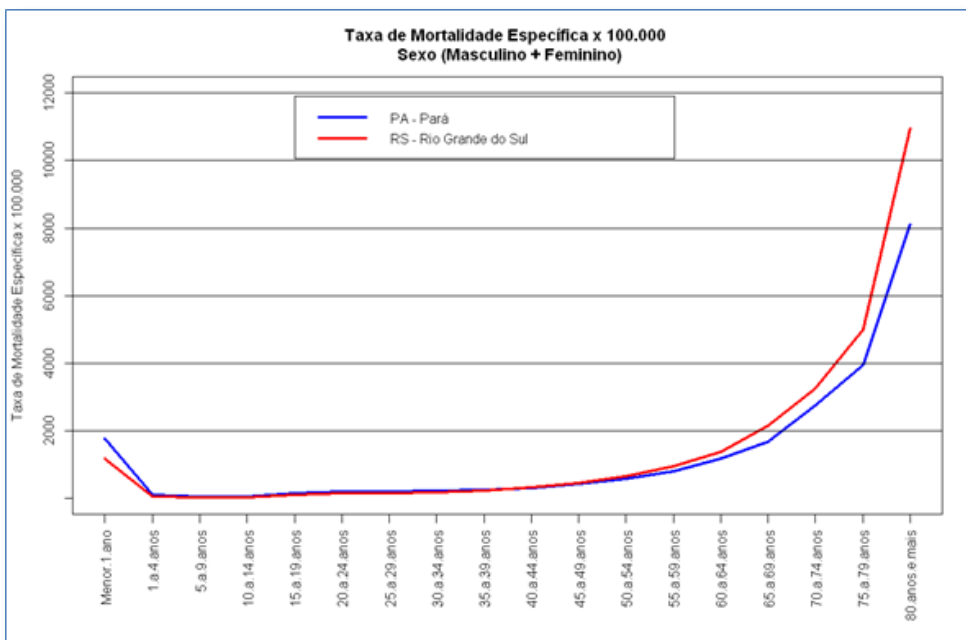
# por último execute este comando
# para restabelecer a geração padrão de gráficos
# par(col.axis='black', las=1, mar = par.mar0 )

```

Quando o gráfico for gerado no R, o mesmo pode ser armazenado no micro da seguinte forma:

1. Deixar a tela do R que apresenta o gráfico ativa
2. Ir ao menu do R em “Arquivo” na opção “Salvar como”
3. Selecionar o formato desejado, exemplo: Png...

Segue dois exemplos dos gráficos gerados em R:



Questão D – Calcule a razão por sexo das taxas de mortalidade específicas por idade para Pará, Bahia e Rio Grande do Sul em 2010. Analise e discuta seus resultados.

Para solucionar este exercício podemos usar o mesmo programa em R desenvolvido na Questão C da Atividade 5. Se você executou o exercício anterior e o R continua aberto não é necessário rodar os comandos novamente. Caso contrário, para que o R carregue os dados necessários execute os comandos da Questão C até a linha:

```
# função para gerar os gráficos com 2 linhas (UF)
```

Desta forma o R já tem em memória os dados de óbitos e de população carregados desde os arquivos CSV e as taxas de mortalidade calculadas para as UF PA e RS. Segue programa em R para preparar as taxas para a UF BA e redefinir, a partir da já construída na Questão C, uma nova função para gerar o gráfico com três linhas:

Questão D

```
# executar os mesmos comandos do exercício 5.3 até a linha:
```

```
# função para gerar os gráficos com 2 linhas (UF)
```

```
# é necessário preparar os dados da BA - Bahia
```

```
# procura a linha de uma determinada UF
```

```
ind.BA = which( uf.nomes == 'Bahia' )
```

```
# UF BA, masculino e feminino
```

```
tmi_BA_m = as.matrix (tmi_m) [ind.BA ,] # pega uma linha da matriz
```

```
tmi_BA_f = as.matrix (tmi_f) [ind.BA ,]
```

```
# UF BA, masculino "+" feminino
```

```
tmi_BA_tot = as.matrix (tmi_tot) [ind.BA ,]
```

```
# função para gerar os gráficos, agora com 3 linhas (UF)
```

```
gera.graf3l = function(
```

```

dad1, dad2, dad3, y.lim, y.lab, tit, leg, leg.pos, ya, fe )
# dad1, dad2 e dad3 - vetor com os dados das linhas 1ª, 2ª e 3ª
respectivamente.
# y.lim - vetor com mínimo e máximo para o eixo Y
# y.lab - legenda para o eixo Y
# tit - variável string com o nome do gráfico
# leg - legenda
# leg.pos - vetor (x,y) com posição da legenda no gráfico
# ya - legenda para eixo Y
# fe - vetor com os nomes das faixas etárias, para legenda do eixo X
{
# configura gráfico
par(col.axis='white', las=3, mar = par.mar2)
# gera a primeira linha na cor azul, espessura nº 3
plot(dad1, type='l', col='blue', lwd=3, ylim=c(0,y.lim),
xlab='', ylab=y.lab,
x.axis=NULL
)
# gera a segunda linha
points(dad2, type='l', col='red', lwd=3)
# gera a terceira linha
points(dad3, type='l', col='dark green', lwd=3)
abline(h=ya) # gera o grid de linhas horizontais
title(tit) # nome do gráfico
# gera a caixa dentro do gráfico com a legenda
legend(leg.pos[1], leg.pos[2], leg,
col=c('blue','red', 'dark green'), lwd=3, lty=1 )

# faz a legenda no eixo "Y"
par(col.axis='black', las=3, mar = par.mar2 )
n.axis <- length(ya)
for (i in 1:n.axis) {
axis(2, ya[i], ya[i])
}

```

```

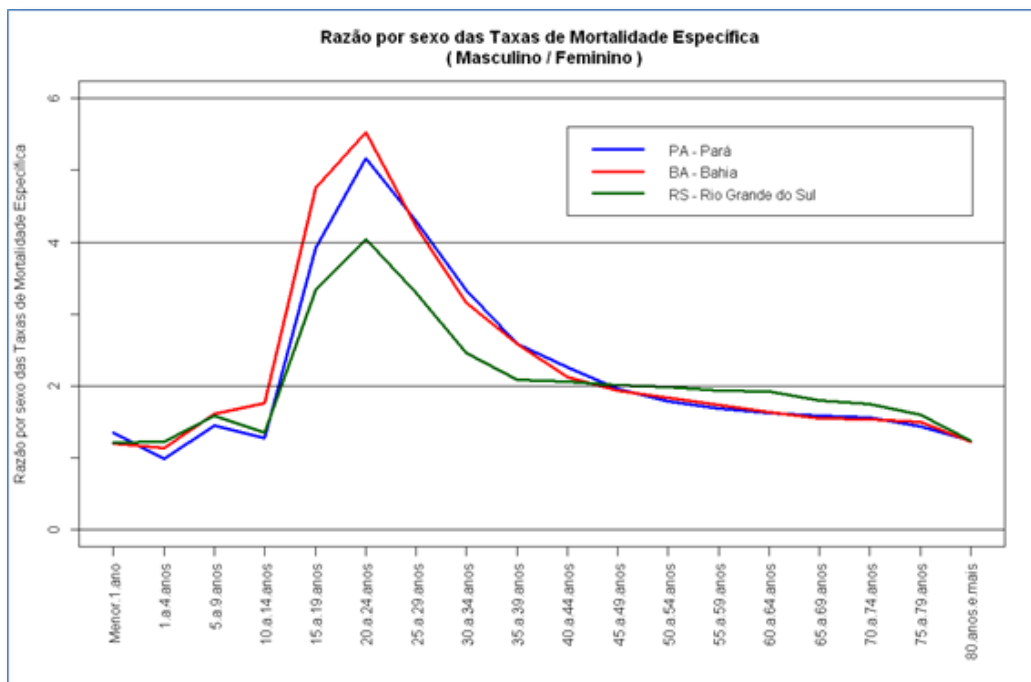
# legenda no eixo "X"
x.axis <- fe
n.axis <- length(x.axis)
for (i in 1:n.axis) {
axis(1, i, x.axis[i])
}} # fim da função que gera o gráfico

tx.mort.raz.sex = 'Razão por sexo das Taxa de Mortalidade Específica'
legenda_uf3 = c('PA - Pará', 'BA - Bahia', 'RS - Rio Grande do Sul')

# Razão das TMI PA, BA e RS, por sexo (Masculino / Feminino)
gera.graf31(tmi_PA_m / tmi_PA_f,
tmi_BA_m / tmi_BA_f,
tmi_RS_m / tmi_RS_f,
6, tx.mort.raz.sex,
paste (tx.mort.raz.sex, '\n ( Masculino / Feminino )'),
legenda_uf3, c(10,5.6), seq(0,6,2), fxs_etarias )

```

Segue o gráfico gerado pelo R:



Questão E – Faça um gráfico de barras da distribuição proporcional dos óbitos por idade e sexo para Pará e Rio Grande do Sul em 2010. Analise e discuta seus resultados.

Para solucionar este exercício empregaremos a leitura no R diretamente das bases do SIM, arquivos em formato DBF. Os arquivos DBF baixados do site do DATASUS para as UF PA e RS do ano 2010 precisam ser copiados na pasta de dados (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

Para organizar o trabalho sugerimos usar os seguintes nomes para os arquivos DBF:

1. Arquivos de declaracao de obitos Para 2010.dbf

Óbitos para UF PA – Pará, ano 2010.

2. Arquivos de declaracao de obitos Rio Grande do Sul 2010.dbf

Óbitos para UF RS – Rio Grande do Sul, ano 2010

Para carregar de arquivos DBF no R, é necessária a utilização do comando **read.dbf**, o qual foi desenvolvido no biblioteca **foreign**. Para realizar a instalação deste pacote consulte a seção 1.3. Instalando pacotes no R.

Segue programa em R para gerar os gráficos em barras das distribuições proporcionais dos óbitos para as UF PA e RS em 2010 por sexo:

Questão E

```
# define a pasta de trabalho, que precisa ser previamente criada
```

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida
```

```
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta
```

```

# verifique que os arquivos de interesse se encontrem nela
dir()

library (foreign) # open DBF

# nome dos arquivos .DBF
ob.pa.arq = 'Arquivos de declaracao de obitos Para 2010.dbf'
ob.rs.arq = 'Arquivos de declaracao de obitos Rio Grande do Sul 2010.dbf'

# carrego o arquivo de óbitos de 2010 por UF
Ob_PA = read.dbf (ob.pa.arq, as.is=TRUE)
Ob_RS = read.dbf (ob.rs.arq, as.is=TRUE)

# para visualizar a estrutura do banco de dados
# número de linhas, variáveis
str( Ob_PA )

# faixas etárias
fe = c(
'Menor.1.ano' ,
'1.a.4.anos' ,
'5.a.9.anos' ,
'10.a.14.anos' ,
'15.a.19.anos' ,
'20.a.24.anos' ,
'25.a.29.anos' ,
'30.a.34.anos' ,
'35.a.39.anos' ,
'40.a.44.anos' ,
'45.a.49.anos' ,
'50.a.54.anos' ,
'55.a.59.anos' ,
'60.a.64.anos' ,
'65.a.69.anos' ,

```



```

'70.a.74.anos',
'75.a.79.anos',
'80.anos.e.mais')

# SIM (DO) $ IDADE ::
# 0xx - minuto
# 1xx - hora
# 2xx - dia
# 3xx - mês
# 4xx - ano
# 5xx - xx + 100 ano

# limite inferior da faixa etária, >= x
fe.li = c(
'001',
'401',
'405',
'410',
'415',
'420',
'425',
'430',
'435',
'440',
'445',
'450',
'455',
'460',
'465',
'470',
'475',
'480')

# limite superior da faixa etária, <= x

```

```
fe.ls = c(
'400',
'404',
'409',
'414',
'419',
'424',
'429',
'434',
'439',
'444',
'449',
'454',
'459',
'464',
'469',
'474',
'479',
'599')
```

```
# faixas etárias, com limite inferior e superior
data.frame(fe, fe.li, fe.ls)
```

```
# função para extrair o número de óbitos no dbf por uf, sex e faixa etária
n.ob_fxEt_sex = function(dad, fx.et, fx.et.li, fx.et.ls, uf, sex){
# dad - matriz com os dados
# fx.et - vetor com os nomes das faixas etárias
# fx.et.li - limite inferior das faixas etárias, exemplo 401=1 ano
# fx.et.ls - limite superior das faixas etárias, exemplo 404=4 ano
# uf - UF, pelo código IBGE, número com 2 dígitos.
# sex - sexo: 1 (masculino) ou 2 (feminino)

# número de faixas etárias
n.fx.et = length(fx.et)
```

```

# inicializando vetor com 0
n.ob = 0
# número de óbitos em dad (dbf) por UF e sexo
# loop nas faixas etárias
for (ind.fx in 1: n.fx.et){
  n.ob[ind.fx] = length(which(
    dad $ IDADE >= fx.et.li [ind.fx]
    & dad $ IDADE <= fx.et.ls [ind.fx]
    & substr(dad $ CODMUNRES,1,2) == uf
    & dad $ SEXO == sex
  ))
}
n.ob
}

# códigos IBGE UF
# PA - 15
# BA - 29
# RS - 43
# ES - 32

# número de óbitos em dad (dbf) por UF e sexo em 2010
# PA, masculino
n.ob.PA_M = n.ob_fxEt_sex ( Ob_PA, fe, fe.li, fe.ls, 15, 1)
# PA, feminino
n.ob.PA_F = n.ob_fxEt_sex ( Ob_PA, fe, fe.li, fe.ls, 15, 2)

# RS, masculino
n.ob.RS_M = n.ob_fxEt_sex ( Ob_RS, fe, fe.li, fe.ls, 43, 1)
# RS, feminino
n.ob.RS_F = n.ob_fxEt_sex ( Ob_RS, fe, fe.li, fe.ls, 43, 2)

# distribuição proporcional dos óbitos
# por idade e sexo para PA e RS em 2010

```

```

dist.prop.ob_PA.m = n.ob.PA_M / sum(n.ob.PA_M) * 100
dist.prop.ob_PA.f = n.ob.PA_F / sum(n.ob.PA_F) * 100
dist.prop.ob_RS.m = n.ob.RS_M / sum(n.ob.RS_M) * 100
dist.prop.ob_RS.f = n.ob.RS_F / sum(n.ob.RS_F) * 100

# configuração do gráfico
par.mar0 = c(5, 4, 4, 2) + 0.1 # padrão
# com margem inferior maior,
# para ter espaço para os nomes das faixas etárias
par.mar2 = c(8, 4, 4, 2) + 0.1

# função para gerar os gráficos de barras (sexo)
gera.graf_barraSex = function(
dad1, dad2, y.lim, y.lab, tit, leg, leg.pos, ya, fxs_etarias )
{
# configura gráfico
par(col.axis='white', las=3, mar = par.mar2)
plot(1,1, col='white', ylim=c(0,y.lim),
xlim=c(1,length(dad1)), xlab='', ylab=y.lab)

# gera as barras para Sexo Masculino
dad = dad1
n.dad = length(dad)
for (i in 1:n.dad) {
points(c(i,i), c(0,dad[i]), type='l', col='blue', lwd=10, x.axis=NULL)
}

# gera as barras para Sexo Feminino
dad = dad2
n.dad = length(dad)
for (i in 1:n.dad) {
points(c(i,i)+.2, c(0,dad[i]), type='l', col='red', lwd=10, x.axis=NULL)
}
}

```

```

abline(h=ya) # gera o grid de linhas horizontais
title(tit) # nome do gráfico
# gera a caixa dentro do gráfico com a legenda
legend(leg.pos[1], leg.pos[2], leg,
col=c('blue','red', 'dark green'), lwd=10, lty=1 )

# faz a legenda no eixo "Y"
par(col.axis='black', las=3, mar = par.mar2 )
n.axis <- length(ya)
for (i in 1:n.axis) {
axis(2, ya[i], ya[i])
}

# legenda no eixo "X"
x.axis <- fxs_etarias
n.axis <- length(x.axis)
for (i in 1:n.axis) {
axis(1, i, x.axis[i])
}} # fim da função que gera o gráfico

dist.prop = 'Distribuição proporcional dos óbitos,'
legenda_sex2 = c('Masculino', 'Feminino')

### Cada bloco abaixo gera um determinado gráfico ###
### Executar cada um deles separadamente ###

# distribuição proporcional dos óbitos
# por idade e sexo para PA em 2010
gera.graf_barraSex( dist.prop.ob_PA.m , dist.prop.ob_PA.f,
40, dist.prop,
paste(dist.prop, '\n PA - Pará, Ano 2010'),
legenda_sex2, c(5,30), seq(0,40,5) , fe)

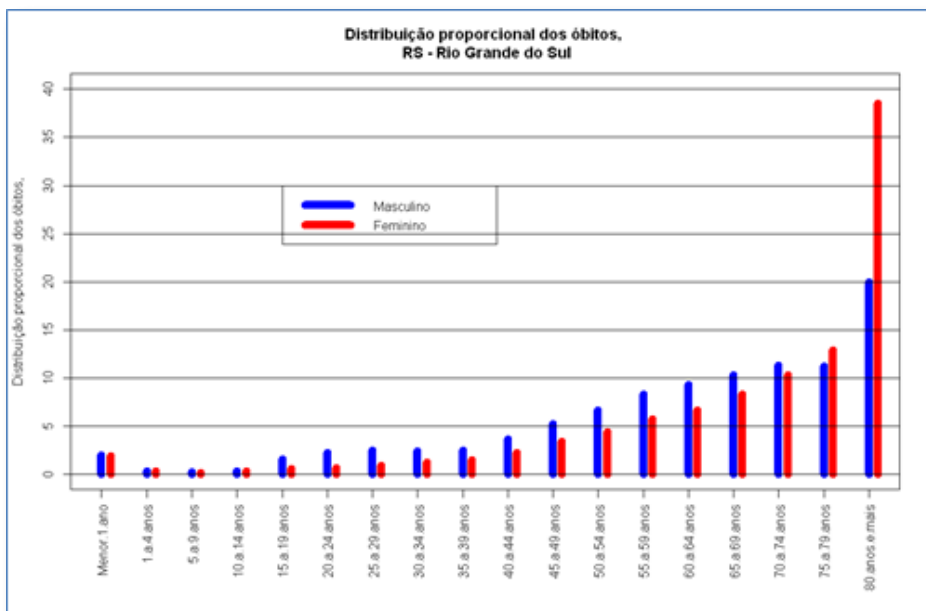
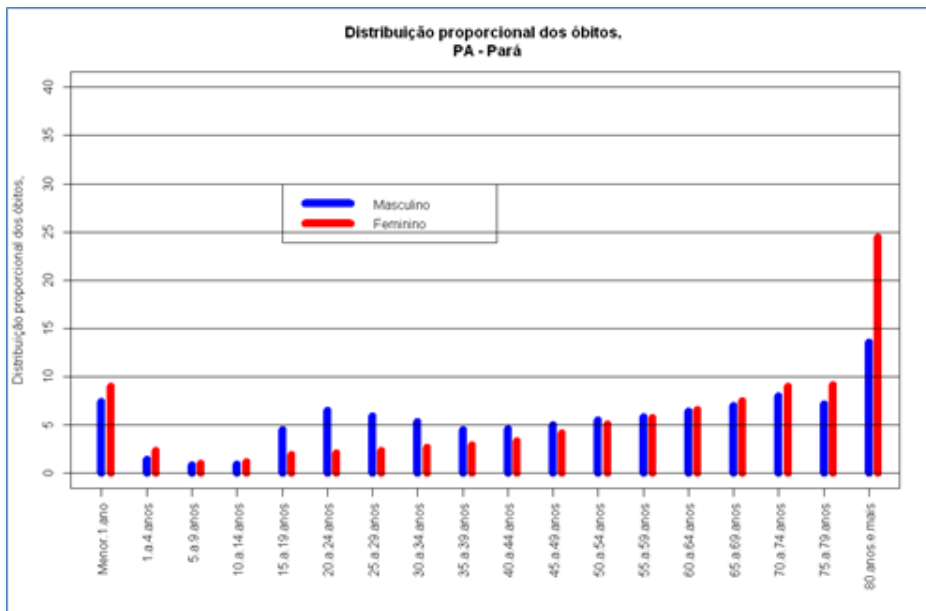
# distribuição proporcional dos óbitos

```

por idade e sexo para RS em 2010

```
gera.graf_barraSex( dist.prop.ob_RS.m , dist.prop.ob_RS.f,  
40, dist.prop,  
paste(dist.prop, '\n RS - Rio Grande do Sul, Ano 2010'),  
legenda_sex2, c(5,30), seq(0,40,5) , fe)
```

Segue os gráficos gerados pelo R:



Questão G – Faça um gráfico para comparação da distribuição proporcional dos três grandes grupos de causas do GBD para Bahia e Rio Grande do Sul em 2010. Avalie.

Para solucionar este exercício empregaremos a leitura no R diretamente das bases do SIM, arquivos em formato DBF. Os arquivos DBF baixados do site do DATASUS para as UF BA e RS do ano 2010 precisam ser copiados na pasta de dados (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

Para organizar o trabalho sugerimos usar os seguintes nomes para os arquivos DBF:

1. Arquivos de declaracao de obitos Bahia 2010.dbf

Óbitos para UF BA – Bahia, ano 2010.

2. Arquivos de declaracao de obitos Rio Grande do Sul 2010.dbf

Óbitos para UF RS – Rio Grande do Sul, ano 2010

Após a extração dos dados de óbitos (arquivos DBF) do site do DATASUS das UF BA e RS, foram gerados os três grupos (gI, gII e gIII) de causas CID-10. Com o número de óbitos por UF e grupo de causa, pode ser calculada distribuição proporcional: $Y_i = (X_i / \sum X_i) * 100$.

Seguem comandos em R para gerar o primeiro gráfico de barras comparativo da distribuição proporcional de óbitos por grupo de causa e UF em 2010:

Questão G

```
# define a pasta de trabalho, que precisa ser previamente criada
```

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida
```

```
setwd (pasta_dados)
```

```

# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()

library (foreign) # open DBF

# nome dos arquivos .DBF
ob.ba.arq = 'Arquivos de declaracao de obitos Bahia 2010.dbf'
ob.rs.arq = 'Arquivos de declaracao de obitos Rio Grande do Sul
2010.dbf'

# carrego o arquivo de óbitos de 2010 por UF
Ob_ba = read.dbf (ob.ba.arq, as.is=TRUE)
Ob_RS = read.dbf (ob.rs.arq, as.is=TRUE)

# para visualizar a estrutura do banco de dados
# número de linhas, variáveis
str( Ob_ba )

# faixas etárias
fe = c(
'Menor.1.ano' ,
'1.a.4.anos' ,
'5.a.9.anos' ,
'10.a.14.anos' ,
'15.a.19.anos' ,
'20.a.24.anos' ,
'25.a.29.anos' ,
'30.a.34.anos' ,
'35.a.39.anos' ,
'40.a.44.anos' ,
'45.a.49.anos' ,
'50.a.54.anos' ,
'55.a.59.anos' ,

```



```

'60.a.64.anos',
'65.a.69.anos',
'70.a.74.anos',
'75.a.79.anos',
'80.anos.e.mais')

# SIM (DO) $ IDADE ::
# 0xx - minuto
# 1xx - hora
# 2xx - dia
# 3xx - mês
# 4xx - ano
# 5xx - xx + 100 ano

# limite inferior da faixa etária, >= x
fe.li = c(
'001',
'401',
'405',
'410',
'415',
'420',
'425',
'430',
'435',
'440',
'445',
'450',
'455',
'460',
'465',
'470',
'475',
'480')

```

```

# limite superior da faixa etária, <= x
fe.ls = c(
'400',
'404',
'409',
'414',
'419',
'424',
'429',
'434',
'439',
'444',
'449',
'454',
'459',
'464',
'469',
'474',
'479',
'599')

# faixas etárias, com limite inferior e superior
data.frame(fe, fe.li, fe.ls)

# função para extrair o número de óbitos no dbf por uf, sex e faixa etária
n.ob_fxEt_sex_grupo_cid = function(dad, fx.et, fx.et.li, fx.et.ls,
uf ){
# dad - matriz com os dados
# fx.et - vetor com os nomes das faixas etárias
# fx.et.li - limite inferior das faixas etárias, exemplo 401=1 ano
# fx.et.ls - limite superior das faixas etárias, exemplo 404=4 ano
# uf - UF, pelo código IBGE, número com 2 dígitos.

```

```

# número de faixas etárias
n.fx.et = length(fx.et)
# inicializando vetor com 0
n.ob_g1 = n.ob_g2 = n.ob_g3 = 0
# número de óbitos em dad (dbf) por UF e sexo
# loop nas faixas etárias
for (ind.fx in 1: n.fx.et){
n.ob_g1[ind.fx] = length(which(
dad $ IDADE >= fx.et.li [ind.fx]
& dad $ IDADE <= fx.et.ls [ind.fx]
& substr(dad $ CODMUNRES,1,2) == uf
&
( # Grupo I (Doenças Infecciosas, causas maternas,perinatais e nu-
tricionais):
(dad $ CAUSABAS >= 'A00' & dad $ CAUSABAS <= 'B99')
| (dad $ CAUSABAS >= 'G00' & dad $ CAUSABAS <= 'G04')
| (dad $ CAUSABAS >= 'N70' & dad $ CAUSABAS <= 'N73')
| (dad $ CAUSABAS >= 'J00' & dad $ CAUSABAS <= 'J06')
| (dad $ CAUSABAS >= 'J09' & dad $ CAUSABAS <= 'J18')
| (dad $ CAUSABAS >= 'J20' & dad $ CAUSABAS <= 'J22')
| (dad $ CAUSABAS >= 'H65' & dad $ CAUSABAS <= 'H66')
| (dad $ CAUSABAS >= 'O00' & dad $ CAUSABAS <= 'O99')
| (dad $ CAUSABAS >= 'P00' & dad $ CAUSABAS <= 'P96')
| (dad $ CAUSABAS >= 'E00' & dad $ CAUSABAS <= 'E02')
| (dad $ CAUSABAS >= 'E40' & dad $ CAUSABAS <= 'E46')
| (dad $ CAUSABAS >= 'E50' & dad $ CAUSABAS <= 'E86')
| (dad $ CAUSABAS >= 'D50' & dad $ CAUSABAS <= 'D53')
| (dad $ CAUSABAS == 'D64' )
| (dad $ CAUSABAS >= 'E51' & dad $ CAUSABAS <= 'E64')
)
))

n.ob_g2[ind.fx] = length(which(

```

```

dad $ IDADE >= fx.et.li [ind.fx]
& dad $ IDADE <= fx.et.ls [ind.fx]
& substr(dad $ CODMUNRES,1,2) == uf
&
( # Grupo II (Doenças crônico-degenerativas):
(dad $ CAUSABAS >= 'C00' & dad $ CAUSABAS <= 'C97')
| (dad $ CAUSABAS >= 'D00' & dad $ CAUSABAS <= 'D48')
| (dad $ CAUSABAS >= 'D55' & dad $ CAUSABAS <= 'D63')
| (dad $ CAUSABAS >= 'D65' & dad $ CAUSABAS <= 'D89')
| (dad $ CAUSABAS >= 'E03' & dad $ CAUSABAS <= 'E07')
| (dad $ CAUSABAS >= 'E10' & dad $ CAUSABAS <= 'E16')
| (dad $ CAUSABAS >= 'E20' & dad $ CAUSABAS <= 'E34')
| (dad $ CAUSABAS >= 'E65' & dad $ CAUSABAS <= 'E85')
| (dad $ CAUSABAS >= 'E87' & dad $ CAUSABAS <= 'E90')
| (dad $ CAUSABAS >= 'F00' & dad $ CAUSABAS <= 'F99')
| (dad $ CAUSABAS >= 'G06' & dad $ CAUSABAS <= 'G98')
| (dad $ CAUSABAS >= 'H00' & dad $ CAUSABAS <= 'H61')
| (dad $ CAUSABAS >= 'H68' & dad $ CAUSABAS <= 'H93')
| (dad $ CAUSABAS >= 'I00' & dad $ CAUSABAS <= 'I99')
| (dad $ CAUSABAS >= 'J30' & dad $ CAUSABAS <= 'J99')
| (dad $ CAUSABAS >= 'K00' & dad $ CAUSABAS <= 'K92')
| (dad $ CAUSABAS >= 'N00' & dad $ CAUSABAS <= 'N64')
| (dad $ CAUSABAS >= 'N75' & dad $ CAUSABAS <= 'N99')
| (dad $ CAUSABAS >= 'L00' & dad $ CAUSABAS <= 'L98')
| (dad $ CAUSABAS >= 'M00' & dad $ CAUSABAS <= 'M99')
)
))

```

```

n.ob_g3[ind.fx] = length(which(
dad $ IDADE >= fx.et.li [ind.fx]
& dad $ IDADE <= fx.et.ls [ind.fx]
& substr(dad $ CODMUNRES,1,2) == uf
&
( # Grupo III (Causas externas):

```

```

dad $ CAUSABAS >= 'V01' & dad $ CAUSABAS <= 'Y98'
)
))
}
res = list()
res[[1]] = n.ob_g1
res[[2]] = n.ob_g2
res[[3]] = n.ob_g3
res
}

# códigos IBGE UF
# PA - 15
# BA - 29
# RS - 43
# ES - 32

# número de óbitos em dad (dbf) por UF em 2010
# BA, masculino + feminino
n.ob.BA = n.ob_fxEt_sex ( Ob_ba, fe, fe.li, fe.ls, 29)

# RS,
n.ob.RS = n.ob_fxEt_sex ( Ob_RS, fe, fe.li, fe.ls, 43)

n.ob.BA_gI = sum(n.ob.BA[[1]])
n.ob.BA_gII = sum(n.ob.BA[[2]])
n.ob.BA_gIII = sum(n.ob.BA[[3]])

n.ob.RS_gI = sum(n.ob.RS[[1]])
n.ob.RS_gII = sum(n.ob.RS[[2]])
n.ob.RS_gIII = sum(n.ob.RS[[3]])

# óbitos por UF e grupo de CID-10
dad.ob.g123_RS = c(n.ob.RS_gI, n.ob.RS_gII, n.ob.RS_gIII ) # UF

```

```

RS: gI, gII e gIII
dad.ob.g123_BA = c(n.ob.BA_gI, n.ob.BA_gII, n.ob.BA_gIII ) # UF
BA: gI, gII e gIII

# distribuição proporcional dos óbitos
# por grupo CID-10 RS e BA em 2010
dist.prop.ob_RS = dad.ob.g123_RS / sum(dad.ob.g123_RS) * 100
dist.prop.ob_BA = dad.ob.g123_BA / sum(dad.ob.g123_BA) * 100

# configuração do gráfico
par.mar0 = c(5, 4, 4, 2) + 0.1 # padrão
# com margem inferior maior,
# para ter espaço para os nomes das faixas etárias
par.mar2 = c(8, 4, 4, 2) + 0.1

# função para gerar os gráficos de barras (sexo)
gera.graf_barra_g123 = function(
dad1, dad2, y.lim, y.lab, tit, leg, leg.pos, ya, legX )
{
# configura gráfico
par(col.axis='white', las=3, mar = par.mar2)
plot(1,1, col='white', ylim=c(0,y.lim),
xlim=c(1,length(dad1)+.2), xlab='', ylab=y.lab)

# gera as barras para Sexo Masculino
dad = dad1
n.dad = length(dad)
for (i in 1:n.dad) {
points(c(i,i), c(0,dad[i]), type='l', col='blue', lwd=12,
x.axis=NULL)
}

# gera as barras para Sexo Feminino
dad = dad2

```

```

n.dad = length(dad)
for (i in 1:n.dad) {
points(c(i,i)+.2, c(0,dad[i]), type='l', col='red', lwd=12,
x.axis=NULL)
}

abline(h=ya) # gera o grid de linhas horizontais
title(tit) # nome do gráfico
# gera a caixa dentro do gráfico com a legenda
legend(leg.pos[1], leg.pos[2], leg,
col=c('blue','red', 'dark green'), lwd=10, lty=1 )

# faz a legenda no eixo "Y"
par(col.axis='black', las=3, mar = par.mar2 )
n.axis <- length(ya)
for (i in 1:n.axis) {
axis(2, ya[i], ya[i])
}

# legenda no eixo "X"
x.axis <- legX
n.axis <- length(x.axis)
for (i in 1:n.axis) {
axis(1, i, x.axis[i])
}} # fim da função que gera o gráfico

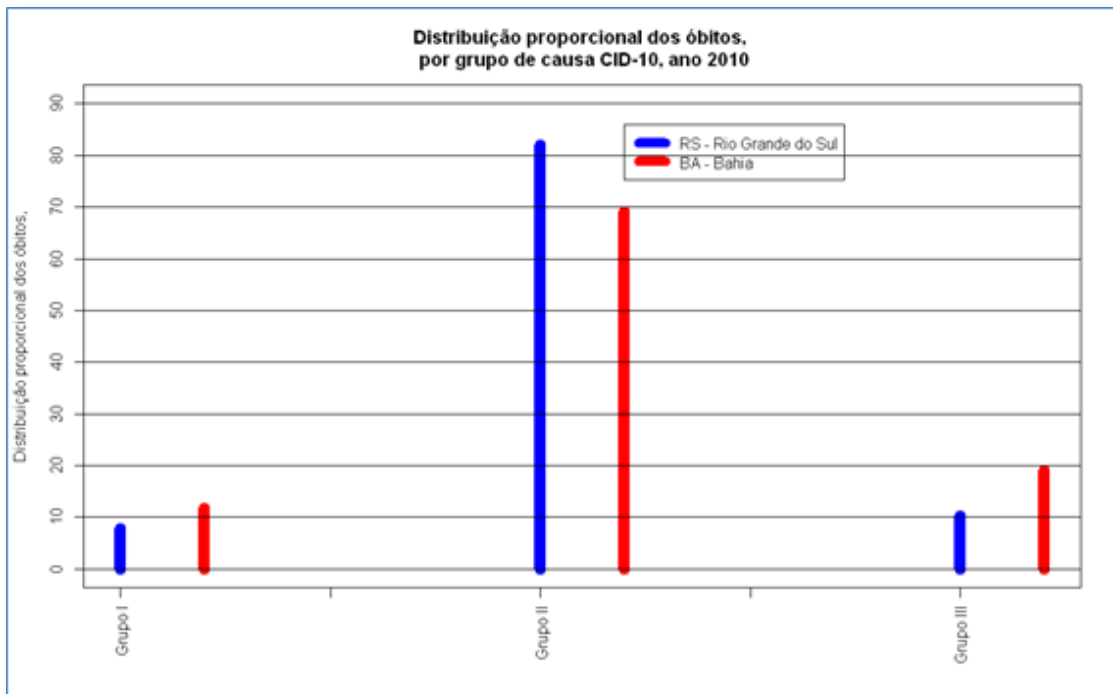
dist.prop = 'Distribuição proporcional dos óbitos,'

# distribuição proporcional dos óbitos
# por idade e sexo para ba em 2010
gera.graf_barra_g123( dist.prop.ob_RS , dist.prop.ob_BA,
90, dist.prop,
paste(dist.prop,'\n por grupo de causa CID-10, ano 2010'),
c('RS - Rio Grande do Sul', 'BA - Bahia' ),

```

```
c(2.2,86), seq(0,90,10),
c('Grupo I', 'Grupo II', 'Grupo III') )
```

Segue gráfico gerado no R:



Para gerar os gráficos de linha é necessário carregar a função **gera.graf3l** da Questão D, para isto execute do código em R desse exercício, desde a linha:

```
# função para gerar os gráficos, agora com 3 linhas (UF)
Até a linha
}} # fim da função que gera o gráfico
```

Após carregar a função necessária, execute os seguintes comandos no R:

```
# 2da parte, ### exercício 5.7
# carregar função "gera.graf3l" do exercício 5.4
# número de óbitos totais por faixa etária, juntando os três grupos de cid
```



```

n.ob.BA_tot_fe = n.ob.BA[[1]] + n.ob.BA[[2]] + n.ob.BA[[3]]
n.ob.RS_tot_fe = n.ob.RS[[1]] + n.ob.RS[[2]] + n.ob.RS[[3]]

# óbitos para UF BA por grupo de CID-10
dad.ob.g1_BA = n.ob.BA[[1]] / n.ob.BA_tot_fe * 100
dad.ob.g2_BA = n.ob.BA[[2]] / n.ob.BA_tot_fe * 100
dad.ob.g3_BA = n.ob.BA[[3]] / n.ob.BA_tot_fe * 100

# óbitos para UF RS e grupo de CID-10
dad.ob.g1_RS = n.ob.RS[[1]] / n.ob.RS_tot_fe * 100
dad.ob.g2_RS = n.ob.RS[[2]] / n.ob.RS_tot_fe * 100
dad.ob.g3_RS = n.ob.RS[[3]] / n.ob.RS_tot_fe * 100

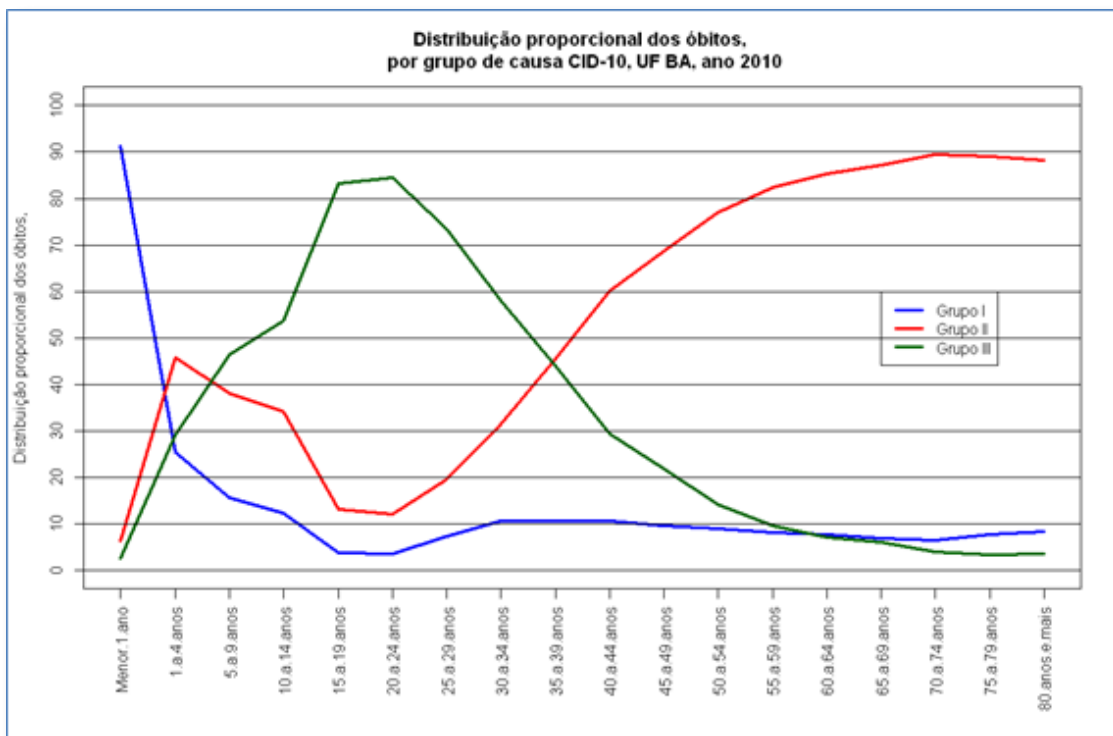
### Cada bloco abaixo gera um determinado gráfico ###
### Executar cada um deles separadamente ###

# Distribuição proporcional por grupo CID, UF BA, ano 2010
gera.graf3l(dad.ob.g1_BA,
dad.ob.g2_BA,
dad.ob.g3_BA,
100, dist.prop,
paste(dist.prop, '\n por grupo de causa CID-10, UF BA, ano 2010'),
c('Grupo I', 'Grupo II', 'Grupo III'),
c(15,60), seq(0,100,10) , fe)

# Distribuição proporcional por grupo CID, UF RS, ano 2010
gera.graf3l(dad.ob.g1_RS,
dad.ob.g2_RS,
dad.ob.g3_RS,
100, dist.prop,
paste(dist.prop, '\n por grupo de causa CID-10, UF RS, ano 2010'),
c('Grupo I', 'Grupo II', 'Grupo III'),
c(15,60), seq(0,100,10) , fe)

```

Segue um dos dois gráficos gerados no R:



Questão H – Compare a razão entre doenças crônicas (Grupo II) e doenças transmissíveis, maternas, perinatais e nutricionais (Grupo I) para Bahia e Rio Grande do Sul em 2010. Interprete.

Este exercício pode ser resolvido no R de forma análoga a Questão G.

Resoluções do Módulo 5

Análise de Inquéritos Populacionais

Nesta seção são apresentadas as soluções dos exercícios do Módulo “Análise de Inquéritos populacionais” na ferramenta R. Apresentamos também, um exercício com os dados do VIGITEL (Vigilância de Fatores de Risco e Proteção para Doenças Crônicas por Inquérito Telefônico) Brasil 2010.

Atividade 1

Questão A - A partir da planilha Excel EAD_Inq_17_09.xls gere o arquivo EAD_Inq_17_09.csv. Para fazer a conversão consulte a seção 1.9. Conversão de arquivos XLSX para CSV. Sugerimos o seguinte nome para o arquivo texto:

Modulo_InqueritoPop_QuestaoA.csv

Segue programa em R para a carga dos dados:

```
### Questão A, Inquérito Populacional
```

```
# define a pasta de trabalho, que precisa ser previamente criada  
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida  
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta  
# verifique que os arquivos de interesse se encontrem nela  
dir()
```

```
# carrega .CSV
```

```
dad = read.table (file = 'Modulo_InqueritoPop_QuestaoA.csv',
  sep=';', header=TRUE,
  stringsAsFactors=FALSE, dec='.')
```

Para gerar a matriz Hepatite B vs Sexo, execute no R os comandos abaixo:

```
# HepatiteB vs Sexo
var.x = dad $ sexo # 0=M, 1=F
var.y = dad $ hbc_1 # HepatiteB: 0 - negativo , 1 - positivo
cat.x = sort(unique(var.x))
cat.y = sort(unique(var.y))
n.x = length(cat.x)
n.y = length(cat.y)
mat = matrix(0, n.x, n.y)
for(j in 1:n.x) {
  for (k in 1:n.y) {
    mat[j,k] = length(which(var.x == cat.x[j] & var.y == cat.y[k]))
  }
}
cat.x # linha
cat.y # coluna
# nomes para linhas e colunas
lin.col = list()
lin.col[[1]] = c('Masculino', 'Feminino') # linha = sexo
lin.col[[2]] = c('Negativo', 'Positivo') # coluna = Hep B
dimnames(mat) = lin.col
# Matriz: linha = Sexo, coluna = Hepatite B
mat
mat_sex = mat
```

Segue saída do R:

```
> mat
      Negativo Positivo
Masculino    608      85
Feminino    704      60
>
```

Podemos escrever um *script* análogo ao anterior para gerar a matriz Hepatite B vs Faixa etária:

```
# HepatiteB vs Faixa etária
var.x = dad $ agegroup # 2 -10 a 19, 3 - 20 a 39, 4 - 40 a 59, 5
- 60 ou mais
var.y = dad $ hbc_1 # HepatiteB: 0 - negativo , 1 - positivo
cat.x = sort(unique(var.x))
cat.y = sort(unique(var.y))
n.x = length(cat.x)
n.y = length(cat.y)
mat = matrix(0, n.x, n.y)
for(j in 1:n.x) {
for (k in 1:n.y) {
mat[j,k] = length(which(var.x == cat.x[j] & var.y == cat.y[k]))
}
}
cat.x # linha
cat.y # coluna
# nomes para linhas e colunas
lin.col = list()
lin.col[[1]] = c('10 a 19', '20 a 39', '40 a 59', '60 ou mais') # li-
nha = faixa etária
lin.col[[2]] = c('Negativo', 'Positivo') # coluna = Hep B
dimnames(mat) = lin.col
# Matriz: linha = faixa etária, coluna = Hepatite B
mat
mat_fx_etaria = mat
```

Segue saída do R:

```
> mat
      Negativo Positivo
10 a 19      496      17
20 a 39      444      46
40 a 59      317      61
60 ou mais      55      21
>
```

Questão B - Após a execução dos comandos do exercício anterior (questão 1), deixe a tela do R aberta. Na sequência execute os comandos abaixo para realizar o ajuste do modelo de regressão logística:

Questão B, Inquérito Populacional

```
# após executar os comandos da Questão A.:

# Regressão Logística, Hepatite B vs Sexo
# glm - modelos lineares generalizados
# family=binomial - modelo de regressão logística
modelo <- glm(dad $ hbc_1 ~ dad $ sexo3, family=binomial)

# Para visualizar algumas informações estatísticas
# do ajuste do modelo, como os coeficientes:
summary(modelo)
```

Segue a saída do R:

```
> summary(modelo)

Call:
glm(formula = dad$hbc_1 ~ dad$sexo3, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5116 -0.5116 -0.4044 -0.4044  2.2558

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.9573     0.2928 -10.099 < 2e-16 ***
dad$sexo3     0.4949     0.1775  2.789  0.00529 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 944.21  on 1456  degrees of freedom
Residual deviance: 936.31  on 1455  degrees of freedom
(102 observations deleted due to missingness)
AIC: 940.31

Number of Fisher Scoring iterations: 5

>
```

Neste caso observa-se associação significativa entre a exposição ao vírus da hepatite B e o sexo, p-valor = 0.00529.

Para fazer as comparações entre homens e mulheres pela % de positivo por sexo e não pelo número de hepatite B positivo, , pois temos mais mulheres (764) do que homens (693), execute os comandos abaixo no R:

```
# totais de Hep B positivo por sexo
tot_sex = apply(mat_sex,1,sum)
tot_sex
hepB_sex_porc = round(mat_sex / tot_sex *100, 1) # %
# verifica que as linhas somam 100%
apply(hepB_sex_porc, 1, sum)
# Hepatite B vs Sexo , %
hepB_sex_porc
```

Na saída do R:

```
> hepB_sex_porc
           Negativo Positivo
Masculino    87.7    12.3
Feminino    92.1     7.9
>
```

Note que a porcentagem de positividade de pessoas com Hepatite B em homens é maior (12,3%), enquanto que em mulheres é menor (7,9%). A partir da matriz de Hepatite B vs Sexo (número de casos positivos) pode ser construída uma tabela de contingência. Com essa tabela o R calcula a significância da associação. Segue comandos em R:

```
# linhas: Hepatite B
# colunas: Sexo

# Pearson's Chi-squared Test for Count Data
chisq.test (mat_sex)

# Fisher's Exact Test for Count Data
fisher.test (mat_sex)
```

Segue a saída do R:

```
>
> # linhas: Hepatite B
> # colunas: Sexo
>
> # Pearson's Chi-squared Test for Count Data
> chisq.test (mat_sex)

        Pearson's Chi-squared test with Yates' continuity correction

data:  mat_sex
X-squared = 7.409, df = 1, p-value = 0.00649

>
>
>
> # Fisher's Exact Test for Count Data
> fisher.test (mat_sex)

        Fisher's Exact Test for Count Data

data:  mat_sex
p-value = 0.005077
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.4229076 0.8750487
sample estimates:
odds ratio
 0.6098418

>
```

Note que pelas duas metodologias há associação significativa entre a exposição ao vírus da hepatite B e o sexo, com p-valores menores do que 0.007.

Para obter os resultados com a variável **agegroup** e **escola3**, segue comandos em R:

```
modelo <- glm(dad $ hbc_1 ~ dad $ escola3, family=binomial)
summary(modelo)
```

```
modelo <- glm(dad $ hbc_1 ~ dad $ agegroup, family=binomial)
summary(modelo)
```



```

# totais de Hep B positivo por faixa etária
tot_fxIdade = apply(mat_fx_etaria,1,sum)
hepB_fxIdade_porcentagem = round(mat_fx_etaria / tot_fxIdade *100, 1) # %
# verifica que as linhas somam 100
apply(hepB_fxIdade_porcentagem, 1, sum)
# Hepatite B vs etária, %
hepB_fxIdade_porcentagem

chisq.test (mat_fx_etaria)
fisher.test (mat_fx_etaria)

```

Segue saída do R para a regressão logística da Hepatite B vs a variável **escola3**:

```

> summary(modelo)

Call:
glm(formula = dad$hbc_1 ~ dad$escola3, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5651 -0.4627 -0.4627 -0.3772  2.4824

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.3270     0.3011  -4.407 1.05e-05 ***
dad$escola3  -0.4268     0.1375  -3.103  0.00191 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 878.28  on 1395  degrees of freedom
Residual deviance: 868.32  on 1394  degrees of freedom
(163 observations deleted due to missingness)
AIC: 872.32

Number of Fisher Scoring iterations: 5
>

```

Segue saída do R para a regressão logística da Hepatite B vs a variável **agegroup**:

```
> modelo <- glm(dad $ hbc_1 ~ dad $ agegroup, family=binomial)
> summary(modelo)

Call:
glm(formula = dad$hbc_1 ~ dad$agegroup, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8376 -0.5962 -0.4152 -0.2857  2.5374

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.71928    0.35893  -13.148 < 2e-16 ***
dad$agegroup  0.77045    0.09898   7.784 7.01e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 944.21  on 1456  degrees of freedom
Residual deviance: 879.77  on 1455  degrees of freedom
(102 observations deleted due to missingness)
AIC: 883.77

Number of Fisher Scoring iterations: 5

>
```

Segue resultados gerados no R para a faixa etária (**agegroup**):

```
> hepB_fxIdade_porc
      Negativo Positivo
10 a 19      96.7      3.3
20 a 39      90.6      9.4
40 a 59      83.9     16.1
60 ou mais    72.4     27.6
>
> chisq.test (mat_fx_etaria)

      Pearson's Chi-squared test

data:  mat_fx_etaria
X-squared = 68.0456, df = 3, p-value = 1.119e-14

> fisher.test (mat_fx_etaria)

      Fisher's Exact Test for Count Data

data:  mat_fx_etaria
p-value = 1.85e-14
alternative hypothesis: two.sided

>
```

Note que há um aumento visível da porcentagem de positividade à Hepatite B com a idade (faixa etária): 3, 9, 16 e 27% respectivamente. A associação entre as variáveis Hepatite B e idade é significativa, a qual é confirmada na regressão logística e nas tabelas de contingência, p-valor <0.0001. Seguem comandos em R para gerar um gráfico da porcentagem crescente de positividade da Hepatite B vs Faixa etária:

```
graf.nome = '% de Hepatite B Positivo'
par(col.axis='white' )
plot(1,1,col='white', ylab=graf.nome, main=graf.nome,
```

```

xlab='Faixa etária', xlim=c(1,4), ylim=c(3,28))
points(hepB_fxIdade_porcentagem[,2],
type='l', pch=15, col='red', lwd=3
)

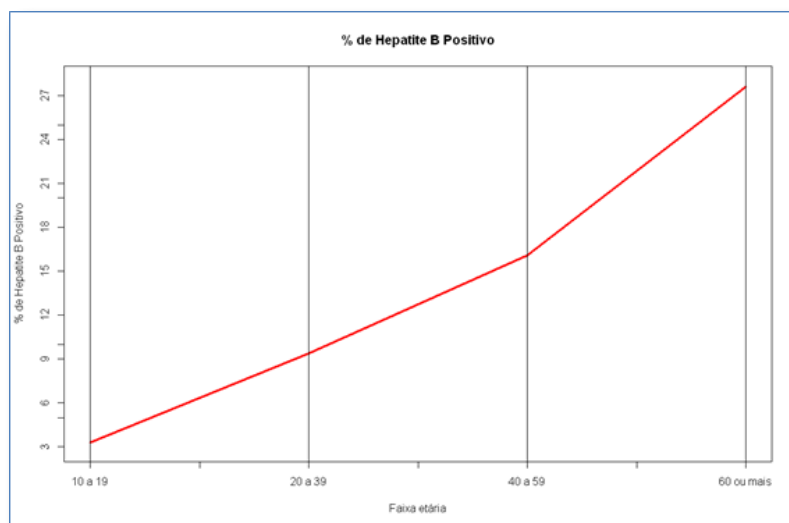
# legenda no eixo "X"
par(col.axis='black')
x.axis <- lin.col[[1]]
n.axis <- length(x.axis)
for (i in 1:n.axis) {
axis(1, i, x.axis[i])
}

# legenda no eixo "Y"
y.axis <- seq(3,27,3)
n.axis <- length(y.axis )
for (i in 1:n.axis) {
axis(2, y.axis[i], y.axis[i])
}

abline(v=1:4)

```

Segue o gráfico gerado no R:



Exemplo VIGITEL

Nesta seção apresentamos a solução de um exercício proposto com as bases de dados do VIGITEL (Vigilância de Fatores de Risco e Proteção para Doenças Crônicas por Inquérito Telefônico) Brasil – 2010.

Questão A - Calcular as estimativas para a prevalência de fumantes com intervalo de confiança de 95% para o total e por sexo para a capital de Goiânia – GO em 2010.

Para acessar e baixar os dados do VIGITEL:

1. No site do **DATASUS**, <www.datasus.gov.br/>
2. Entre na área **Informações de Saúde (TABNET)**
3. Acesse **Inquéritos e Pesquisas**
4. Escolha **Vigitel 2010**
5. Não é necessário fazer qualquer tabulação. Dirija-se até o final desta página e acesse: **Bases de Dados disponíveis da SVS**
6. Nesta tela clique no botão **VIGITEL – Inquérito**
7. É necessário preencher as informações solicitadas:
 - a. Nome completo do solicitante
 - b. CPF do solicitante
 - c. Endereço do solicitante
 - d. CEP
 - e. Motivo da solicitação
8. Selecione Capitais = **Goiânia** e ano solicitado = **2010**
9. Clique no botão **Gerar CSV**
10. Clique no *link* **VIGITEL_MicroDado_2010_c10.CSV**, para baixar o micro dado, arquivo texto em formato CSV
11. Salve o arquivo gerado com nome: **VIGITEL_MicroDado_2010_c10.CSV** na **pasta de dados** (seção 1.5. Criando uma pasta de trabalho para o R)

Os dados baixados referem-se ao inquérito realizado no município de Goiânia – GO em 2010. Para realizarmos esta análise estatística com amostra complexa, precisamos trabalhar

com os pesos finais, já calculados nos dados disponíveis, na coluna *pesofim*. Esta medida é utilizada no cálculo das prevalências e seu intervalo de confiança.

Na solução deste exercício, para determinação dos IC95%, usaremos um pacote pronto (**survey**) no R para o trabalho com amostra complexa. Para realizar a instalação deste pacote consulte a seção 1.3. Instalando pacotes no R.

Para carregar os dados (arquivo CSV) e a biblioteca **survey** na memória do R, execute os comandos abaixo:

Exemplo VIGITEL, Inquérito Populacional

```
# prevalência de fumantes em Goiânia - GO por sexo, com IC
```

```
# define a pasta de trabalho, que precisa ser previamente criada  
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida  
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta  
# verifique que os arquivos de interesse se encontrem nela  
dir()
```

```
# carrega .CSV
```

```
dad = read.table (file = 'VIGITEL_MicroDado_2010_c10.CSV',  
  sep=';', header=TRUE,  
  stringsAsFactors=FALSE, dec='.')
```

```
library(survey)
```

```
niv.conf = 0.95 # IC 95%
```

Primeiramente vamos calcular a estimativa para a prevalência de fumantes com IC de 95% para o total (Masculino + Feminino). Segue programa em R:

```

# prevalência de fumante
# total
fumante = dad $ fumante
peso = dad $ pesofim
numerador = sum(peso [ which(fumante==1) ])
denominador = sum(peso)
fum.prev = round(numerador / denominador * 100, 1)
# Prevalência Fumante Total
fum.prev

dad2 = dad[, c('q7', 'fumante', 'pesofim')] # sexo, variável, peso
final
dclus1 <- svydesign(id=~1, weights=~pesofim, data=dad2)
res.tot = svyciprop(~I(fumante==1), dclus1, method="me", level=-
niv.conf)
# Prevalência Fumante Total
round( res.tot[[1]] * 100, 1 )
# # # IC 95%
round( attr(res.tot, "ci" ) *100, 1)

```

Segue a saída do R com o cálculo da estimativa da prevalência de fumantes total (14.6%) e o IC95% [2.5% - 97.5%] (11.7% , 17.4):

```

> # Prevalência Fumante Total
> round( res.tot[[1]] * 100, 1 )
[1] 14.6
> # # # IC 95%
> round( attr(res.tot, "ci" ) *100, 1)
2.5% 97.5%
11.7 17.4
>

```

Para calcular a estimativa da prevalência de fumantes com intervalo de confiança de 95% para o sexo Masculino:

```
# prevalência de fumante
# sexo Masculino
ind_Sexo = which(dad $ q7 ==1) # Sexo: 1-Masculino, 2-Feminino
fumante = dad $ fumante [ind_Sexo]
peso = dad $ pesofim [ind_Sexo]
numerador = sum(peso [ which(fumante==1) ])
denominador = sum(peso)
fum.prevM = round(numerador / denominador * 100, 1)
# Prevalência Fumante sexo Masculino
fum.prevM

dad2 = dad[ ind_Sexo, c('q7', 'fumante', 'pesofim')] # sexo, variável, peso final
dclus1 <- svydesign(id=~1, weights=~pesofim, data=dad2)
res.tot = svyciprop(~I(fumante==1), dclus1, method="me", level=-niv.conf)
# Prevalência Fumante sexo Masculino
round( res.tot[[1]] * 100, 1 )
# # # IC 95%
round( attr(res.tot, "ci" ) *100, 1)
```

Segue a saída do R com o cálculo da estimativa da prevalência de fumantes do sexo Masculino (19.7%) e o IC95% [2.5% - 97.5%] (14.8% , 24.6):

```
> # Prevalência Fumante sexo Masculino
> round( res.tot[[1]] * 100, 1 )
[1] 19.7
> # # # IC 95%
> round( attr(res.tot, "ci" ) *100, 1)
2.5% 97.5%
14.8 24.6
>
```


Para calcular a estimativa da prevalência de fumantes com intervalo de confiança de 95% para o sexo Feminino:

```
# prevalência de fumante
# sexo Feminino
ind_Sexo = which(dad $ q7 ==2) # Sexo: 1-Masculino, 2-Feminino
fumante = dad $ fumante [ind_Sexo]
peso = dad $ pesofim [ind_Sexo]
numerador = sum(peso [ which(fumante==1) ])
denominador = sum(peso)
fum.prevF = round(numerador / denominador * 100, 1)
# Prevalência Fumante sexo Feminino
fum.prevF

dad2 = dad[ ind_Sexo, c('q7', 'fumante', 'pesofim')] # sexo, variável, peso final
dclus1 <- svydesign(id=~1, weights=~pesofim, data=dad2)
res.tot = svyciprop(~I(fumante==1), dclus1, method="me", level=niv.conf)
# Prevalência Fumante sexo Feminino
round( res.tot[[1]] * 100, 1 )
# # # IC 95%
round( attr(res.tot, "ci" ) *100, 1)
```

Segue a saída do R com o cálculo da estimativa da prevalência de fumantes do sexo Feminino (10.1%) e o IC95% [2.5% - 97.5%] (7.0% , 13.2):

```
> # Prevalência Fumante sexo Feminino
> round( res.tot[[1]] * 100, 1 )
[1] 10.1
> # # # IC 95%
> round( attr(res.tot, "ci" ) *100, 1)
2.5% 97.5%
 7.0 13.2
>
```

Resoluções do Módulo 6

Análise de Séries Temporais na Epidemiologia

Nesta seção apresentamos a solução das atividades do Módulo de “Análise de Séries Temporais na Epidemiologia”.

Atividade 1

Tendência da mortalidade infantil em dois Estados brasileiros: São Paulo (SP) e Maranhão (MA).

Segue uma tabela com os valores apresentados no caderno de exercícios:

Ano	Obitos Menor 1 ano SP	NV SP	TMI x 1000 SP	Obitos Menor 1 ano MA	NV MA	TMI x 1000 MA
1996	15710	699013	22,47	1069	61056	17,51
1997	15159	701947	21,60	1213	75392	16,09
1998	13756	693413	19,84	1501	79272	18,93
1999	12796	714428	17,91	1543	96587	15,98
2000	11922	687779	17,33	1898	100811	18,83
2001	10437	632483	16,50	2188	108527	20,16
2002	9534	623302	15,30	2425	117917	20,57
2003	9273	610555	15,19	2473	127920	19,33
2004	8959	618080	14,49	2213	126518	17,49
2005	8353	618880	13,50	2465	130266	18,92
2006	8078	603368	13,39	2240	127724	17,54
2007	7774	595408	13,06	2164	127307	17,00
2008	7585	601795	12,60	2110	128302	16,45
2009	7482	598473	12,50	2051	123635	16,59
2010	7163	601352	11,91	1860	119566	15,56

Para trabalhar com estes valores, é necessário converter o arquivo para CSV. Consulte a seção 1.9. Conversão de arquivos XLSX para CSV. Sugerimos o seguinte nome para o arquivo de dados: `Modulo_SerieTemporal_Atividade1.csv`

Segue programa em R:

Atividade 1, Módulo Séries Temporais

```
# define a pasta de trabalho, que precisa ser previamente criada
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()
```

```
# Carrega o arquivo texto CSV com os dados
dad = read.table (file = 'Modulo_SerieTemporal_Atividade1.csv',
    sep=';',
    header=TRUE,
    stringsAsFactors=FALSE,
    dec=',')
```

```
# legenda dos eixos perpendiculares aos eixos
par (las=2)
```

```
# gera a 1ª linha, SP
```

```

plot(1996:2010, dad $ TMI.x.1000.SP,
     type='l', lwd=3, col='blue',
     ylim=c(0,25), xlab='Ano',
     ylab='Mortalidade Infantil (por 1000 nascidos vivos)')

# gera a segunda 2ª linha, MA
points(1996:2010, dad $ TMI.x.1000.MA,
       type='l', lwd=3, col='red')

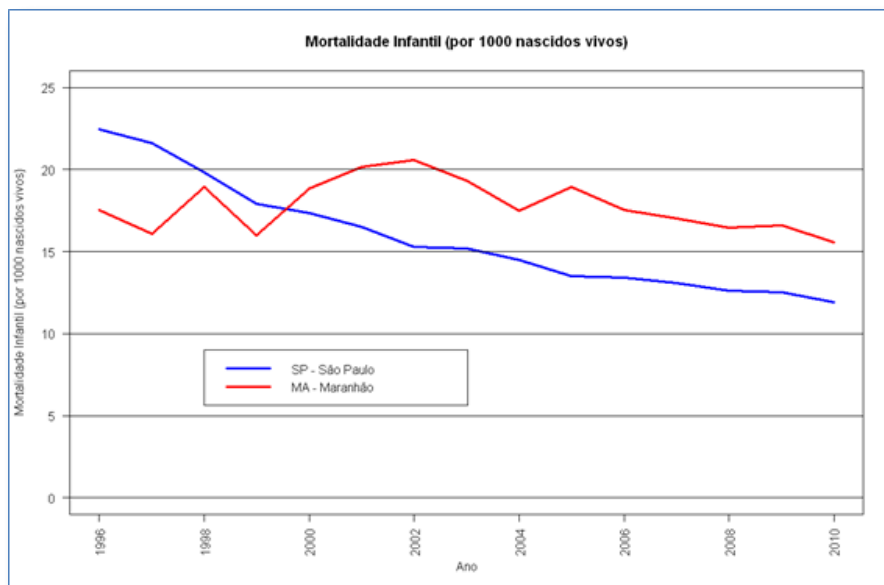
# constroi a grade de linhas horizontais
abline(h=seq(0,25,5))

# especifica a legenda do gráfico
legend(1998,9,c('SP - São Paulo','MA - Maranhão'),
      lwd=3, col=c('blue','red'), lty=1)

# nome do gráfico
title('Mortalidade Infantil (por 1000 nascidos vivos)')

```

Segue o gráfico gerado no R com o *script* anterior:



Segue *script* em R para gerar o gráfico das taxas de mortalidade Infantil (séries históricas) das UF SP e MA na escala Log:

```
# logaritmo dos coeficiente de mortalidade infantil
mort.inf_SP.log = log(dad $ TMI.x.1000.SP)
mort.inf_MA.log = log(dad $ TMI.x.1000.MA)

#gráfico na escala LOG

# legenda dos eixos perpendiculares aos eixos
par(las=2)

# gera a 1ª linha, SP
plot(1996:2010, mort.inf_SP.log,
     type='l', lwd=3, col='blue',
     ylim=c(2.4,3.2), xlab='Ano',
     ylab='Log da Mortalidade Infantil (por 1000 nascidos vivos)')

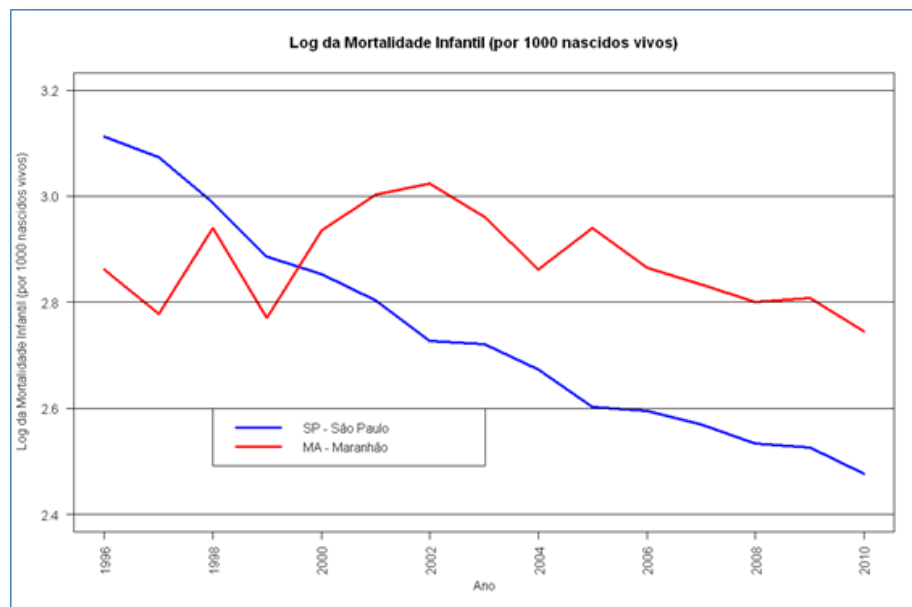
# gera a segunda 2ª linha, MA
points(1996:2010, mort.inf_MA.log,
       type='l', lwd=3, col='red')

# constroi a grade de linhas horizontais
abline(h=seq(2.4,3.2,.2))

# especifica a legenda do gráfico
legend(1998,2.6,c('SP - São Paulo','MA - Maranhão'),
      lwd=3, col=c('blue','red'), lty=1)

# nome do gráfico
title('Log da Mortalidade Infantil (por 1000 nascidos vivos)')
```

Segue o gráfico gerado no R com o *script* anterior:



Especificamente a técnica para análise de série temporal Prais-Winsten não está disponível no R. Pode ser utilizada uma regressão linear para avaliar a existência de tendência significativa e a inclinação da reta, para os coeficientes de mortalidade infantil na escala log. Seguem os comandos em R para gerar o resultado do ajuste linear para os dados da UF SP:

```

serie = mort.inf_SP.log
alpha = 0.05 # 5%
n.serie = length(serie) # n° de anos analisados
x.ind = 1 : n.serie # ano 1996 a 2010

lm.res = lm ( serie ~ x.ind ) # Ajuste Linear
lm.coef = lm.res $ coefficients
names(lm.coef) = NULL
tendencia = FALSE
tendencia = summary(lm.res) $ coefficients [2,4] <= alpha
if (is.na(tendencia) | is.null(tendencia)) { tendencia = FALSE }
if (!tendencia) {pos.neg = ''} else {
pos.neg = '(+)'
if (summary(lm.res) $ coefficients [2,1] < 0 ) { pos.neg = '(-)' }
}

```

```

pv.lm = round(summary(lm.res) $ coefficients [2,4], digits=4)
pv.lm2 = ifelse(pos.neg=='', '', pv.lm)
summary(lm.res)

```

Segue a saída em R para a análise de tendência da mortalidade infantil (por 1.000 nascidos vivos) na escala log:

```

> summary(lm.res)

Call:
lm(formula = serie ~ x.ind)

Residuals:
    Min       1Q   Median       3Q      Max
-0.05936 -0.02651 -0.01431  0.03452  0.06247

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.099421    0.022720  136.42 < 2e-16 ***
x.ind        -0.044601    0.002499  -17.85 1.59e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04181 on 13 degrees of freedom
Multiple R-squared:  0.9608,    Adjusted R-squared:  0.9578
F-statistic: 318.6 on 1 and 13 DF,  p-value: 1.589e-10

>

```

Com este resultado confirmamos que sim, há tendência significativa de queda, com inclinação de -0.0446, e p-valor <0.0001.

Para realizar a estimação da tendência pelo modelo de regressão, execute os comandos abaixo no R:

```

# procedimento de estimação de tendências
anos = 1996:2010
# anos = 1:length(anos)
glm.res = glm(dad $ TMI.x.1000.SP ~ anos, family = "poisson")
summary( glm.res )

```

Segue a saída do R:

```

> summary( glm.res )

Call:
glm(formula = dad$TMI.x.1000.SP ~ anos, family = "poisson")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.24099 -0.12416 -0.03668  0.13548  0.24624

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  95.04133   30.42451   3.124  0.00179 **
anos         -0.04608    0.01520  -3.032  0.00243 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 9.67664  on 14  degrees of freedom
Residual deviance: 0.37099  on 13  degrees of freedom
AIC: Inf

Number of Fisher Scoring iterations: 3

>

```

Note que a estimativa para o coeficiente da variável ano, feita pelo R (-0.046) fica muito próxima do cálculo (-0.045) do Stata (Prais-Winsten). E o intercepto (constante) também resulta no valor aproximado 95 (R) vs 93 (Stata). Note que o p-valor (0.00243) obtido para o coeficiente “anos” é significativo (<0.01). Isto confirma a hipótese de diminuição significativa da taxa de mortalidade infantil em São Paulo no período de 1996 a 2010.

Para fazer o mesmo procedimento para a UF MA, segue *script* em R:

```

anos = 1996:2010
glm.res = glm(dad $ TMI.x.1000.MA ~ anos, family = "poisson")
summary( glm.res )

```


Segue a saída do R:

```
> summary( glm.res )

Call:
glm(formula = dad$TMI.x.1000.MA ~ anos, family = "poisson")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.55276 -0.21993 -0.04768  0.24212  0.61799

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 14.29581    28.37847   0.504   0.614
anos        -0.00570     0.01417  -0.402   0.687

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 1.9328  on 14  degrees of freedom
Residual deviance: 1.7709  on 13  degrees of freedom
AIC: Inf

Number of Fisher Scoring iterations: 3

>
```

Note que para a UF MA não obtemos p-valor significativo. Este resultado nos confirma a hipótese de tendência estacionária para a taxa de mortalidade infantil no estado de Maranhão.

Atividade 2

Nesta seção apresentamos dois exemplos de série temporal de análise de sazonalidade significativa.

Exemplo com dados de Leptospirose - Certifique-se que os dados secundários necessários tenham sido baixados do site do DATASUS <www.datasus.gov.br> e armazenados na pasta de dados (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

O passo a passo para obter os dados foi descrito na Atividade 2 do caderno com os exercícios propostos no Módulo “Análise de Séries Temporais na Epidemiologia”. Para organizar o trabalho sugerimos usar o seguinte nome para o arquivo **CSV: Modulo_SerieTemporal_Atividade2.csv**

Segue programa em R para carregar os dados e gerar o gráfico de número mensal (2001 a 2006) de casos com primeiros sintomas de Leptospirose:

Atividade 2, Módulo Séries Temporais

```
# define a pasta de trabalho, que precisa ser previamente criada
pasta_dados = 'C:/Curso_EAD_ASA/'

# o R se posiciona na pasta definida
setwd (pasta_dados)

# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()

# Carrega o arquivo texto CSV com os dados
lepto_sp = read.table (file = 'Modulo_SerieTemporal_Atividade2.csv',
sep=';', header=TRUE, skip=4,
stringsAsFactors=FALSE, dec='.')

lepto_sp
# linha=mês(Jan a Dez), coluna=ano(2001 a 2006)
dad = lepto_sp[1:12, 2:7]
# mínimo e máximo, juntando os 6 anos
limy = c(min(dad), max(dad))

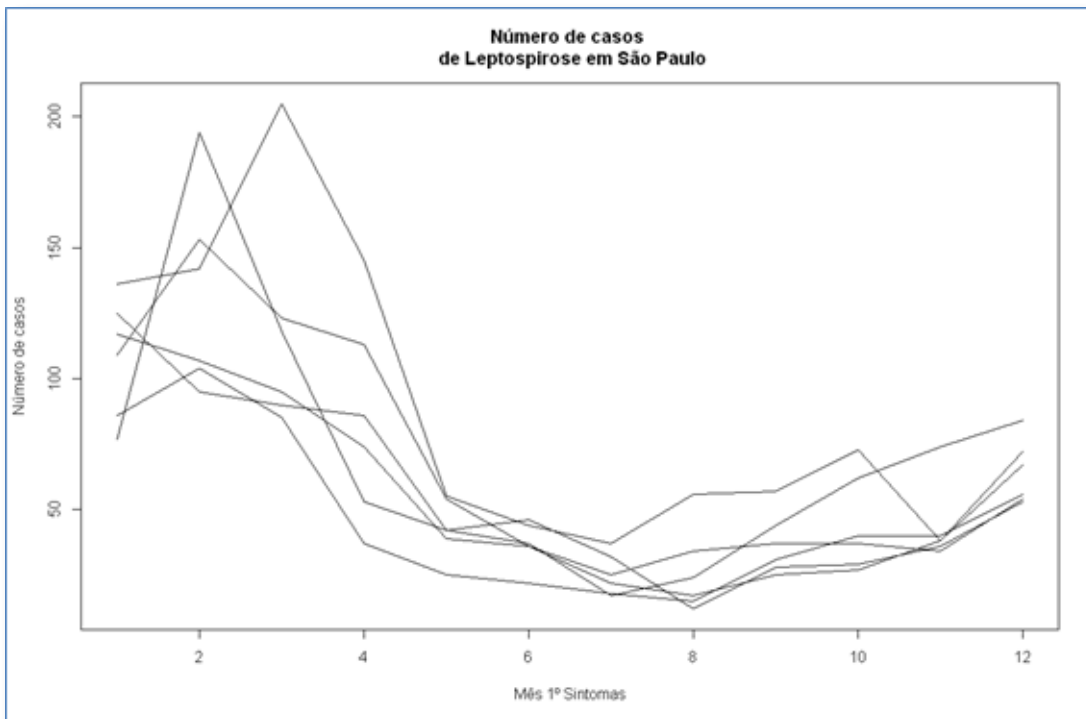
# gráfico preliminar das linhas anuais
plot (lepto_sp $ X2001[1:12], type='l', ylim=limy,
xlab='Mês 1º Sintomas', ylab='Número de casos',
main= 'Número de casos \n de Leptospirose em São Paulo')
```

```

points(lepto_sp $ X2002[1:12], type='l')
points(lepto_sp $ X2003[1:12], type='l')
points(lepto_sp $ X2004[1:12], type='l')
points(lepto_sp $ X2005[1:12], type='l')
points(lepto_sp $ X2006[1:12], type='l')

```

Segue o gráfico gerado no R:



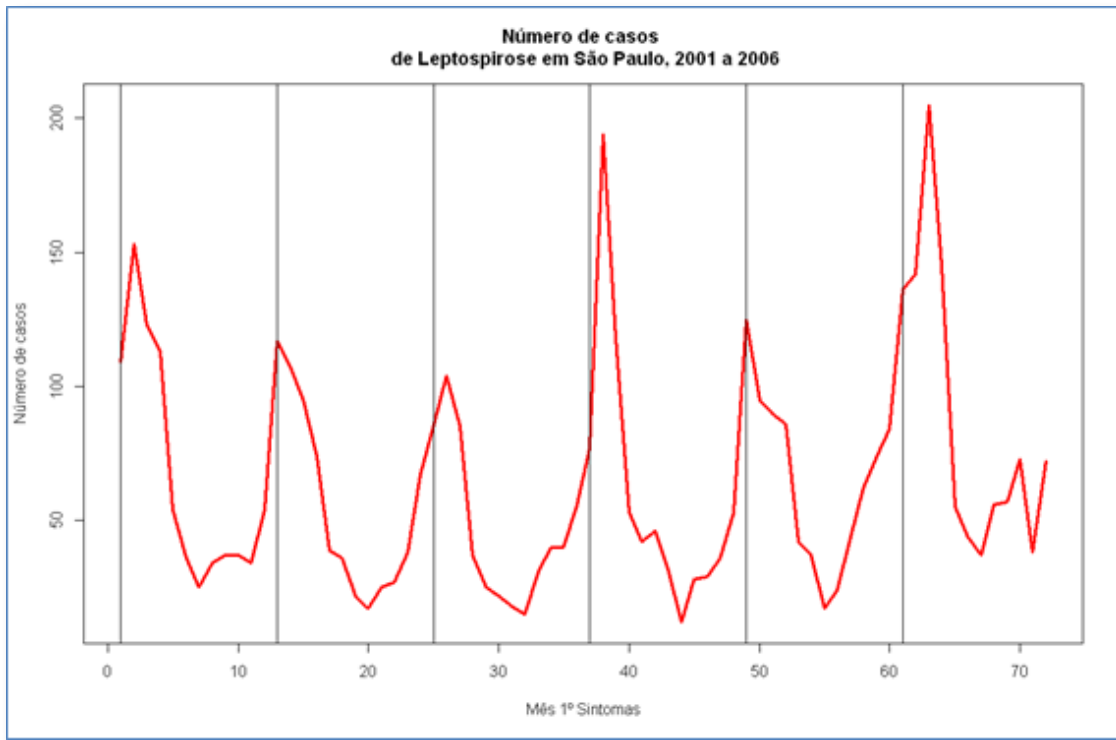
Na figura anterior confirma-se visualmente a sazonalidade anual apresentada pelos dados. Cada linha refere a um determinado ano (2001 a 2006). Segue comando em R para gerar outro gráfico juntando os seis anos. As linhas verticais indicam os meses de janeiro:

```

serie.hist = as.vector(as.matrix(dad))
plot(serie.hist, type='l',
     xlab='Mês 1º Sintomas', ylab='Número de casos', col='red', lwd=3,
     main='Número de casos \n de Leptospirose em São Paulo, 2001 a 2006')
abline(v=seq(1,61,12))

```

Segue o gráfico gerado no R:



A sazonalidade percebida visualmente no gráfico anterior pode ser descrita da seguinte forma: aumento expressivo do número de casos de Leptospirose no Estado de São Paulo no início e final dos anos, e forte diminuição no meio dos anos. Este comportamento verifica a hipótese de que esta série histórica possa estar correlacionada com as mudanças climáticas inverno/verão. Desta forma, precisa ser feito o cálculo Estatístico do p-valor (Kruskal-Wallis Rank Sum) para testar a possível variação sazonal na incidência desta doença. A finalidade com este cálculo é determinar se a sazonalidade é estatisticamente significativa. Segue programa em R:

```
# Kruskal-Wallis Rank Sum Test

serie = serie.hist # série histórica
periodo = 6 # em meses
alpha2 = .05 # significância

# número de pontos na série
n.serie = length (serie)
```

```

# número de grupos
n.grupos = as.integer(n.serie / periodo)
kruskal.g = NULL # inicializa o valor da variável
sazon = FALSE # inicializa o valor da variável
# loop gerando os grupos
for (ind.g in 1:n.grupos) {
  kruskal.g = c(kruskal.g, rep(ind.g, periodo))
}
# Kruskal-Wallis Rank Sum Test
temppv = kruskal.test (serie, kruskal.g) $ p.value
# p-valor
sazon.pv = ifelse (!is.na(temppv), round(temppv,6), 0.5)
# significância estatística
sazon = sazon.pv < alpha2 # 90% -> .1

# tem sazonalidade?
sazon

# p-valor
sazon.pv

```

Segue saída do R:

```

>
> # Kruskal-Wallis Rank Sum Test
> temppv = kruskal.test (serie, kruskal.g) $ p.value
> # p-valor
> sazon.pv = ifelse (!is.na(temppv), round(temppv,6), 0.5)
> # significância estatística
> sazon = sazon.pv < alpha2 # 90% -> .1
>
> # tem sazonalidade?
> sazon
[1] TRUE
>
> # p-valor
> sazon.pv
[1] 0.000643
>

```

Na saída anterior do R, verifica-se que a sazonalidade apresentada pelos dados é significativa, com p-valor 0.000643 (< 0.001).

Atividade 3

Exemplo com dados de óbitos em pessoas com 60 anos ou mais - A partir da Tabela 2 (Número de óbitos por semana, pessoas com 60 anos ou mais, residentes nos Estados do Ceará e Rio Grande do Sul, 2007-2009. Fonte: DATASUS) gere um arquivo CSV. Consulte a seção 1.9. Conversão de arquivos XLSX para CSV. A planilha criada deve iniciar com a linha (CE 491 434 478 ...) e finalizar com a linha (RS 941 937 884 ...). Ou seja, não usar a opção de cabeçalho (tabela sem nomes nas colunas).

Segue programa em R para a geração do primeiro gráfico a partir da tabela 2 em formato CSV:

Atividade 3, Módulo Séries Temporais

```
# Tabela 2. Número de óbitos por semana, pessoas com 60 anos ou mais,  
# residentes nos Estados do Ceará (CE) e Rio Grande do Sul (RS),  
# Fonte: DATASUS. 2007-2009.
```

```
# define a pasta de trabalho, que precisa ser previamente criada  
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida  
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta  
# verifique que os arquivos de interesse se encontrem nela  
dir()
```

```
# Carrega o arquivo texto CSV com os dados  
dad_ob = read.table (  
  file = 'Modulo_SerieTemporal_Atividade3_tab2ob.csv',  
  sep=';', header=FALSE,  
  stringsAsFactors=FALSE, dec='.')
```

```

n.lin = nrow(dad_ob)

# linhas por UF
ind.CE = which(dad_ob $ V1 == 'CE')
ind.RS = which(dad_ob $ V1 == 'RS')

# série por UF, sem 1ª coluna (nome das UF)
serie_ob.CE = as.matrix( dad_ob [ ind.CE, ] ) [,-1]
serie_ob.RS = as.matrix( dad_ob [ ind.RS, ] ) [,-1]

# traspondo a matriz e pegando as séries por coluna
serie_ob.CE = as.vector(t(serie_ob.CE))
serie_ob.RS = as.vector(t(serie_ob.RS))

# transformando para numérico
serie_ob.CE = as.numeric(serie_ob.CE)
serie_ob.RS = as.numeric(serie_ob.RS)

# retirando os NA
ind = which(!is.na(serie_ob.CE)) ; serie_ob.CE = serie_ob.CE[ind]
ind = which(!is.na(serie_ob.RS)) ; serie_ob.RS = serie_ob.RS[ind]

# mín e máx no eixo "Y"
limy = c(serie_ob.CE, serie_ob.RS)
limy2 = c(min(limy),max(limy))

# índice da primeira semana do ano
ind_sem1_ano = 0:2 *52 +1

# gráfico, linha UF CE
plot(serie_ob.CE, type='l', ylim=limy2,

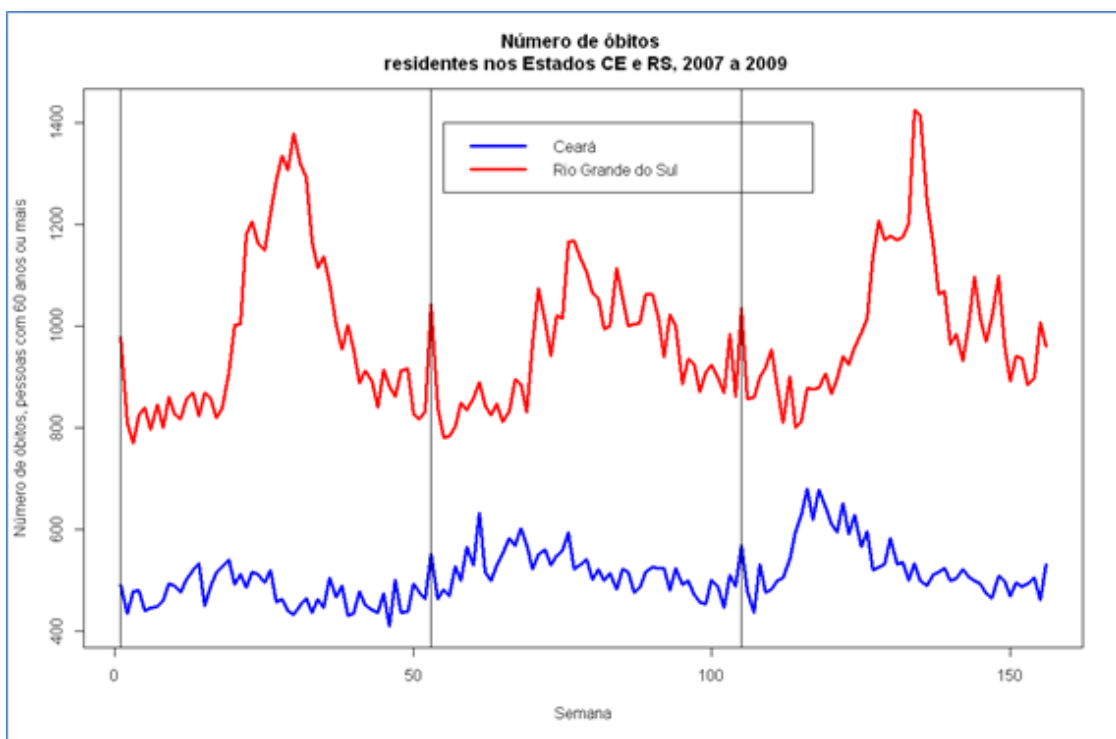
```

```

xlab='Semana', ylab='Número de óbitos, pessoas com 60 anos ou
mais', col='blue', lwd=3,
main='Número de óbitos \n residentes nos Estados CE e RS, 2007 a 2009')
# gráfico, linha UF RS
points(serie_ob.RS, type='l', col='red', lwd=3)
# linhas verticais na primeira semana dos anos
abline(v= ind_sem1_ano)
# especifica a legenda do gráfico
legend(55, 1400, c('Ceará', 'Rio Grande do Sul'),
lwd=3, col=c('blue','red'), lty=1)

```

Segue o gráfico com os valores observados gerado no R:



Para gerar o gráfico com a série alisada (médias móveis de ordem 4), execute os comandos abaixo no R:

```
## Média Móvel

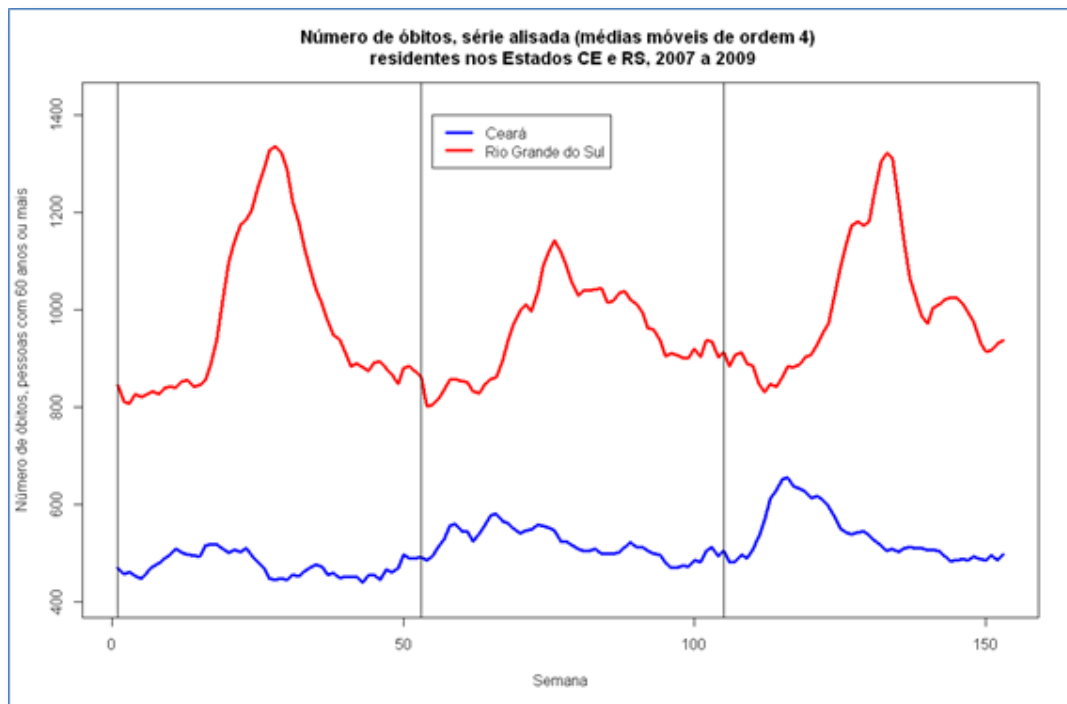
ser = serie_ob.CE
n.pto = length( ser )
res.mm.CE = apply ( matrix (
c( ser [1: (n.pto -3)],
ser [2: (n.pto -2)],
ser [3: (n.pto -1)],
ser [4: (n.pto  )]),
ncol=4), 1, mean)

ser = serie_ob.RS
n.pto = length( ser )
res.mm.RS = apply ( matrix (
c( ser [1: (n.pto -3)],
ser [2: (n.pto -2)],
ser [3: (n.pto -1)],
ser [4: (n.pto  )]),
ncol=4), 1, mean)

# gráfico, linha UF CE
plot(res.mm.CE, type='l', ylim=limy2,
xlab='Semana', ylab='Número de óbitos, pessoas com 60 anos ou
mais', col='blue', lwd=3,
main= 'Número de óbitos, série alisada (médias móveis de ordem 4)
\n residentes nos Estados CE e RS, 2007 a 2009')
# gráfico, linha UF RS
points(res.mm.RS, type='l', col='red', lwd=3)
# linhas verticais na primeira semana dos anos
abline(v= ind_sem1_ano)
```

```
# especifica a legenda do gráfico
legend(55, 1400, c('Ceará', 'Rio Grande do Sul'),
lwd=3, col=c('blue', 'red'), lty=1)
```

Segue o gráfico gerado no R:



Seguem os comandos em R para analisar se a sazonalidade apresentada visualmente nos gráficos anteriores é significativa:

```
periodo = 6 # em meses
alpha2 = .05 # significância
```

```
# função para avaliar a sazonalidade significativa da série
# Kruskal-Wallis Rank Sum Test
func.sazon = function(serie) {
# número de pontos na série
n.serie = length (serie)
# número de grupos
```

```

n.grupos = as.integer(n.serie / periodo)
kruskal.g = NULL # inicializa o valor da variável
sazon = FALSE # inicializa o valor da variável
# loop gerando os grupos
for (ind.g in 1:n.grupos) {
kruskal.g = c(kruskal.g, rep(ind.g, periodo))
}
# Kruskal-Wallis Rank Sum Test
temppv = kruskal.test (serie, kruskal.g) $ p.value
# p-valor
sazon.pv = ifelse (!is.na(temppv), temppv, 0.5)
# significância estatística
sazon = sazon.pv < alpha2 # 90% -> .1

saida = list()
# tem sazonalidade? TRUE ou FALSE
saida[[1]] = sazon
# p-valor
saida[[2]] = sazon.pv
# valor devolvido pela função:
# a) se tem sazonalidade
# b) p-valor
saida
}

# Número de óbitos, pessoas residentes
# com 60 anos ou mais, 2007 a 2009

# Ceará
func.sazon(serie_ob.CE)

# Rio Grande do Sul
func.sazon(serie_ob.RS)

```

```
# Ceará, série alisada (médias móveis de ordem 4)
func.sazon(res.mm.CE[4:153])
```

```
# Rio Grande do Sul,
# série alisada (médias móveis de ordem 4)
func.sazon(res.mm.RS[4:153])
```

Com o *script* anterior podemos avaliar a sazonalidade significativa nas séries analisadas. Foi criada a função **func.sazon** que dada uma determinada série histórica devolve duas informações: a) se a sazonalidade é significativa (TRUE ou FALSE), b) o p-valor associado.

Segue a saída no R do *script* anterior para as séries originais (sem alisamento):

```
>
> # Número de óbitos, pessoas residentes
> # com 60 anos ou mais, 2007 a 2009
>
> # Ceará
> func.sazon(serie_ob.CE)
[[1]]
[1] TRUE

[[2]]
[1] 3.084609e-11

>
> # Rio Grande do Sul
> func.sazon(serie_ob.RS)
[[1]]
[1] TRUE

[[2]]
[1] 4.621543e-15

>
```

Note que ambas as séries (UF CE e RS) apresentam sazonalidade significativa, p-valor < 0.0001.

Segue o resultado para as séries alisadas pelo método de médias móveis de ordem 4:

```
>
> # Ceará, série alisada (médias móveis de ordem 4)
> func.sazon(res.mm.CE[4:153])
[[1]]
[1] TRUE

[[2]]
[1] 5.414845e-17

>
> # Rio Grande do Sul,
> # série alisada (médias móveis de ordem 4)
> func.sazon(res.mm.RS[4:153])
[[1]]
[1] TRUE

[[2]]
[1] 1.777302e-17

>
```

Neste caso foram desconsiderados os primeiros três pontos das séries, e se inicia no quarto ponto ([4:154]) para ter bloco de 6 meses (período). Note que ambas as séries alisadas (UF CE e RS) também apresentam sazonalidade significativa, p-valor < 0.0001. Ou seja, confirma o resultado que já tínhamos. Neste caso específico o alisamento só nos trouxe uma melhor interpretação visual, pois as duas séries mesmo antes do alisamento já apresentavam diferenças significativas na sazonalidade.

Resoluções do Módulo 3

Análise de Dados Demográficos

Nesta seção apresentamos os *script* em R para solucionar as atividades do Módulo de “Análise de Dados Demográficos”.

Atividade 1

Certifique-se que os dados secundários necessários foram baixados do site do DATASUS <www.datasus.gov.br> e armazenados na **pasta de dados** (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

O passo a passo para obter os dados foi descrito no caderno de atividades do Módulo “Análise de Dados Demográficos”. Para organizar o trabalho sugerimos usar os seguintes nomes para os arquivos CSV:

1. SINASC_Regiao_x_TipoParto_ResMae_2010.csv

Para nascimentos por residência da mãe por Região (linha), tipo de parto (coluna), ano 2010.

2. SINASC_Regiao_x_TipoParto_OcorMae_2010.csv

Para nascimentos por local de ocorrência por Região (linha), tipo de parto (coluna), ano 2010.

3. SINASC_Regiao_x_ConsPreNatal_ResMae_2010.csv

Para nascimentos por residência da mãe por Região (linha), número de consultas no pré-natal (coluna), ano 2010.

4. SINASC_Regiao_x_ConsPreNatal_OcorMae_2010.csv

Para nascimentos por local de ocorrência por Região (linha), número de consultas no pré-natal (coluna), ano 2010.

Segue *script* em R:

```
### Atividade 1, Módulo Análise de dados demográficos.
```

```
# define a pasta de trabalho, que precisa ser previamente criada  
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida  
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta  
# verifique que os arquivos de interesse se encontrem nela  
dir()
```

```
# define uma função que carrega arquivos .CSV  
# com ela vamos poder carregar os nascimentos por região, res e ocor  
# são definidos valores padrão para os parâmetros  
read.dn = function (arquivo, separadorCol=';',  
cabecalho=TRUE, factors=FALSE,  
sepDecimal='.', pulaLinhaInicio=3,  
NoLinhaCarrega=6) {  
  read.table (file = arquivo, sep=separadorCol, header=cabecalho,  
stringsAsFactors=factors, dec=sepDecimal,  
skip=pulaLinhaInicio, nrows = NoLinhaCarrega)  
}
```

```
# para chamar a função "read.dn"  
# podemos ou não especificar os parâmetro  
# pois eles estão pré-definidos na definição da função  
# a leitura pula 3 linhas do cabeçalho do arquivo,  
# antes dos nomes das colunas  
# e depois dos nomes das colunas,
```

```

#carrega 6 linhas das Regiões e o Total
DN_tipoParto_res = read.dn (
  'SINASC_Regiao_x_TipoParto_ResMae_2010.csv',
  ';',TRUE, FALSE,',',',3,6)
DN_tipoParto_ocor = read.dn (
  'SINASC_Regiao_x_TipoParto_OcorMae_2010.csv')
DN_ConsPreNatal_res = read.dn (
  'SINASC_Regiao_x_ConsPreNatal_ResMae_2010.csv')
DN_ConsPreNatal_ocor = read.dn (
  'SINASC_Regiao_x_ConsPreNatal_OcorMae_2010.csv')

# nascimentos por res e ocor, Região vs Tipo de Parto
DN_tipoParto_res
DN_tipoParto_ocor
# nascimentos por res e ocor, Região vs Consultas de Pré Natal
DN_ConsPreNatal_res
DN_ConsPreNatal_ocor

# % de partos Cesário
procCesario.res = round( DN_tipoParto_res $ Cesário / DN_tipoParto_res
$ Total * 100, 1)
# % de partos Vaginal
procVaginal.res = round( DN_tipoParto_res $ Vaginal / DN_tipoParto_res
$ Total * 100, 1)
# % de tipoParto = Ignorado
procIgn.res = round( DN_tipoParto_res $ Ignorado / DN_tipoParto_res
$ Total * 100, 2)

result = data.frame(DN_tipoParto_res, procCesario.res,
procVaginal.res, procIgn.res)
# matriz com resultados
result

```

Segue a saída apresentada no R:


```
> result
      Região Vaginal Cesário Ignorado   Total procCesario.res procVaginal.res procIgn.res
1  Região Norte  178059  127991    372  306422      41.8      58.1      0.12
2  Região Nordeste  467032  372390   1738  841160      44.3      55.5      0.21
3  Região Sudeste  468520  654036   1037 1123593      58.2      41.7      0.09
4  Região Sul    154761  214926    218  369905      58.1      41.8      0.06
5  Região Centro-Oeste  93915  126691    182  220788      57.4      42.5      0.08
6      Total 1362287 1496034   3547 2861868      52.3      47.6      0.12
>
```

Atividade 2

Nesta seção trabalharemos com os dados PNAD, especificamente com os anos 2011, 2005 e 2001. Apresentaremos o resultado em R com a geração de gráficos a partir da Tabela 3.2 da PNAD - Taxa de analfabetismo das pessoas de 10 anos ou mais por Grandes Regiões, segundo idade e sexo.

Certifique-se que os dados secundários necessários tenham sido baixados do site do IBGE <<http://www.ibge.gov.br>>, como indicado neste exercício. Armazene os mesmos na **pasta de dados** (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

O passo a passo para obter os dados foi descrito no caderno de atividades do Módulo “Análise de Dados Demográficos”. Para organizar o trabalho sugerimos usar os seguintes nomes para os arquivos CSV:

1. PNAD_2009-2011_Educacao_tabela3_2.csv

Para os dados da PNAD - Educação em 2009 e 2011, Tabela 3.2.

2. PNAD_2004-2005_Educacao_tabela3_2.csv

Para os dados da PNAD - Educação em 2004 e 2005, Tabela 3.2.

Para a conversão dos arquivos para formato texto CSV, consulte a seção 1.9. Conversão de arquivos XLSX para CSV.

Segue *script* em R:

Atividade 2, Módulo Análise de dados demográficos.

```
# define a pasta de trabalho, que precisa ser previamente criada
pasta_dados = 'C:/Curso_EAD_ASA/'

# o R se posiciona na pasta definida
setwd (pasta_dados)

# com o comando "dir" pode ser visualizado o conteúdo da pasta
# verifique que os arquivos de interesse se encontrem nela
dir()

# dados da PNAD 2011, tabela 3.2
# carrega .CSV
tab32_2011 = read.table (file = 'PNAD_2009-2011_Educacao_tabela3_2.csv',
sep=';', header=FALSE,
stringsAsFactors=FALSE, dec='.')
# transforma de data.frame para matriz
tab32_2011 = as.matrix(tab32_2011)
# apaga os nomes das linhas e das colunas
dimnames(tab32_2011)=NULL

# posição onde inicia dados de 2011
lin.ini = which(tab32_2011 [,1] == '2011')
# bloco de 2011
tab32_2011_dad = tab32_2011 [(1:21) + lin.ini, 2:7]

# nomes das linhas (sexo) e colunas (regiões)
lin.descri = tab32_2011 [(1:21) + lin.ini, 1]
col.descri = tab32_2011 [10, 2:7]
col.descri [1] = 'Brasil'
```

```

# dados: Taxa de analfabetismo 2011 PNAD
tab32_2011_dad
data.frame(lin.descri) # linhas: Sexo
col.descri # colunas: Regiões

# configuração padrão do gráfico
par.mar0 = c(5, 4, 4, 2) + 0.1
# com margem inferior maior,
# para ter espaço para os nomes das faixas etárias
par.mar2 = c(7, 4, 4, 2) + 0.1

# função para geração do gráfico com 3 linhas
gera.graf3L.Br_M_F = function(lin1, lin2, lin3,
nomeEy, nomeG, nomeleg, reg)
# lin1 - vetor com dados do Total
# lin2 - vetor com dados do Homens
# lin3 - vetor com dados do Mulheres
# nomeEy - nome do eixo Y
# nomeg = nome geral = nome do gráfico
# nomeleg - nome da legenda
# reg - regiões
{

# mín e máx para o eixo Y
lim = c(lin1, lin2, lin3)
limym = max(lim)

# configuração do gráfico
par(col.axis='white', las=3, mar = par.mar2)
plot(1,1, col='white', ylim=c(0,limym),
xlim=c(1,6), xlab='', ylab=nomeEy)

# gera a 1ª linha, Total
points(1:6, lin1, type='l', lwd=3, col='dark green', x.axis=NULL)

```

```

points(1:6, lin1, lwd=3, col='dark green', pch=14, x.axis=NULL)

# 2da Linha, Homens
points(1:6, lin2, type='l', lwd=3, col='blue', x.axis=NULL)
points(1:6, lin2, lwd=3, col='blue', pch=15, x.axis=NULL)

# 3ra Linha, Mulheres
points(1:6, lin3, type='l', lwd=3, col='red', x.axis=NULL)
points(1:6, lin3, lwd=3, col='red', pch=16, x.axis=NULL)

# faz a legenda no eixo "Y"
par(col.axis='black', las=3, mar = par.mar2 )
ya = seq(0,limym,1)
n.axis <- length(ya)
for (i in 1:n.axis) {
axis(2, ya[i], ya[i])
}

# legenda no eixo "X"
x.axis <- reg
n.axis <- length(x.axis)
for (i in 1:n.axis) {
axis(1, i, x.axis[i])
}

# especifica a legenda do gráfico
legend(4, max(ya), nomeleg,
lwd=3, col=c('dark green','blue','red'), lty=1, pch=c(14,15,16))

# nome do gráfico
title(nomeG)
}

# Tabela 3.2 , PNAD

```

```
tab32.nome = 'Taxa de analfabetismo das pessoas de 10 anos ou mais de idade'
```

```
### Cada bloco abaixo gera um determinado gráfico ###
```

```
### Executar cada um deles separadamente ###
```

```
fx_etaria = 'Total'
```

```
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[1,]),  
as.numeric(tab32_2011_dad [2,]), # conversão de stirng para numérico  
as.numeric(tab32_2011_dad [3,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri  
)
```

```
fx_etaria = '10 a 14 anos'
```

```
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[4,]),  
as.numeric(tab32_2011_dad [5,]),  
as.numeric(tab32_2011_dad [6,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri  
)
```

```
fx_etaria = '15 anos ou mais'
```

```
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[7,]),  
as.numeric(tab32_2011_dad [8,]),  
as.numeric(tab32_2011_dad [9,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri
```

)

```
fx_etaria = '15 a 17 anos'  
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[10,]),  
as.numeric(tab32_2011_dad [11,]),  
as.numeric(tab32_2011_dad [12,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri  
)
```

```
fx_etaria = '15 a 24 anos'  
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[13,]),  
as.numeric(tab32_2011_dad [14,]),  
as.numeric(tab32_2011_dad [15,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri  
)
```

```
fx_etaria = '18 anos ou mais'  
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[16,]),  
as.numeric(tab32_2011_dad [17,]),  
as.numeric(tab32_2011_dad [18,]),  
tab32.nome,  
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),  
c('Total', 'Homens', 'Mulheres'),  
col.descri  
)
```

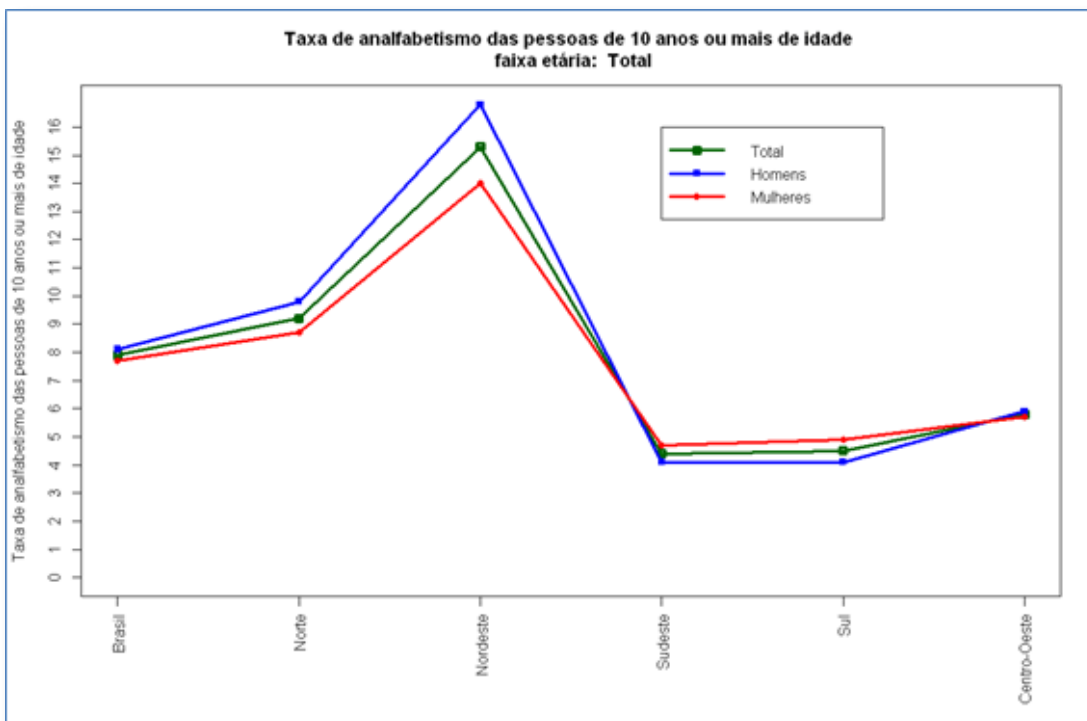
```
fx_etaria = '25 anos ou mais'  
gera.graf3L.Br_M_F( as.numeric(tab32_2011_dad[19,]),
```

```

as.numeric(tab32_2011_dad [20,]),
as.numeric(tab32_2011_dad [21,]),
tab32.nome,
paste (tab32.nome, '\n', 'faixa etária: ', fx_etaria),
c('Total', 'Homens', 'Mulheres'),
col.descri
)

```

Segue exemplo de um dos gráficos gerados no R:



Atividade 3

Os dados solicitados podem ser extraídos do site do DATASUS <www.datasus.gov.br> e armazenados na **pasta de dados** (seção 1.5. Criando uma pasta de trabalho para o R). Caso a pasta de dados não tenha sido criada no caminho ou com nome indicado, alterar o *script* abaixo na linha:

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

Seguem os dados da População residente - Brasil, por sexo e faixa etária detalhada, período 2010.

FxEtaria	Homens	Mulheres
0 a 4 anos	7016987	6779172
5 a 9 anos	7624144	7345231
10 a 14 anos	8725413	8441348
15 a 19 anos	8558868	8432002
20 a 24 anos	8630227	8614963
25 a 29 anos	8460995	8643418
30 a 34 anos	7717657	8026855
35 a 39 anos	6766665	7121916
40 a 44 anos	6320570	6688797
45 a 49 anos	5692013	6141338
50 a 54 anos	4834995	5305407
55 a 59 anos	3902344	4373875
60 a 64 anos	3041034	3468085
65 a 69 anos	2224065	2616745
70 a 74 anos	1667373	2074264
75 a 79 anos	1090518	1472930
80 anos e mais	1133122	1802463

Fonte: IBGE - Censos Demográficos.

Para a conversão dos arquivos para formato texto CSV, consulte a seção 1.9. Conversão de arquivos XLSX para CSV. Sugerimos o seguinte nome para o arquivo texto: **Modulo_Demografia_Atividade3.csv**

Segue o *script* em R:

Atividade 3, Módulo Análise de dados demográficos

```
# define a pasta de trabalho, que precisa ser previamente criada
```

```
pasta_dados = 'C:/Curso_EAD_ASA/'
```

```
# o R se posiciona na pasta definida
```

```
setwd (pasta_dados)
```

```
# com o comando "dir" pode ser visualizado o conteúdo da pasta
```

```
# verifique que os arquivos de interesse se encontrem nela
```

```
dir()
```

```
# dados de população residente Brasil 2010, por sexo e faixa etária
```

```
# fonte: IBGE
```

```
# carrega .CSV
```

```
pop_2010_Br = read.table (file = 'Modulo_Demografia_Atividade3.csv',  
  sep=';', header=TRUE,  
  stringsAsFactors=FALSE, dec='.')
```

```
# nome das faixas etárias
```

```
faixas_nome = pop_2010_Br $ FxEtaria
```

```
n.fxs = length(faixas_nome) # número de faixas etárias
```

```
# Homens
```

```
pop_H = pop_2010_Br $ Homens
```

```
maxH = max(pop_H)
```

```
# Mulheres
```

```
pop_M = pop_2010_Br $ Mulheres
```

```
maxM = max(pop_M)
```

```
# configuração padrão do gráfico
```

```

par.mar0 = c(5, 4, 4, 2) + 0.1
# com margem esquerda (eixo Y = faixa etária) maior,
# para ter espaço para os nomes das faixas etárias
par.mar2 = c(5, 7, 4, 2) + 0.1

# escala da população no gráfico: em milhões
escala = 1000000

# configuração do gráfico
par(col.axis='white', las=1, mar = par.mar2)
plot(0,1, col='white', ylim=c(1,n.fxs),
     xlim=c(maxH *(-1), maxM), xlab='', ylab='')

# loop nas faixas etárias, que geram as linhas da pirâmide
for (i in 1:n.fxs) {
  # linha da pirâmide, lado esquerdo = homem
  points(c(-0.1*escala, pop_H[i] *(-1)), c(i,i), type='l', lwd=7,
        col='blue', x.axis=NULL)
  # linha da pirâmide, lado direito = mulher
  points(c(0.1*escala, pop_M[i]), c(i,i), type='l', lwd=7, col='red',
        x.axis=NULL)
}
abline(v=0) # eixo x=0

# faz a legenda no eixo "Y"
par(col.axis='black', las=1, mar = par.mar2 )
ya = faixas_nome
n.axis <- length(ya)
for (i in 1:n.axis) {
  axis(2, i, ya[i])
}

# nome do gráfico
title('Pirâmide etária - Brasil 2010 \n População (em milhões de

```

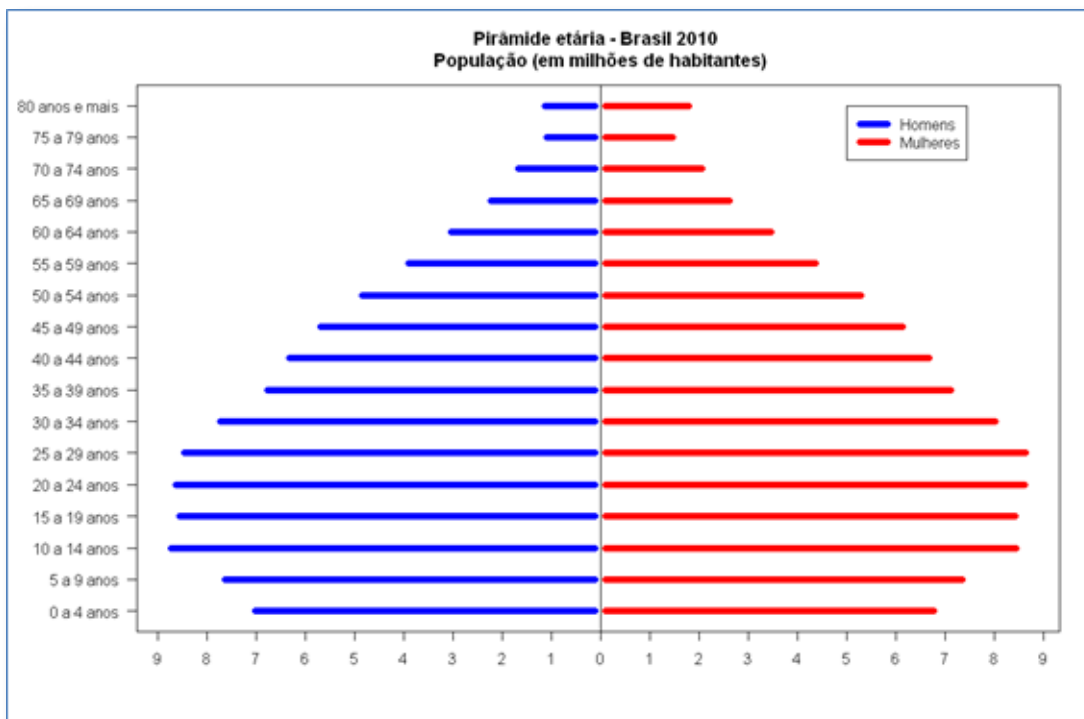
```
habitantes)')
```

```
# especifica a legenda do gráfico  
legend(5*escala, n.fxs, c('Homens','Mulheres'),  
lwd=7, col=c('blue','red'), lty=1)
```

```
# legenda no eixo "X"
```

```
x.axis <- seq(-9,9,1) * escala  
n.axis <- length(x.axis)  
for (i in 1:n.axis) {  
axis(1, x.axis[i], abs(x.axis[i])/escala)  
}
```

Segue o gráfico com a pirâmide etária gerada no R:



Considerações Gerais

Para a resolução dos exercícios dos demais Módulos do Curso EAD, não apresentados neste volume, o aluno poderá desenvolver os comandos no R, com base nos *scripts* apresentados.

No R o padrão para a pontuação numérica é a seguinte: “,” para separador de milhar, e “.” para separador decimal. Para manter a notação uniforme neste volume foi mantido o padrão do R. Caso seu Excel esteja configurado no padrão brasileiro (português), nos casos que o comando **read.table** (leitura de arquivo CSV) foi usado, troque `dec=','` por `dec='.'`

Referências

1. R Statistical Software, <<http://www.r-project.org/>>
2. Pedro Morettin e Clélia M. C. Toloi Análise de Séries Temporais. Blucher 2009, ABE – Projeto Fisher.
3. Pedro A. Morettin e Wilton O. Bussab Estatística Básica. Saraiva, 2010.
4. Introdução ao uso do programa R. Victor Lemes Landeiro, Ago/2011. <<http://cran.r-project.org/doc/contrib/Landeiro-Introducao.pdf>>
5. Introdução à Programação em R. Luis Torgo, Out/2006. <<http://cran.r-project.org/doc/contrib/Torgo-ProgrammingIntro.pdf>>
6. Tópicos de Estatística utilizando R. Fernando Itano, Ago/2007. <<http://cran.r-project.org/doc/contrib/Itano-descriptive-stats.pdf>>
7. Bioestatística usando R. Colin Robert Beasley, 2004. <<http://cran.r-project.org/doc/contrib/Beasley-BioestatisticaUsandoR.pdf>>

8. Mardia K, Kent J and Bibby J. *Multivariate Analysis*. New York: Academic Press, 1979.
9. Johnson R and Wichern D. *Applied Multivariate Statistical Analysis*. New Jersey: Prentice-Hall, 1995.
10. Kohonen T. *Self-Organizing Maps*. Springer, 1997.
11. Tibshirani R, Walther G and Hastie T. Estimating the number of clusters in a dataset via the Gap statistic. *JRSSB*, March 2000.