# Aula 4 - Importação e exportação de dados e uso de bibliotecas no R

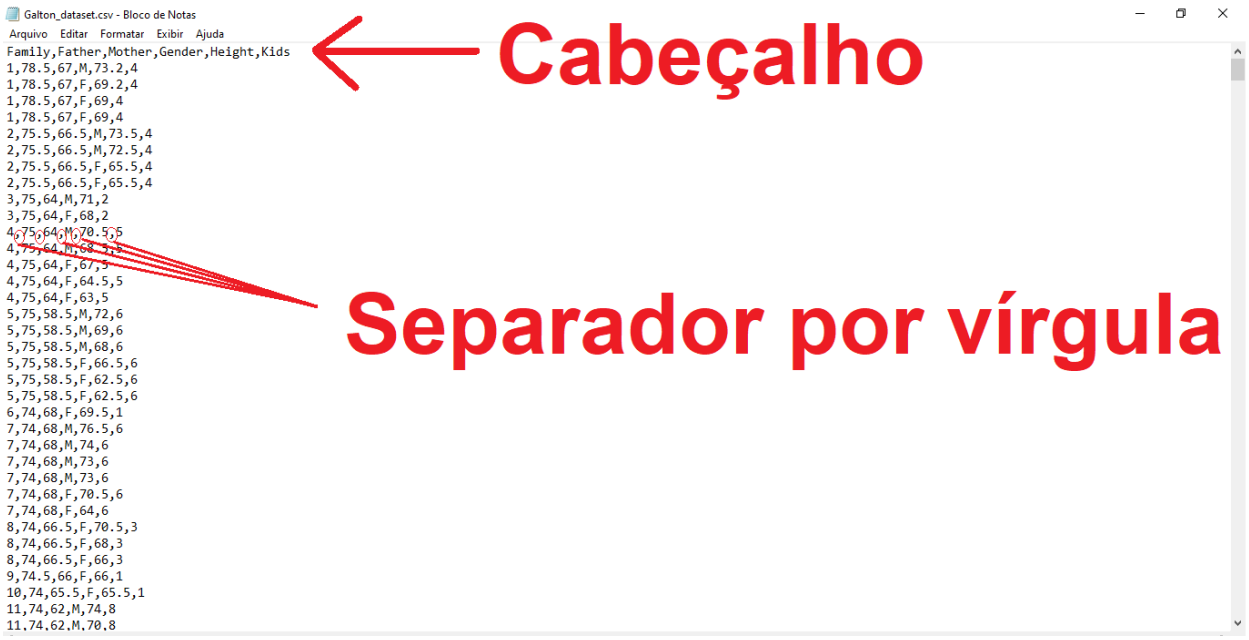
Eduardo Koerich Nery

Para desenvolver projetos, nós normalmente vamos utilizar dados que foram armazenados em outros formatos fora do R. Assim, temos que aprender a importar dados para a plataforma R. Além disso, nós normalmente vamos precisar executar tarefas que não estão originalmente implementadas na plataforma R. Contudo, essas tarefas podem estar disponíveis em uma **biblioteca**. As bibliotecas são um compilado de funções elaboradas por alguém.

### Importação de dados

A maioria dos dados é armazenada de forma tabular, ou seja, em linhas e em colunas. A função para importar dados tabulares é `read.table()`. Essa função possui diferentes argumentos, mas aqui nós vamos explorar os argumentos de uso mais rotineiro.

Primeiro vamos analisar o conjunto de dados `Galton_dataset.csv`. Abra esse conjunto de dados no bloco de notas,note que a primeira linha possui o nome das variáveis, formando um cabeçalho. Além disso, você verá que as colunas de dados estão separadas por vírgula `,`, por isso a extensão do arquivo é *.csv* (*comma-separated values*, ou seja, valores separados por vírgula). Por último, você verá que os valores numéricos decimais são sinalizados pelo ponto `.`.




Assim, para importar esse conjunto de dados, nós usamos o comando:

```
read.table(file="Galton_dataset.csv", header=T, sep=",", dec=".")
```

O argumento `file` indica qual arquivo queremos importar; `header=T` indica que os dados possuem cabeçalho (`T` é abreviação de *TRUE*); `sep=","` indica que o separador das colunas é a vírgula `,`; `dec="."` indica que os valores numéricos decimais ocorrem após o ponto `.`.

Agora, vamos analisar o conjunto de dados `Fisher_dataset.txt`. Abra esse conjunto de dados no bloco de notas, note que **não** existe uma linha nomeando as colunas, e que as colunas estão separadas por tabulação. Além disso, verá que os valores numéricos decimais ocorrem depois da vírgula `,`. Por último, você verá que existem asteriscos `*` em algumas posições. O asterisco foi colocado para indicar dados faltantes, ou seja, dados que não foram coletados para aquela observação (linha). *Qualquer outro símbolo ou expressão poderia ter sido usado para indicar os dados faltantes.*


Assim, para importar esses dados, nós utilizamos o comando:

```
read.table(file="Fisher_dataset.txt", header=F, sep="", dec=",", na.strings="*")
```

Onde o argumento `file` indica o arquivo que queremos importar; `header=F` indica que os dados **não** possuem cabeçalho (`F` é abreviação de *FALSE*); `sep=""` indica que o separador das colunas é por tabulação; `dec=","`indica que os valores numéricos decimais ocorrem após a vírgula `,`; `na.strings="*"` indica que temos dados faltantes e que os mesmos foram sinalizados por um asterisco `*`.

Ainda, podemos importar dados de forma interativa. Para isso, você deve usar a função `file.choose()` como valor do argumento `file` na função 'read.table()".

```
read.table(file=file.choose(), header=F, sep="", dec=",", na.strings="*")
```

A prática mais recomendada é definir um diretório de trabalho contendo os arquivos que serão usados no projeto, ao invés de usar a escolha interativa. Uma vez definido o diretório de trabalho, você pode verificar todos os arquivos dentro do mesmo com o comando `list.files(getwd())`.

Por fim, podemos importar dados direto de algum repositório *online* se o computador tiver acesso a internet. Para usar conjunto de dados *online*, nós incluimos o endereço eletrônico dos dados como caracteres entre aspas `""` no argumento `file`. Por exemplo, visualize o conjunto de dados desta página. Estes dados são sobre a atividade sexual e longevidade de moscas de frutas. Repare que os dados não possuem cabeçalho e estão separados por espaço.


Assim, para importar esses dados, nós utilizamos o comando:

```
read.table(file="http://jse.amstat.org/datasets/fruitfly.dat.txt", header=F, sep="")
```

**NOTA**: A interface amigável popularizou o EXCEL da Microsoft como o programa de armazenamento e gerenciamento de dados. Contudo, os formatos *.xls* ou *.xlsx* usados pelo EXCEL não tem compatibilidade pronta com o R, necessitando pacotes específicos para serem importados. Mesmo assim, os formatos *.xls* ou *.xlsx* podem conter caracteres ocultos que dificultam a leitura pelo R. Então, eu sugiro salvar suas planilhas de dados em formato *.csv* ou *.txt* para facilitar a importação para o R e outras plataformas de programação.

### Exportação de dados

A exportação de dados em formato tabular é feita pela função `write.table()`. Essa função possui diferentes argumentos, mas aqui vamos explorar os mais rotineiros. A exportação sempre acontecerá no diretório de trabalho do R caso um endereço não seja especificado.

Por exemplo, eu criei uma tabela de dados simples com alguns dados sobre algumas pessoas. A tabela possui nomes tanto nas linhas quanto nas colunas.

```
tabela<-data.frame(c(1.68,1.84,1.72),c(72,95,67),c("F","M","F"))

rownames(tabela)<-c("Ana","Pedro","Maria")

colnames(tabela)<-c("alturas","massas","sexo")

tabela
```

Para exportar essa tabela, nós podemos utilizar o comando:

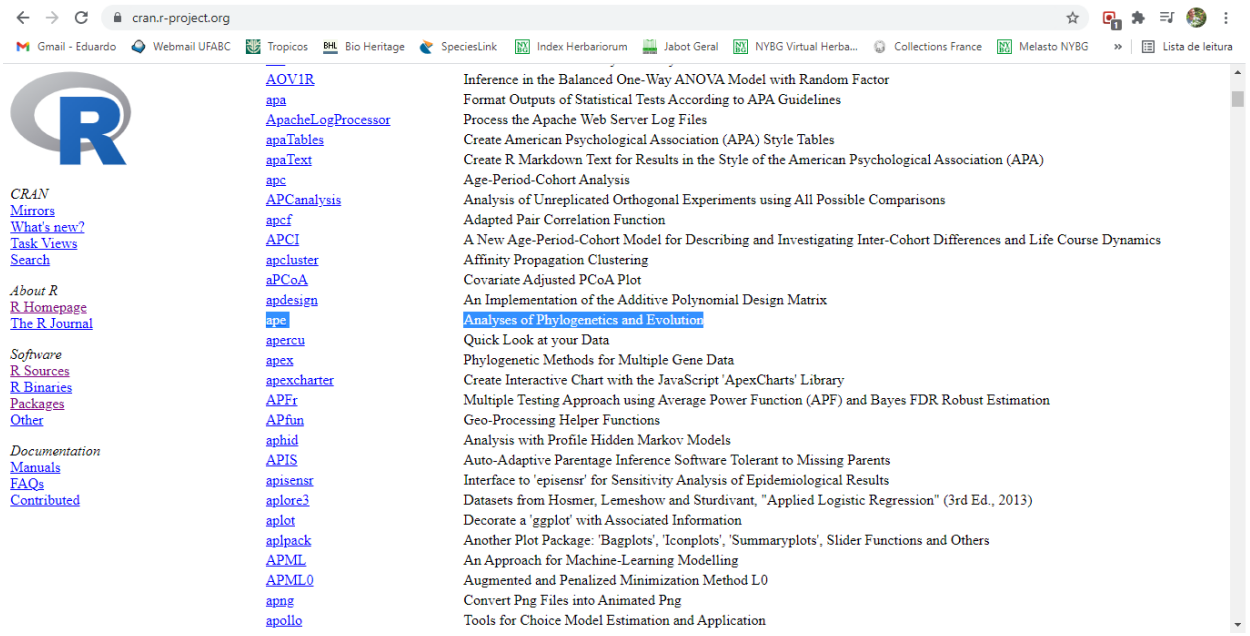
```
write.table(x=tabela, file="tabela.csv", sep=",", quote=F, row.names=T, col.names=T)
```

Onde `x=tabela` indica os dados do R que serão exportados; `file="tabela.csv"` indica o nome e a extensão do arquivo que vai ser exportado; `sep=","` indica que as colunas de dados serão separadas por vírgula; `quote=F` indica que os dados **não** devem ser exportados como texto (ou seja, entre `""`); `row.names=T` indica que o nome das linhas deve ser armazenado; `col.names=T` indica que o nome das colunas deve ser armazenado. O resultado deste comando foi um arquivo chamado *tabela.csv* no meu diretório de trabalho. Abra o arquivo recém-exportado!

### Instalação e uso de bibliotecas

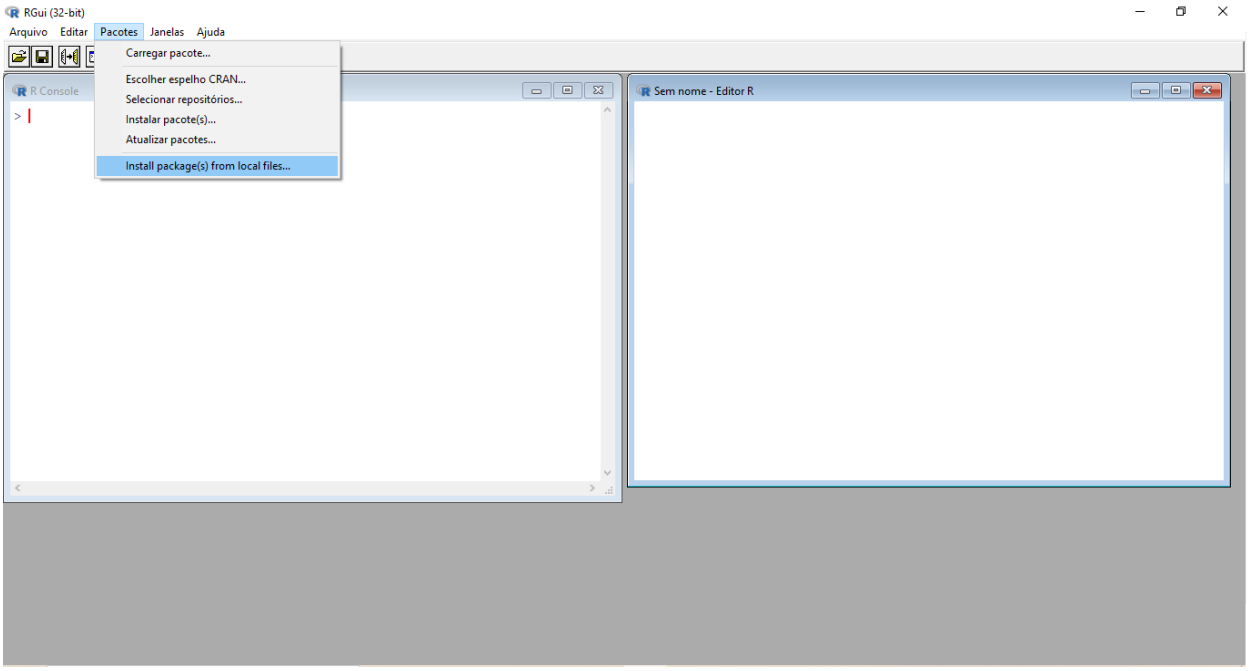
As bibliotecas são um conjunto de funções e, as vezes, estruturas de dados desenvolvidas por outras pessoas. A maioria das bibliotecas está armazenada no repositório CRAN, sob a aba "*Packages*", opção "*Table of available packages, sorted by name*". As bibliotecas também são chamadas de "pacotes" pois elas expdandem as funcionalidades do R. Por exemplo, o R originalmente **não** consegue interpretar e manipular árvores filogenéticas, que normalmente são armazenadas em formato parentético (Newick, extensão *.nwk*). Para tornar possível a leitura de árvores filogenéticas, nós precisamos instalar e ativar a biblioteca `ape`. Podemos proceder de duas formas:

1) Nós podemos baixar o código fonte da biblioteca e instalar "manualmente" no R. Por exemplo, para baixar o pacote `ape`, nós podemos ir até a lista de "pacotes" do CRAN e clicar na opção *ape* (*Analyses of Phylogenetics and Evolution*).

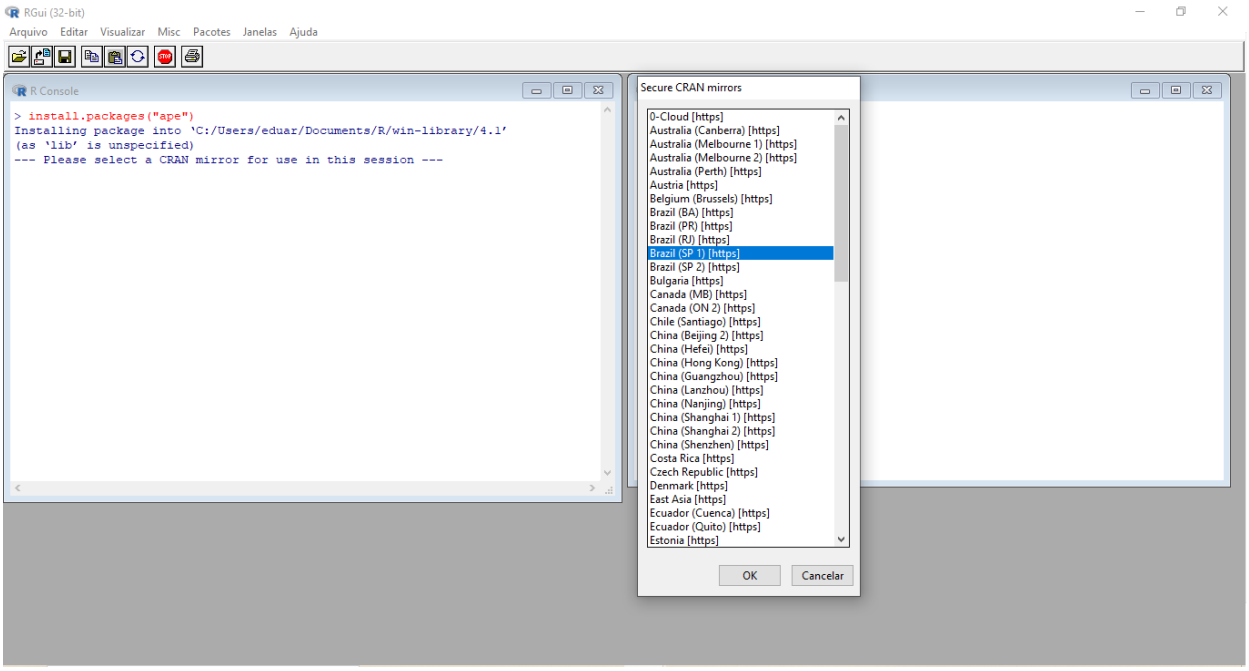



Na página da biblioteca, nós vamos até as opções de *downloads*, onde podemos ver o código fonte do pacote ("*Package source*", extensão *.gz*) e um manual para o uso do pacote ("*Reference manual*",extensão *.pdf*). Para baixar estes arquivos no seu computador, basta clicar no nome dos mesmos.


Agora na plataforma R, nós vamos até a aba "Pacotes" e clicamos na opção "*Install packages from local file...*". Isto abrirá uma janela para buscar arquivos no seu computador. Procure o arquivo com o código fonte (extensão *.gz*) e clique no mesmo. Com isso, o R começará a compilar o código da biclioteca.




2) Nós podemos usar o R para procurar o código fonte. Por exemplo, para instalar o mesmo pacote `ape`, nós podemos digitar no console do R `install.packages("ape")`. Isto abrirá uma nova janela que mostra os servidores disponíveis para procurar o código fonte. (Servidores são apenas outros computadores que estão armazenando/trocando informação em rede). Procure o servidor mais perto de você e clique em "*OK*".



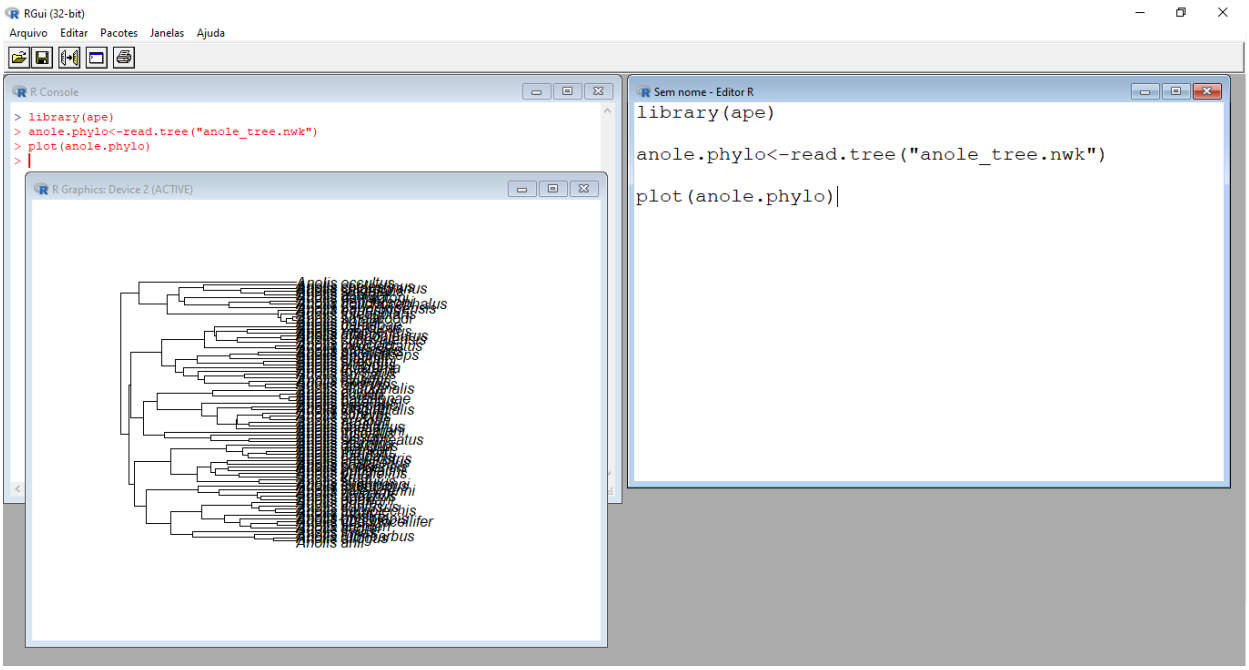

Após a instalação, nós precisamos ativar a biblioteca com a função `library()`, inserindo dentro dos parênteses o nome da biblioteca desejada.

Por exemplo, vamos ativar a biblioteca 'ape' e em seguida utilizar funções para importar e visualizar uma árvore filogenética:

```
library("ape")

anole.phylo<-read.tree("anole_tree.nwk")

plot(anole.phylo)
```



**Não esqueça de citar!** Por último, mas não menos importante, não esqueça de citar as bibliotecas utilizadas. Caso tenha dúvida de como citar, utilize a função `citation()`, inserindo dentro dos parênteses o nome da biblioteca de interesse entre aspas `""`. A função retorna a referência pronta e também o formato *BibTeX* para ser incluído em programas de gerenciamento de referências (e.g., Mendeley e Zotero).