

# Aula 02 – Mapas Temáticos – Parte 1

## Sumário

<b>Aula 02 – Mapas Temáticos – Parte 1 .....</b>	<b>1</b>
<b>1. Apresentação .....</b>	<b>2</b>
<b>2. Pacotes mais utilizados.....</b>	<b>3</b>
<b>3. Carregando objetos espaciais do R .....</b>	<b>5</b>
<b>4. Combinando dados de objetos espaciais e tabelas.....</b>	<b>10</b>
<b>5. Elaborando um mapa de taxas de incidência .....</b>	<b>12</b>
<b>6. Referências Bibliográficas .....</b>	<b>15</b>
<b>7. Exercícios .....</b>	<b>17</b>

## 1. Apresentação

Olá! Seja bem-vindo ao segundo capítulo do curso de “R aplicado a Análises Espaciais em Saúde”!

Nosso objetivo é promover o uso do R, uma linguagem de programação de alto nível, para análises espaciais em Saúde. Continuamos usando a interface de desenvolvimento do R, RStudio. Isso facilita a escrita dos nossos *scripts* e a visualização de *datasets* e mapas.

Em nosso segundo e terceiro capítulo veremos os mapas temáticos. Nessa primeira parte, abordaremos os pacotes *ggplot2*, *ggspatial* e o *maptools*. Vamos construir nossos primeiros mapas.

Antes de iniciar, você deve ter baixado o arquivo compactado “curso\_r\_geoanalise”. Após descompactar o arquivo, clique no projeto R com o nome “curso\_r\_geonanalise.Rproject”. Esse é o arquivo projeto do nosso curso, você deverá abri-lo e depois selecionar o *script* “02\_mapas\_tematicos\_1.R”.

Abrindo pelo Rproject, você fica com todas as pastas alinhadas e terá acesso a todos os arquivos que usaremos nas aulas sem dificuldade. Embarque conosco mais uma vez e vamos continuar nessa jornada! Que seu aprendizado seja muito útil no SUS!

## 2. Pacotes mais utilizados

Nos módulos anteriores, apresentamos o conceito de *packages* ou *libraries* (pacotes ou bibliotecas, em tradução livre). Apenas para recapitular, vamos lembrar que os *packages* são conjuntos de funções para finalidades específicas que podem ser instalados no R. Por ser uma linguagem de código aberto, o R permite que qualquer desenvolvedor crie pacotes que podem ser instalados e modificados por qualquer usuário do R.

Assim como para a análise de dados e criação de gráficos, existem pacotes voltados para análise geoespacial. Nesse capítulo, usaremos alguns desses pacotes e alguns pacotes auxiliares que nos ajudarão em análises espaciais e na construção de indicadores de saúde, sendo o nosso foco as taxas de incidência por estado.

Conversamos anteriormente sobre a instalação de pacotes que ocorre por meio do CRAN (*The Comprehensive R Archive Network* [Rede Abrangente de Arquivos R, em tradução livre]), que é o repositório oficial de pacotes do R. Para que os pacotes sejam hospedados no CRAN, é necessário que eles sigam critérios de segurança e usabilidade específicos. A função “*install.packages()*” direciona o R para realizar o *download* de pacotes diretamente do CRAN (1).

É possível que você encontre pacotes fora o CRAN e eles não serão instalados por meio da função “*install.packages()*”. Uma forma de instalar esses pacotes é utilizando as funções da biblioteca *devtools*. A forma mais comum é carregando pacotes de algum repositório *online*, sendo o mais popular deles o *GitHuB*.

Inicialmente, vamos instalar alguns pacotes importantes para os nossos exercícios. Vamos instalar 3 pacotes:

- *devtools*: pacote utilizado para conceder ferramentas de desenvolvedor ao usuário do R. Nesse capítulo, usaremos o pacote para ensinar uma forma de instalação alternativa da biblioteca *geobr* (2).
- *geobr*: é um pacote para *download* oficial de *datasets* espaciais do Brasil. Esse pacote possui diversos arquivos geoespaciais em um formato de *geopackage*, que é análogo aos arquivos *shapefile*, mas

que apresentam uma variedade de dados que possibilitam análises espaciais mais completas. Esse pacote está disponível para o R e para o Python, porém foi recentemente removido do CRAN. Embora possamos baixar a versão arquivada do pacote, podemos baixar a versão atualizada diretamente do repositório do GitHub (3).

- *pacman*: esse pacote automatiza a instalação de pacotes do R e, ao mesmo tempo, carrega os pacotes que precisamos usar naquele *script*(4).

Para instalar esses pacotes, vamos utilizar os seguintes comandos:

```
#INSTALANDO OS PACOTES UTILIZADOS NA AULA

##INSTALANDO PACOTES DIRETAMENTE DO CRAN
install.packages("devtools", dependencies = T) #ferramentas de
desenvolvedor do R

install.packages("geobr", dependencies = T) #pacote de geoanálise de
arquivos do Brasil

install.packages("pacman", dependencies = T) #pacote para instalação e
carregamento de outros pacotes)

##INSTALANDO PACOTE GEOBR DIRETAMENTE DO GITHUB
devtools::install_github("ipeaGIT/geobr", subdir = "r-package", force
= T) #instalando geobr do GitHub, caso a instalação do CRAN não tenha
funcionado
```

Após essa instalação, vamos carregar todos os outros pacotes que utilizaremos na aula, utilizando o pacote *pacman* e a função “*p\_load*” (5–18). A função “*p\_load*” verifica se os pacotes já estão instalados em seu computador. Para os que não estão instalados, ele faz a instalação e carrega todos os pacotes de uma única vez.

```
##INSTALANDO E CARREGANDO PACOTES COM O PACMAN
library(pacman)
p_load(
  geobr,          #pacote com datasets geoespaciais do Brasil
  ggsn,           #símbolos de direção e barras de escala para
pacotes #criados com "ggplot2" ou "ggmap"
  ggspatial,      #dados espaciais de suporte para o pacote "ggplot2"
  gpclib,         #tratamento de feições de polígonos/áreas para o R
  jsonlite,       #leitor e gerador de arquivos no formato JSON
  leaflet,        #versão do R para a biblioteca Leaflet do
Javascript #para mapas interativos
  lubridate,      #pacote para tratamento de datas no R
  maptools,       #manipulação de dados geográficos
  RColorBrewer,   #paletas de cor para visualizações do R
  RCurl,          #interface de client para o R (usado para simular
#visualizações web)
  rgdal,          #funções de integração para a biblioteca gdal
```

```

rgeos,          #operações com topologia e geometrias
rio,            #importação e exportação de arquivos variados
rjson,         #conversão de arquivos JSON
tidyverse      #conjunto de pacotes de manipulação e tratamento de
#dados e criação de gráficos
)

```

### 3. Carregando objetos espaciais do R

Após carregar os pacotes, vamos importar alguns objetos espaciais para o R. Além da forma gráfica (o desenho em si), os objetos espaciais possuem atributos, registros de características de cada uma de suas feições espaciais, e uma projeção.

Antes de importarmos os pacotes, vamos usar a função “*list\_geobr()*” para identificar quais são as funções disponíveis no pacote *geobr*. Para isso, vamos criar um objeto chamado “*funcoes\_geobr*” que vai receber a lista de todas as funções do pacote *geobr*.

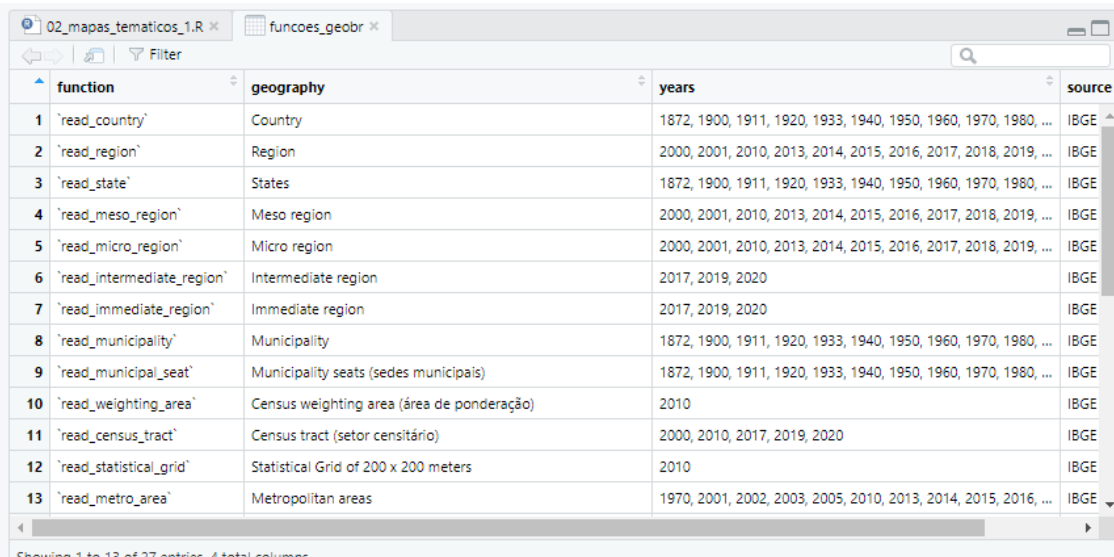
```

#IMPORTANDO OBJETOS ESPACIAIS

##VERIFICANDO FUNÇÕES DISPONÍVEIS NO PACOTE GEOBR
funcoes_geobr = list_geobr()
View(funcoes_geobr)

```

A função “*View*” nos permitirá ver o objeto “*funcoes\_geobr*”, que é um *data.frame*, ou seja, um objeto em forma de tabela em que as variáveis são representadas por colunas e as observações por linhas. O objeto será visualizado no primeiro quadro esquerdo do RStudio, em uma aba da mesma

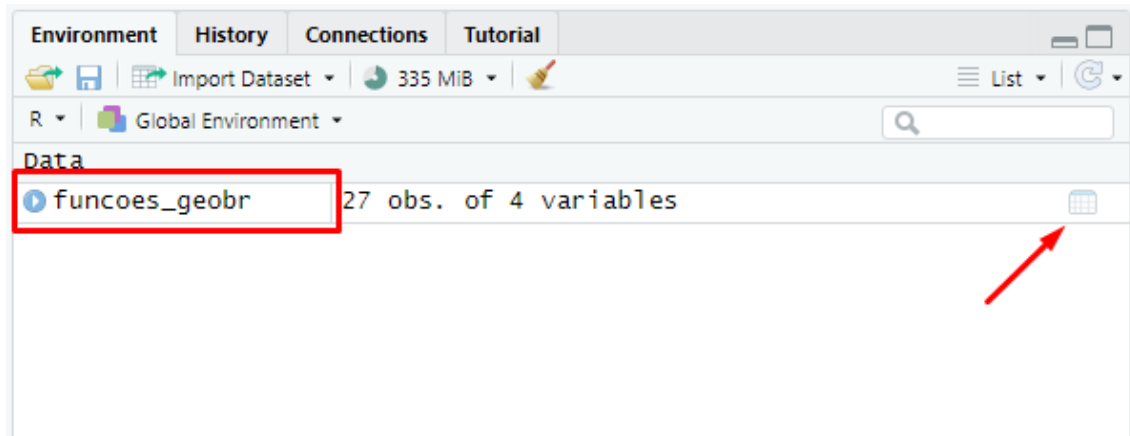


	function	geography	years	source
1	'read_country'	Country	1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, ...	IBGE
2	'read_region'	Region	2000, 2001, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019, ...	IBGE
3	'read_state'	States	1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, ...	IBGE
4	'read_meso_region'	Meso region	2000, 2001, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019, ...	IBGE
5	'read_micro_region'	Micro region	2000, 2001, 2010, 2013, 2014, 2015, 2016, 2017, 2018, 2019, ...	IBGE
6	'read_intermediate_region'	Intermediate region	2017, 2019, 2020	IBGE
7	'read_immediate_region'	Immediate region	2017, 2019, 2020	IBGE
8	'read_municipality'	Municipality	1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, ...	IBGE
9	'read_municipal_seat'	Municipality seats (sedes municipais)	1872, 1900, 1911, 1920, 1933, 1940, 1950, 1960, 1970, 1980, ...	IBGE
10	'read_weighting_area'	Census weighting area (área de ponderação)	2010	IBGE
11	'read_census_tract'	Census tract (setor censitário)	2000, 2010, 2017, 2019, 2020	IBGE
12	'read_statistical_grid'	Statistical Grid of 200 x 200 meters	2010	IBGE
13	'read_metro_area'	Metropolitan areas	1970, 2001, 2002, 2003, 2005, 2010, 2013, 2014, 2015, 2016, ...	IBGE

Showing 1 to 13 of 27 entries, 4 total columns

janela em que editamos o *script*. Para voltar ao *script*, clique na aba em que seu nome aparece.

Outra maneira de visualizar o objeto é clicar no objeto data no quadro superior direito, em cima do seu nome ou no botão em forma de tabela da aba *Environment*.



O pacote possui 27 funções, que estão disponíveis de acordo com o ano e com o território e possui dados das regiões, dos estados e municípios, além de outras divisões territoriais, como microrregiões, territórios indígenas, áreas com risco de desastre, entre outros.

Inicialmente, vamos importar os dados geográficos dos estados brasileiros no ano de 2018 para um objeto chamado “brasil” e depois vamos plotar o mapa de UF do país.

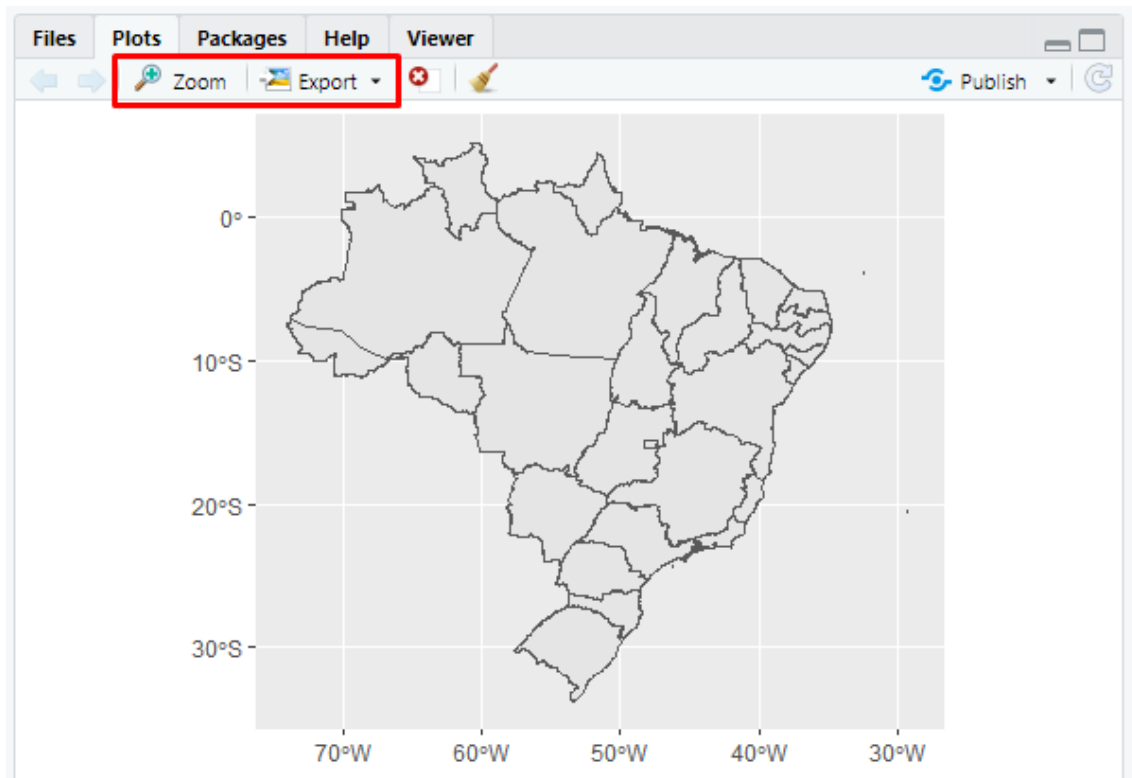
```
##IMPORTANDO DADOS DOS ESTADOS BRASILEIROS
brasil = read_state(code_state = "all", year = 2018)

##PLOTANDO OS DADOS COM O PACOTE GGLOT2
ggplot() +
  geom_sf(data= brasil)
```

Ao executar esses comandos, você poderá visualizar os resultados(*outputs*) no quadro inferior direito do RStudio, na aba “*Plots*”. Você conseguirá visualizar um mapa do Brasil, com a latitude e a longitude representados nos eixos y e x, respectivamente, do plano cartesiano. Lembre-se que o mapa, assim como os gráficos que usamos anteriormente, são representações espaciais de dados em um plano cartesiano.

Nós temos as feições de cada um dos estados brasileiros, representados no mapa. Nesse caso, podemos exportar o mapa usando o botão “*Export*”, que

permite que configuremos o tamanho da imagem e ainda podemos ampliá-lo usando o comando “Zoom”.



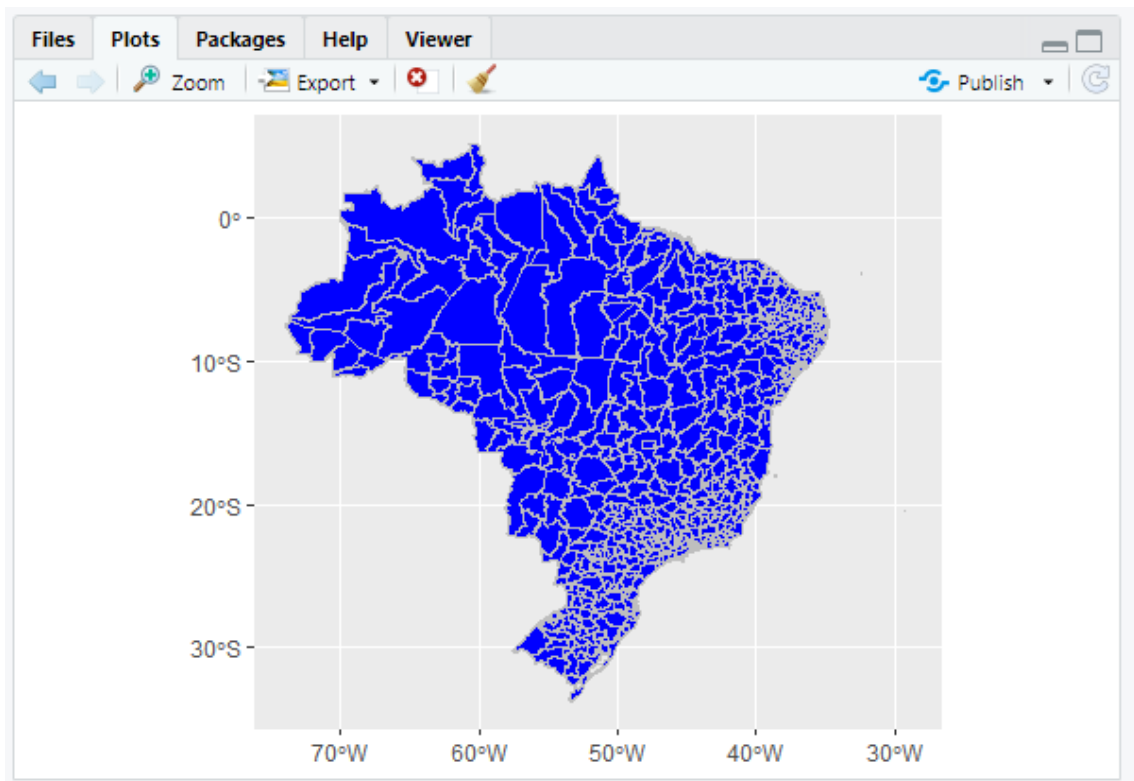
Nós vamos plotar os dados de microrregiões para o ano de 2015 e acrescentar alguns argumentos nas funções do *ggplot2* para conseguirmos modificar as cores do nosso mapa. Apenas recapitulando, argumentos são os parâmetros das funções que nos permitem modificar sua execução.

Por padrão (*default*), a função “*geom\_sf*”, que usamos para plotar o mapa, exibe um mapa na cor cinza, com bordas pretas. Vamos plotar um mapa com preenchimento azul e bordas cinza. Para isso, usaremos os argumentos “*fill*”, responsável pela cor do preenchimento, e o argumento “*color*”, responsável pela cor da borda.

```
##CARREGANDO E PLOTANDO DADOS DAS MICRORREGIÕES NO ANO DE 2015
micro_reg = read_micro_region(code_micro = "all", year = 2015)

ggplot() +
  geom_sf(data= micro_reg,
          fill = "blue",
          color = "grey",
          show.legend = T)
```

Como resultado, nós teremos um mapa das microrregiões de 2015, de bordas cinza e preenchimento azul.



Com esse pacote é possível fazer esse tipo de visualização com qualquer outra divisão administrativa, usando a função correspondente para cada uma delas.

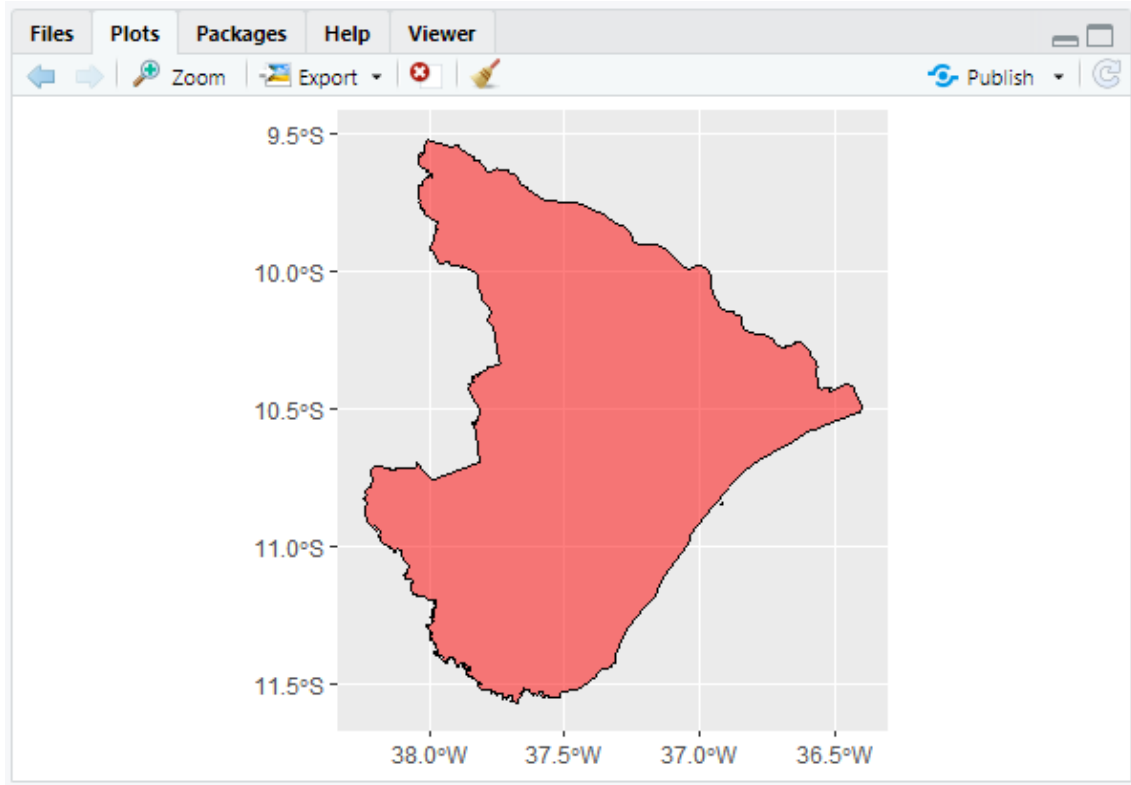
Além dos mapas nacionais, podemos importar mapas de territórios menores, como estados ou municípios, dependendo da nossa análise. Nós vamos importar os dados do estado do Sergipe, usando também o argumento “*alpha*”, que permite graduar a transparência do mapa.

```
##CARREGANDO E PLOTANDO MAPA DO ESTADO DE SERGIPE
sergipe <- read_state(code_state="SE", year=2018)

ggplot() +
  geom_sf(data= sergipe,
    fill = "red",
    color = "black",
    alpha = 0.5)
```

Nesse caso, teremos uma visualização do estado de Sergipe, sem as divisões intermunicipais, em um tom de vermelho levemente transparente.





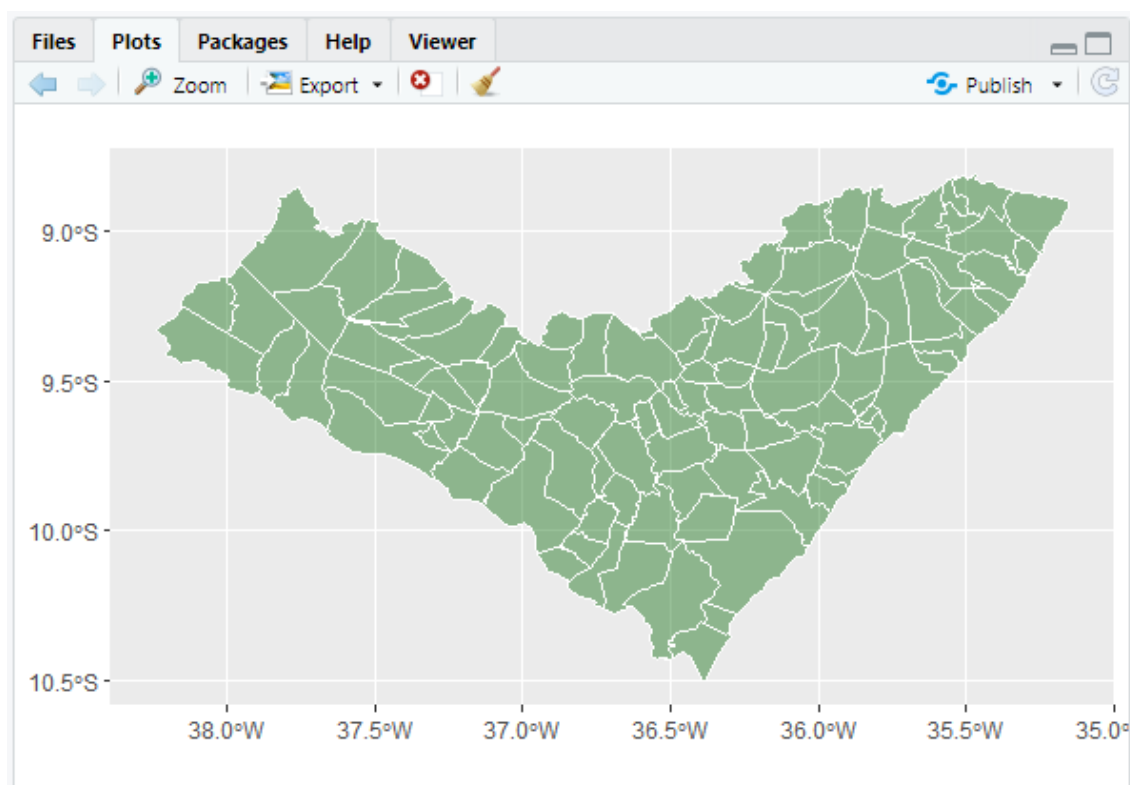
Dessa forma, você também pode conseguir os dados de um único município com a função `read_municipality()`, usando o código geográfico do IBGE do município no argumento `code_muni`.

Para carregar os dados de todos os municípios de um estado você usará essa mesma função, mas o argumento `code_muni` receberá a sigla do estado. Vamos executar o código dessa função para os municípios do estado de Alagoas, para o ano de 2007.

```
##CARREGANDO E PLOTANDO MAPA DOS MUNICÍPIOS DO ESTADO DE ALAGOAS
munic_al <- read_municipality(code_muni= "AL", year=2007)

ggplot() +
  geom_sf(data= munic_al,
          fill = "darkgreen",
          color = "white",
          alpha = 0.4)
```

O *output* desse código retornará um mapa de bordas brancas e preenchimento verde, com uma leve transparência.



Ainda é possível realiza a seleção de setores censitários, áreas urbanas e rurais entre outras seleções alterando apenas as funções e seus respectivos argumentos. O modo de criação do mapa será igual.

#### **4. Combinando dados de objetos espaciais e tabelas**

Compreender a plotagem dos mapas e suas seleções é absolutamente importante, entretanto não é suficiente para realizarmos análises de saúde. Os pacotes que importamos também não possuem todas as informações de saúde que precisamos para realizar análises de saúde.

Por isso, nós importaremos os dados de número de casos de covid no Brasil por UF em 2020 e os dados de população estimada para cada estado nesse ano para que possamos calcular as taxas de incidência e plotarmos um mapa temático por UF.

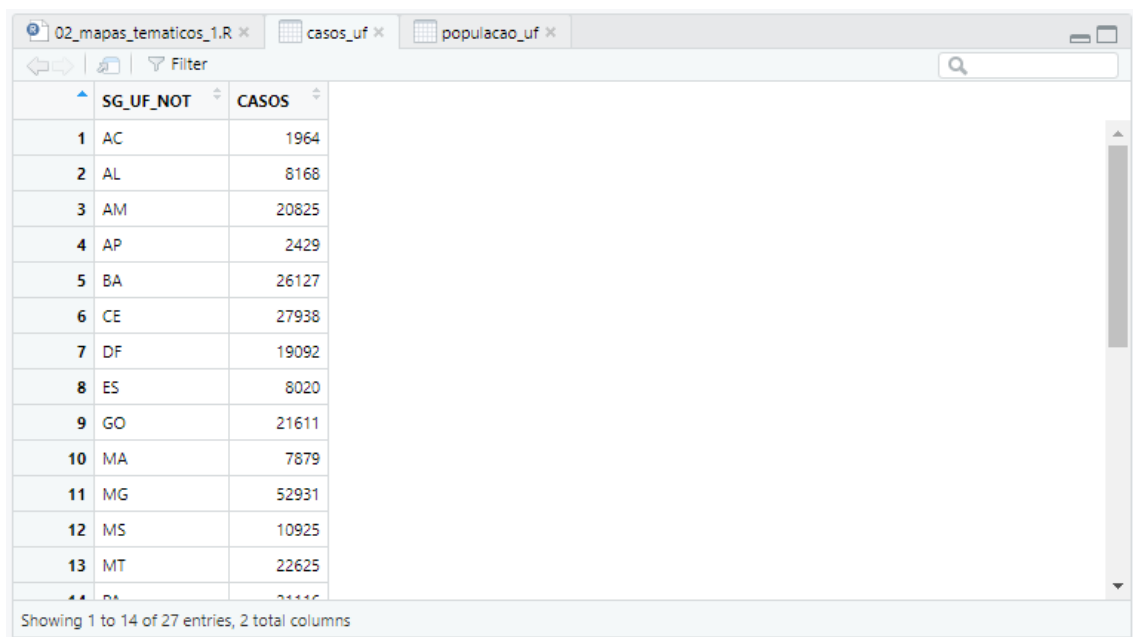
Inicialmente, vamos importar os dados do Sivep-Gripe de Síndrome Respiratória Aguda Grave por Covid-19 no ano de 2020. Nós importamos os dados referentes à base de dados disponível *online* para o Sivep-Gripe, que

possui dados de Síndrome Respiratória Aguda Grave por diversas causas. Vamos selecionar apenas os dados de Covid-19 e vamos contar a frequência desses dados por UF, para isso usaremos as variáveis de UF de notificação (SG\_UF\_NOT) e de classificação final (CLASSI\_FIN), em que selecionaremos todos os casos com classificação 5, correspondente a Covid-19. Usaremos funções do pacote *tidyverse* para realizar essas transformações e ao final obteremos uma tabela com as UF e com os casos de Covid-19 em 2020.

```
##IMPORTANDO DADOS DE SARS POR COVID-19 EM 2020
sivep2020 = import("dados/sivep_gripe_2020.csv")

sivep2020_casos = sivep2020|>      #criando um data.frame
  select(CLASSI_FIN, SG_UF_NOT)|>  #selecionando as variáveis
  filter(CLASSI_FIN == 5) |>      #selecionando dados de Covid-19
  select(SG_UF_NOT) |>           #selecionando apenas dados de UF
  group_by(SG_UF_NOT) |>         #agrupando dados por UF
  summarise(CASOS = n())         #contando número de casos por UF
```

Após a sumarização dos dados, vamos visualizar os dados do *data.frame* que criamos. Ele deverá conter 27 observações de duas variáveis, as 27 UF e o número de casos de cada uma delas.



	SG_UF_NOT	CASOS
1	AC	1964
2	AL	8168
3	AM	20825
4	AP	2429
5	BA	26127
6	CE	27938
7	DF	19092
8	ES	8020
9	GO	21611
10	MA	7879
11	MG	52931
12	MS	10925
13	MT	22625
14	PA	31116

Nós precisaremos unir os dados de população e de casos. Nós vamos uni-los com base na sigla de cada UF criando um *data.frame* chamado “*incid\_uf*”. Em geral, recomendamos utilizar o código IBGE dos dados, principalmente em caso de união de diferentes *datasets* de municípios, em que ocorrem diversas repetições de nomes.

Nesse caso, como cada sigla de UF é única e como não há código do IBGE na tabela de casos que criamos, nós utilizaremos as siglas de UF. Na tabela “*casos\_uf*”, usaremos a variável “SG\_UF\_NOT” e na tabela “*populacao\_uf*”, usaremos a variável “sigla”.

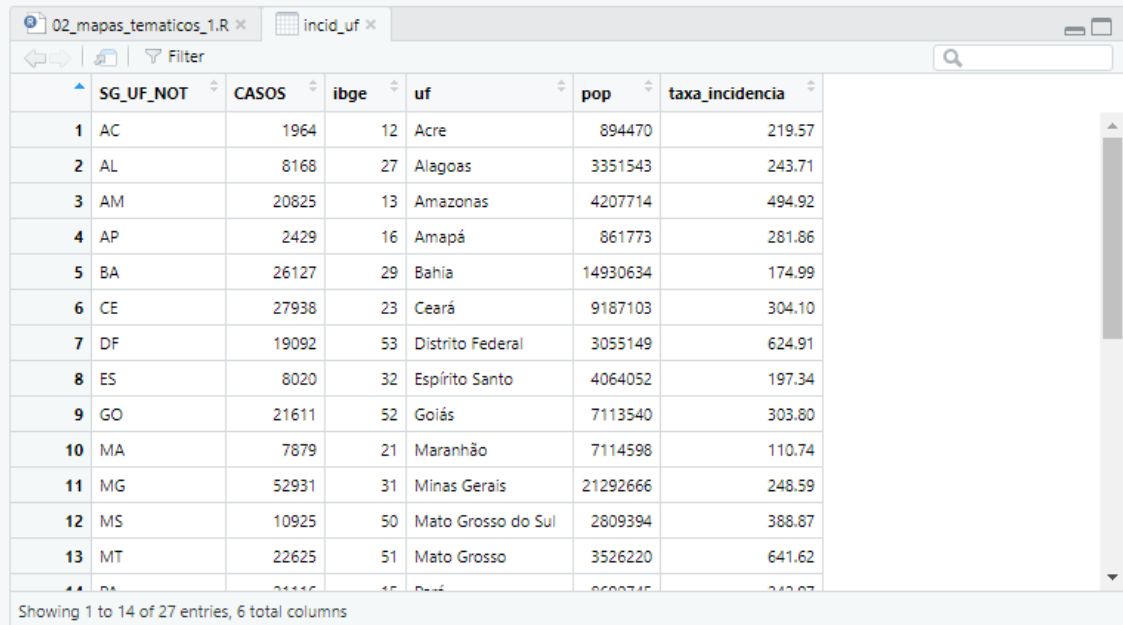
```
##UNINDO DATA.FRAMES DE CASOS E DE POPULAÇÕES
incid_uf = casos_uf |>
  left_join(populacao_uf, by = c("SG_UF_NOT" = "sigla"))
```

Nós obteremos um *data.frame* que possui todos os dados de casos e população e poderemos calcular a taxa de incidência para o ano de 2020. Após isso, salvaremos nossa tabela novamente como um arquivo .RDS. Faremos isso usando o seguinte código.

```
##CALCULANDO TAXA DE INCIDÊNCIA
incid_uf$taxa_incidencia = round(incid_uf$CASOS/incid_uf$pop*100000,2)

##SALVANDO O ARQUIVO EM FORMATO .RDS
export(sivep2020_casos, "dados/sivep2020_casos.RDS")
```

Obteremos o seguinte *data.frame*:

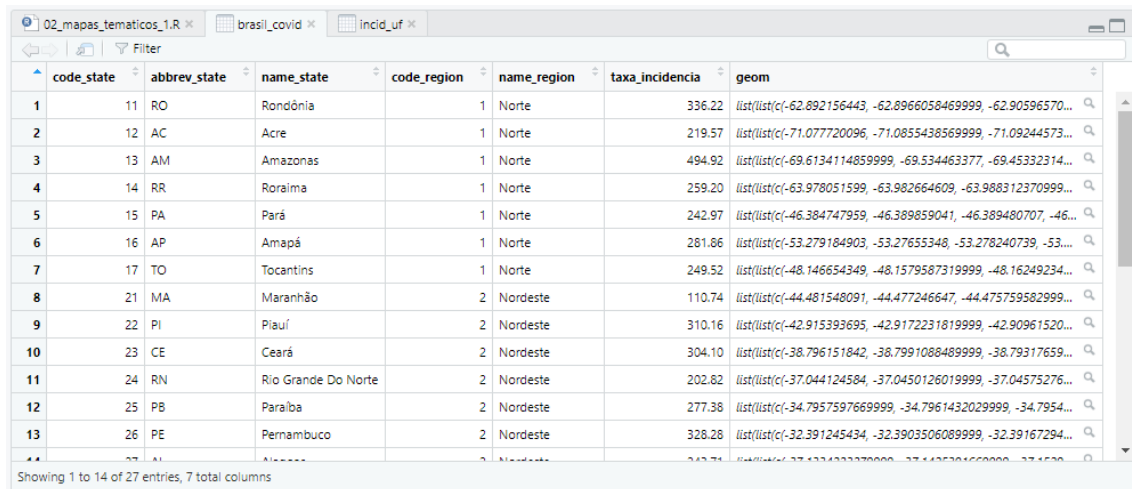


	SG_UF_NOT	CASOS	ibge	uf	pop	taxa_incidencia
1	AC	1964	12	Acre	894470	219.57
2	AL	8168	27	Alagoas	3351543	243.71
3	AM	20825	13	Amazonas	4207714	494.92
4	AP	2429	16	Amapá	861773	281.86
5	BA	26127	29	Bahia	14930634	174.99
6	CE	27938	23	Ceará	9187103	304.10
7	DF	19092	53	Distrito Federal	3055149	624.91
8	ES	8020	32	Espírito Santo	4064052	197.34
9	GO	21611	52	Goiás	7113540	303.80
10	MA	7879	21	Maranhão	7114598	110.74
11	MG	52931	31	Minas Gerais	21292666	248.59
12	MS	10925	50	Mato Grosso do Sul	2809394	388.87
13	MT	22625	51	Mato Grosso	3526220	641.62

## 5. Elaborando um mapa de taxas de incidência

Vamos utilizar os dados do *data.frame* “brasil”, que criamos anteriormente para unir os dados de taxa de incidência por Covid-19 em 2020. Para isso, selecionaremos apenas os dados de sigla da UF e de taxa de incidência por

Covid-19 para após isso, unir à tabela de atributos do objeto “brasil”, usando a função “*left\_join()*” e a variável “abbrev\_state”, e plotar o mapa de taxas de



	code_state	abbrev_state	name_state	code_region	name_region	taxa_incendencia	geom
1	11	RO	Rondônia	1	Norte	336.22	list(list(c(-62.892156443, -62.8966058469999, -62.90596570...
2	12	AC	Acre	1	Norte	219.57	list(list(c(-71.077720096, -71.0855438569999, -71.09244573...
3	13	AM	Amazonas	1	Norte	494.92	list(list(c(-69.6134114859999, -69.534463377, -69.45332314...
4	14	RR	Roraima	1	Norte	259.20	list(list(c(-63.978051599, -63.982664609, -63.988312370999...
5	15	PA	Pará	1	Norte	242.97	list(list(c(-46.384747959, -46.389859041, -46.389480707, -46...
6	16	AP	Amapá	1	Norte	261.86	list(list(c(-53.279184903, -53.27655348, -53.278240739, -53...
7	17	TO	Tocantins	1	Norte	249.52	list(list(c(-48.146654349, -48.1579587319999, -48.16249234...
8	21	MA	Maranhão	2	Nordeste	110.74	list(list(c(-44.481548091, -44.477246647, -44.475759582999...
9	22	PI	Piauí	2	Nordeste	310.16	list(list(c(-42.915393695, -42.9172231819999, -42.90961520...
10	23	CE	Ceará	2	Nordeste	304.10	list(list(c(-38.796151842, -38.7991088489999, -38.79317659...
11	24	RN	Rio Grande Do Norte	2	Nordeste	202.82	list(list(c(-37.044124584, -37.0450126019999, -37.04575276...
12	25	PB	Paraíba	2	Nordeste	277.38	list(list(c(-34.7957597669999, -34.7961432029999, -34.7954...
13	26	PE	Pernambuco	2	Nordeste	326.28	list(list(c(-32.391245434, -32.3905506089999, -32.39167294...

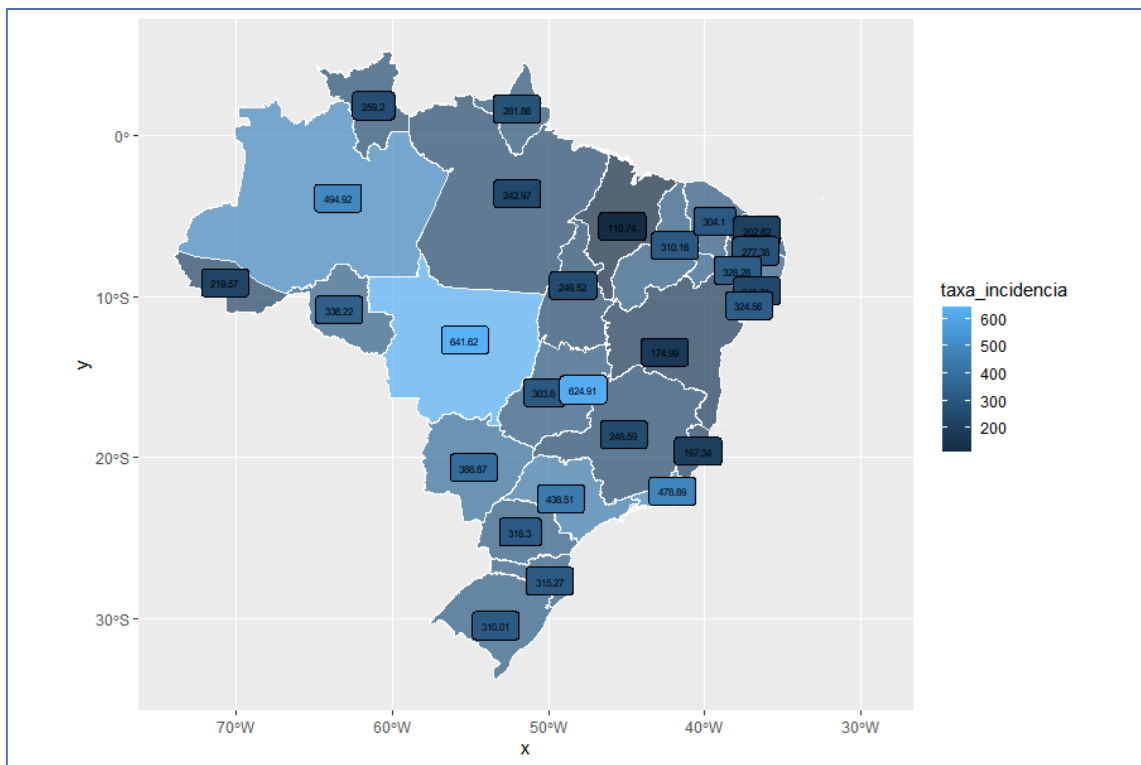
incidência do país. Nós obteremos o seguinte *data.frame*:

Perceba que esse *data.frame* possui as informações geográficas e também informações de taxa de incidência de Covid-19, que nos permite criar um mapa temático com esses dados.

Nós criaremos um mapa temático simples, com a taxa de incidência e com rótulos com as taxas de incidência de cada UF.

```
##CRIANDO UM MAPA TEMÁTICO
ggplot(data = brasil_covid, aes(fill = taxa_incendencia)) +
  geom_sf(color = "white", #cor da borda
          size = .15,      #tamanho da borda
          show.legend = F, #não mostrar legendas
          alpha = 0.7) +   #transparência
  geom_sf_label(aes(label=taxa_incendencia), #escolhendo variável
               #para os rótulos
               label.padding = unit(2, "mm"), size = 2) #tamanho dos
#rótulos
```

Obteremos assim nosso primeiro mapa temático de incidência de Covid-19 no Brasil em 2020, por UF.



## 6. Referências Bibliográficas

1. The Comprehensive R Archive Network [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://cran.r-project.org/>
2. devtools package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/devtools/versions/1.13.6>
3. geobr package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/geobr/versions/1.6.5>
4. Rinker T, Dason Kurkiewicz, Pastoor D. Pacman: Pacman Version 0.2.0 [Internet]. Zenodo; 2015 [citado 16 de julho de 2022]. Disponível em: <https://zenodo.org/record/15406>
5. ggsm package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/ggsm/versions/0.5.0>
6. ggspatial package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/ggspatial/versions/1.1.6>
7. gpclib package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/gpclib/versions/1.5-6>
8. jsonlite package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/jsonlite/versions/1.8.0>
9. leaflet package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/leaflet/versions/2.1.1>
10. lubridate package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/lubridate/versions/1.8.0>
11. maptools package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/maptools/versions/1.1-4>
12. RColorBrewer package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/RColorBrewer/versions/1.1-3>
13. RCurl package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/RCurl/versions/1.98-1.7>

14. rgdal package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/rgdal/versions/1.5-32>
15. rgeos package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/rgeos/versions/0.5-9>
16. rio package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/rio/versions/0.5.29>
17. rjson package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/rjson/versions/0.2.21>
18. tidyverse package - RDocumentation [Internet]. [citado 16 de julho de 2022]. Disponível em: <https://www.rdocumentation.org/packages/tidyverse/versions/1.3.1>



## 7. Exercícios

Marque verdadeiro ou falso para a afirmativa	
<i>Packages</i> são conjuntos de funções para finalidades específicas que podem ser instalados no R	V
O R não é uma linguagem de código aberto.	F
A função " <i>p_load()</i> " do pacote <i>pacman</i> verifica se o pacote está instalado e carrega todos os pacotes selecionados.	V
O pacote <i>geobr</i> não dá suporte para criação de mapas de municípios, apenas de mapas das UF	F
O pacote <i>geobr</i> não dá suporte para criação de mapas de setores censitários, apenas de mapas das UF	F
É possível unir um <i>data.frame</i> a um objeto espacial para criar categorias de análise em saúde	V
O pacote <i>geobr</i> possui funções de diversos tipos de território brasileiros, incluindo estado, municípios e setores censitários.	V
Não é possível criar mapas com o pacote <i>ggplot2</i> , que é voltado apenas para gráficos carterianos.	F
É possível criar mapas de UF de forma isolada com o pacote <i>geobr</i>	V
Não é possível mudar a cor de um objeto do tipo mapa, visto que ele possui cores padrão como preto para as bordas, marrom para os polígonos e azul para as linhas.	F