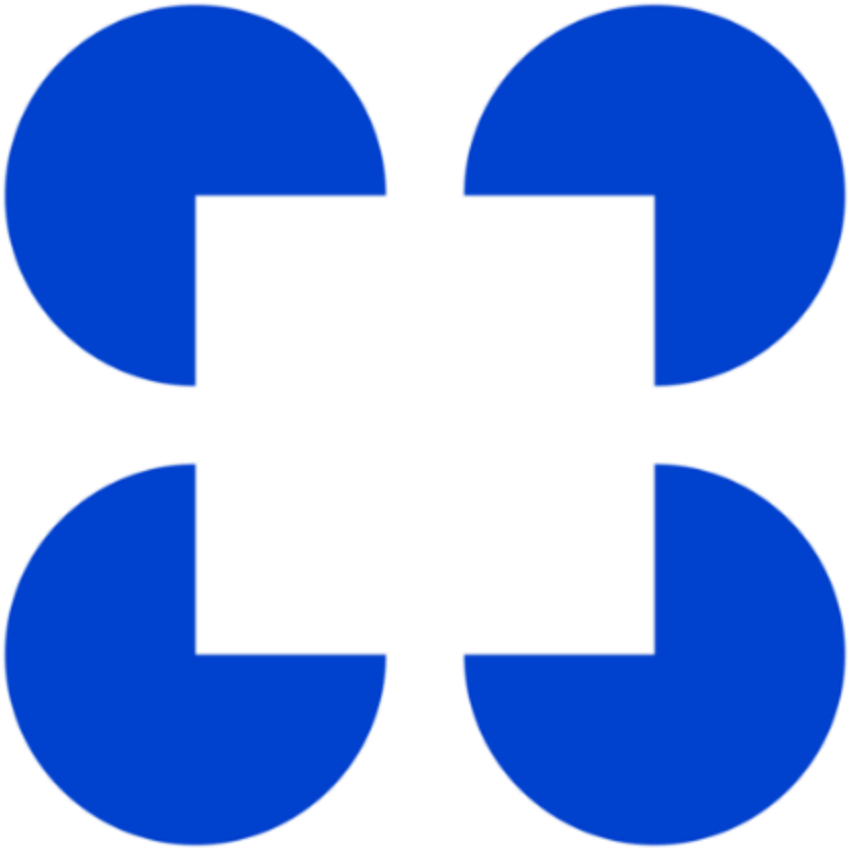
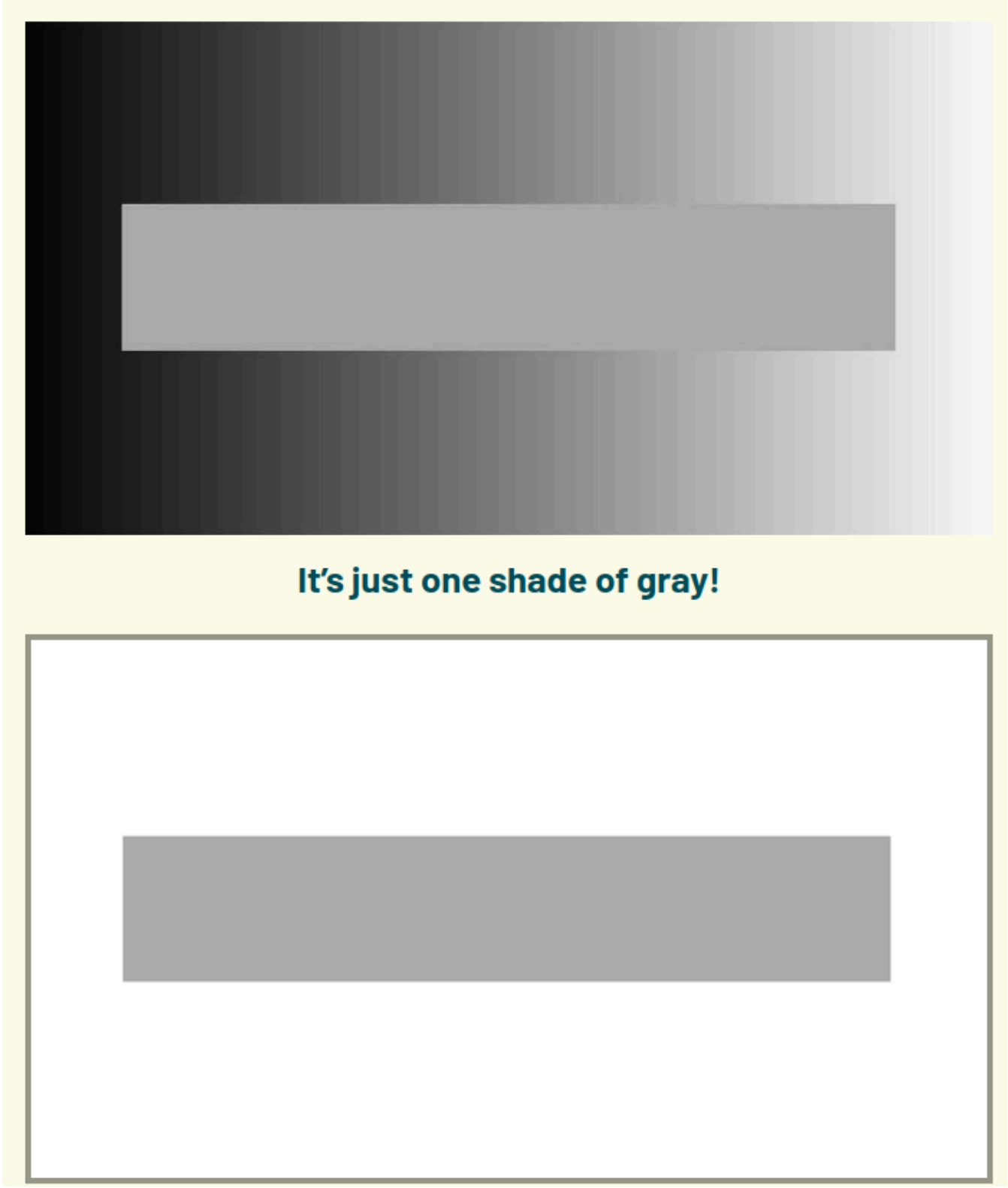
Participação 3

Exercício 1



A imagem acima é um exemplo de fechamento pois a colocação dos 4 3/4 de circulos cria uma ilusão de 4 circulos sobrepostos por um quadrado em branco, quando na verdade são 4 3/4 de circulos.



Já esta imagem nos apresenta um exemplo de contraste percebido, onde a percpção da faixa central é afetada pelo contexto, no caso o fundo da imagem, fazendo com que pareça que existe mais de uma tonalidade na faixa.

Exercício 2

A primeira solução apresentada é devido a como gráficos baseados em área são ruins para a percepção humana, sendo pior do que apesta uma lista com as porcentagens a serem mostradas.

Logo, como humanos são muito melhores em perceber medidas lineares, recomenda-se usar comprimento e posição como indicativos visuais, como barplots.

- barplots sem iniciar em 0 pode ser desinformativo.
- Porém gráficos baseados em posição como stratfield são mais recomendados iniar em um valor perto do menos para melhor visualização
- Usar comprimento em formas geométricas pode enganar também como o diametro de um círculo ao invés de sua área.

ggplot organiza alfabéticamente por padrão, mas o ideal é por algo como quantidade. Para isso pode ser usado o reorder

In [ ]:
```
import pandas as pd
import numpy as np
from plotnine import *
```

In [ ]:
```
murders = pd.read_csv("murders.csv")
murders.head()
```

Out[ ]:

	state	abb	region	population	total
0	Alabama	AL	South	4779736	135
1	Alaska	AK	West	710231	19
2	Arizona	AZ	West	6392017	232
3	Arkansas	AR	South	2915918	93
4	California	CA	West	37253956	1257

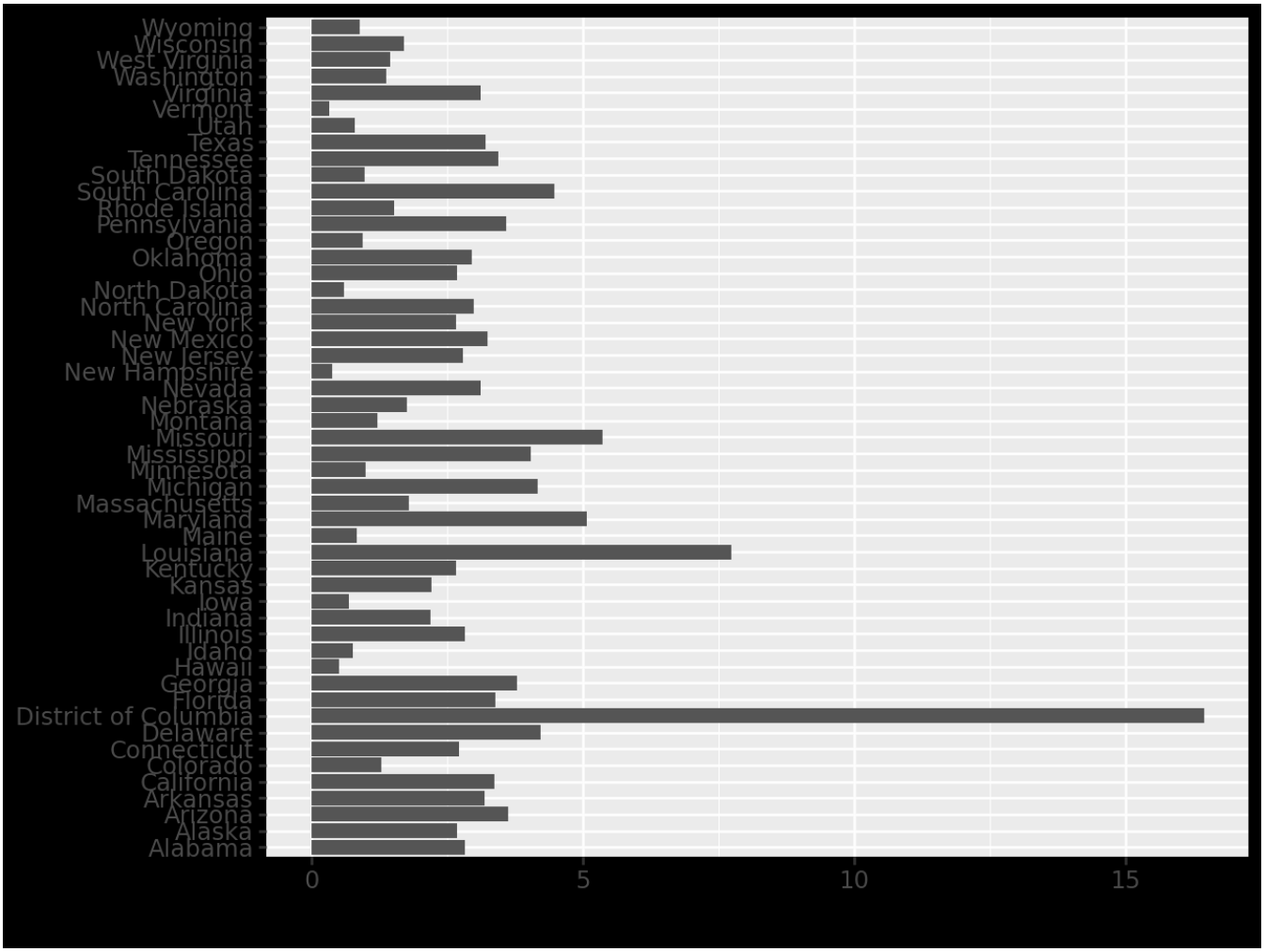
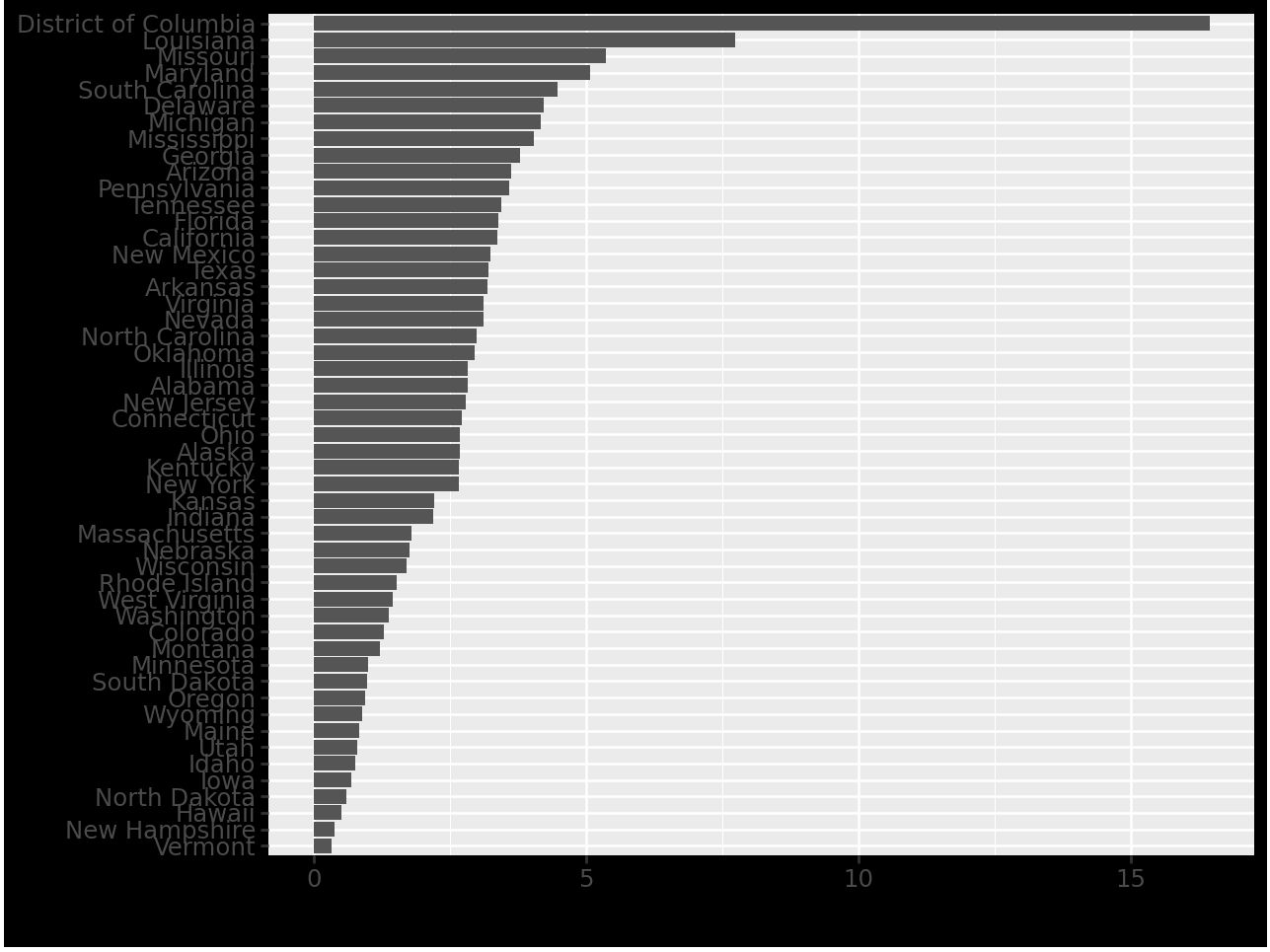


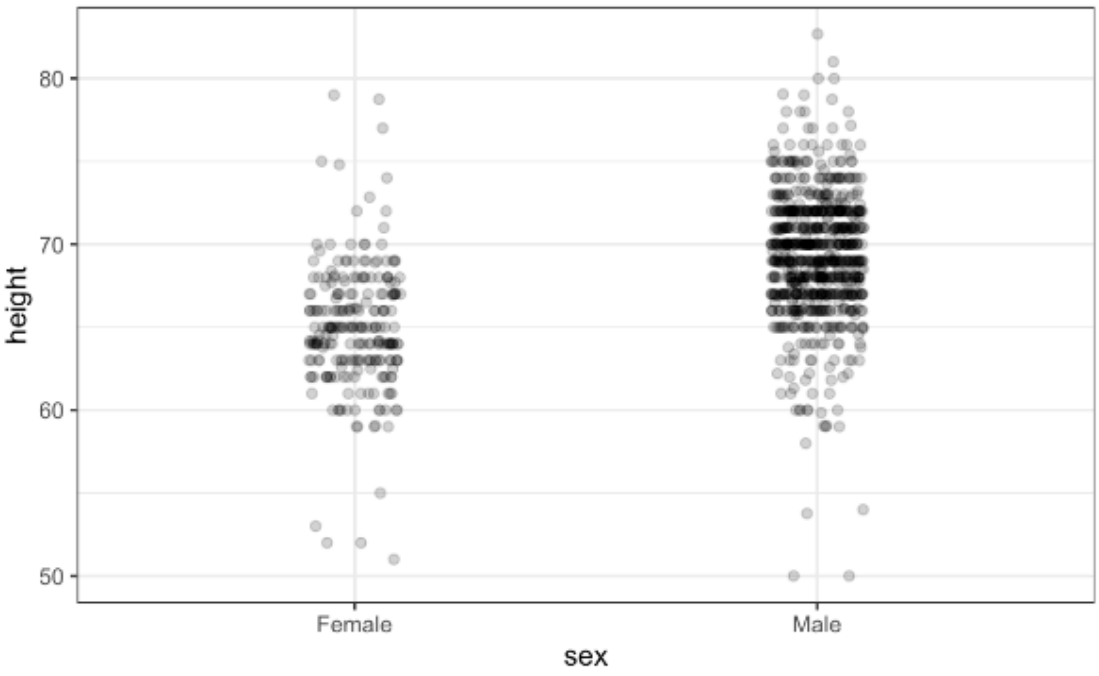
Imagem de plot gerado sem reorder

In [ ]:

```
# murders |> mutate(murder_rate = total / population * 100000) |>
#   mutate(state = reorder(state, murder_rate)) |>
#   ggplot(aes(state, murder_rate)) +
#   geom_bar(stat="identity") +
#   coord_flip() +
#   theme(axis.text.y = element_text(size = 6)) +
#   xlab("")

murders["murder_rate"] = murders["total"]/murders["population"]*100000

plot = ggplot(murders, aes("reorder(state, murder_rate)", "murder_rate"))
plot += geom_bar(stat="identity")
plot += coord_flip()
plot += theme()
plot += xlab("")

plot
```



Como com o o geom_point() para muitos dados fica dificil de enxergar a difereça pode se usar o geom_jitter que deixa os pontos distribuídos e transparentes, possibibilitando melhor observação de diferença entre as categorias



Mostrar histogramas para diferentes grupos é uma ótima forma de compará-los. Melhor ainda se alinha-los verticalmente para observar mudanças horizontais.

Considerar Transformações:

- Um histograma está sensível a outliers, porém com um plot é possível indentificar melhor a relação entre os elementos de cada categoria. Por exemplo países Ásiaticos tem uma média populacional maior devido a China e India.
- Colocar dados a serem compados próximos também é uma boa ideia
- Usar cor para comparar, melhor ainda

Use cores pensando em daltonismo

Visualizações pseudo 3D são ruins

Diminua o número de digitos significantes para diminuir a carga cognitiva do observador da tabela/gráfico. Reduzir o número de casas depois da vírgula é uma boa solução. Outra também é modificar os dados como colunas virarem valores de outras ou valores de colunas virarem suas próprias colunas dependendo do objetivo de visualização.

Pense em quem deverá obter informações do gráfico, por exemplo, para um publico menos familiar com analise de dados, tecnicas mais simples podem ser mais recomendadas.

Exercício 3

In [ ]:
```
import pandas as pd
import numpy as np
```

In [ ]:
```
df = pd.read_csv("us_contagious_diseases.csv")
df.info
```

Out[ ]:
```
<bound method DataFrame.info of        rownames      disease    state  year  weeks_repor
ting  count  \
0             1  Hepatitis A  Alabama  1966              50    321
1             2  Hepatitis A  Alabama  1967              49    291
2             3  Hepatitis A  Alabama  1968              52    314
3             4  Hepatitis A  Alabama  1969              49    380
4             5  Hepatitis A  Alabama  1970              51    413
...         ...          ...      ...   ...             ...    ...
16060     16061     Smallpox  Wyoming  1948              24      1
16061     16062     Smallpox  Wyoming  1949               0      0
16062     16063     Smallpox  Wyoming  1950               1      2
16063     16064     Smallpox  Wyoming  1951               1      1
16064     16065     Smallpox  Wyoming  1952               1      1

       population
0       3345787.0
1       3364130.0
2       3386068.0
3       3412450.0
4       3444165.0
...           ...
16060    280803.0
16061    285544.0
16062    290529.0
16063    295744.0
16064    301083.0

[16065 rows x 7 columns]>
```

In [ ]:
```
df = df[df["disease"] == "Measles"]
df.info
```

Out[ ]:
```
<bound method DataFrame.info of       rownames  disease    state  year  weeks_reporting
count  population
2346      2347  Measles  Alabama  1928               52    8843   2589923.0
2347      2348  Measles  Alabama  1929               49    2959   2619131.0
2348      2349  Measles  Alabama  1930               52    4156   2646248.0
2349      2350  Measles  Alabama  1931               49    8934   2670818.0
2350      2351  Measles  Alabama  1932               41     270   2693027.0
...        ...      ...      ...   ...              ...     ...         ...
6166      6167  Measles  Wyoming  1998                0       0    479897.0
6167      6168  Measles  Wyoming  1999                0       0    486758.0
6168      6169  Measles  Wyoming  2000                0       0    493782.0
6169      6170  Measles  Wyoming  2001                0       0    500794.0
6170      6171  Measles  Wyoming  2002                0       0    507765.0

[3825 rows x 7 columns]>
```

In [ ]:
```
df = df[df["state"] != "Hawaii"]
df = df[df["state"] != "Alaska"]
df.info
```

Out[ ]:
```
<bound method DataFrame.info of       rownames  disease    state  year  weeks_reporting
count  population
2346      2347  Measles  Alabama  1928               52    8843   2589923.0
2347      2348  Measles  Alabama  1929               49    2959   2619131.0
2348      2349  Measles  Alabama  1930               52    4156   2646248.0
2349      2350  Measles  Alabama  1931               49    8934   2670818.0
2350      2351  Measles  Alabama  1932               41     270   2693027.0
...        ...      ...      ...   ...              ...     ...         ...
6166      6167  Measles  Wyoming  1998                0       0    479897.0
6167      6168  Measles  Wyoming  1999                0       0    486758.0
6168      6169  Measles  Wyoming  2000                0       0    493782.0
6169      6170  Measles  Wyoming  2001                0       0    500794.0
6170      6171  Measles  Wyoming  2002                0       0    507765.0

[3675 rows x 7 columns]>
```

In [ ]:
```
df.head()
```

Out[ ]:

	rownames	disease	state	year	weeks_reporting	count	population
2346	2347	Measles	Alabama	1928	52	8843	2589923.0
2347	2348	Measles	Alabama	1929	49	2959	2619131.0
2348	2349	Measles	Alabama	1930	52	4156	2646248.0
2349	2350	Measles	Alabama	1931	49	8934	2670818.0
2350	2351	Measles	Alabama	1932	41	270	2693027.0

In [ ]:
```
df["rate"] = df["count"]/(df["population"]*10000*52)/df["weeks_reporting"]
df.head()
```

Out[ ]:

	rownames	disease	state	year	weeks_reporting	count	population	rate
2346	2347	Measles	Alabama	1928	52	8843	2589923.0	1.262717e-10
2347	2348	Measles	Alabama	1929	49	2959	2619131.0	4.433925e-11
2348	2349	Measles	Alabama	1930	52	4156	2646248.0	5.808156e-11
2349	2350	Measles	Alabama	1931	49	8934	2670818.0	1.312811e-10
2350	2351	Measles	Alabama	1932	41	270	2693027.0	4.702576e-12

In [ ]:
```
rate_avg = df[["state","rate"]].groupby('state').mean()
rate_avg =  pd.DataFrame(rate_avg)
```

```
rate_avg = rate_avg.rename(columns={"rate":"rate_avg"})
```

In [ ]:
```
df = pd.merge(df,rate_avg, on="state", how="left")
df.head()
```

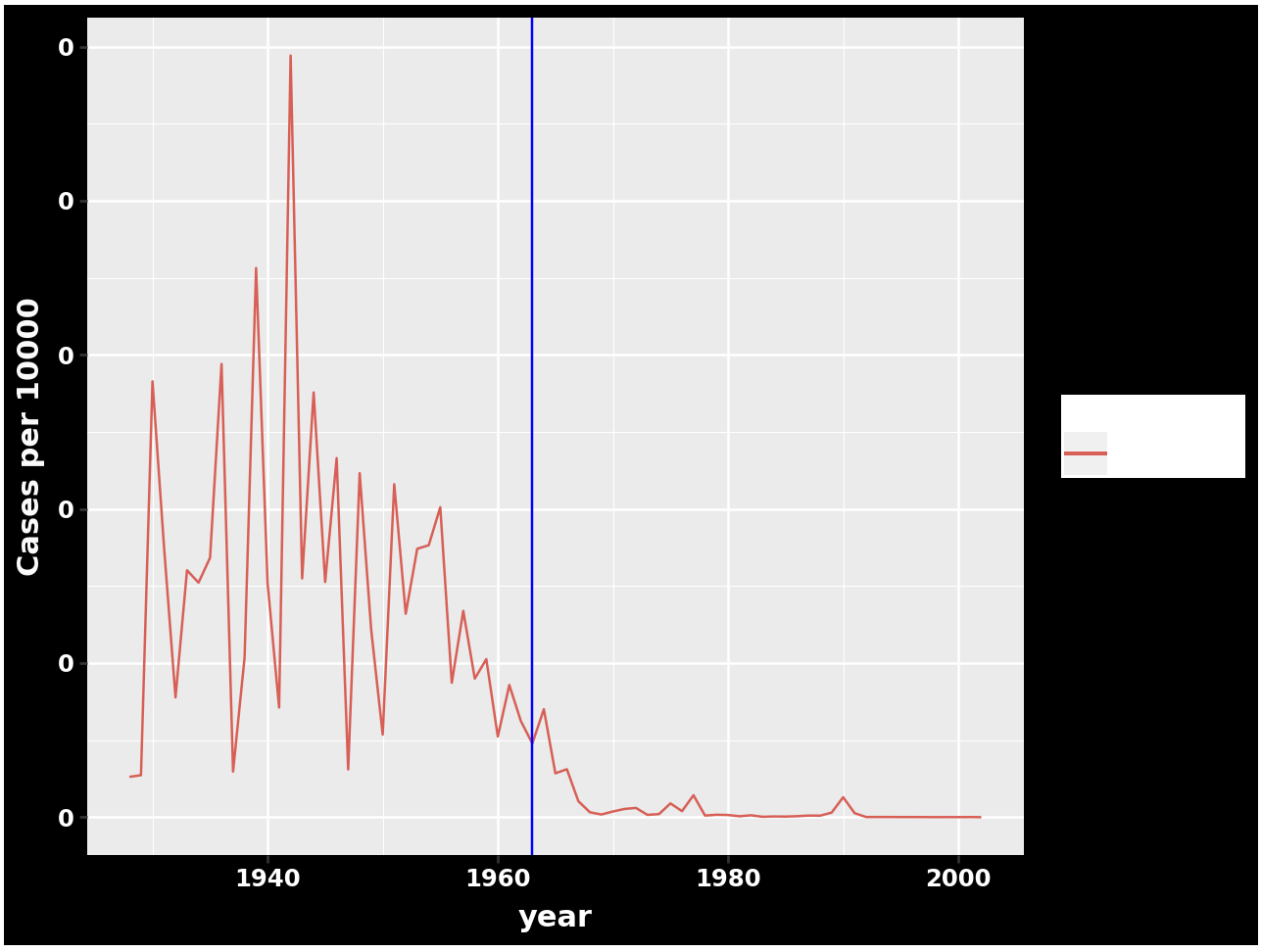
Out[ ]:

	rownames	disease	state	year	weeks_reporting	count	population	rate	rate_avg
0	2347	Measles	Alabama	1928	52	8843	2589923.0	1.262717e-10	4.132508e-11
1	2348	Measles	Alabama	1929	49	2959	2619131.0	4.433925e-11	4.132508e-11
2	2349	Measles	Alabama	1930	52	4156	2646248.0	5.808156e-11	4.132508e-11
3	2350	Measles	Alabama	1931	49	8934	2670818.0	1.312811e-10	4.132508e-11
4	2351	Measles	Alabama	1932	41	270	2693027.0	4.702576e-12	4.132508e-11

In [ ]:
```
df = df.sort_values(by=["rate_avg"])
```

In [ ]:
```
from plotnine import *
```

In [ ]:
```
california = df[df["state"] == "California"]
ggplot(california) \
  + aes(x="year", y="rate", color="state") \
  + geom_line() \
  + geom_vline(xintercept=1963, color = "blue") \
  + labs(y='Cases per 10000', x='year') \
  + theme(text= element_text(colour = "white", face = "bold")
          , strip_text = element_text(colour = "black", face = "bold")
          )
```



In [ ]:
```
ggplot(df) \
  + aes(x="year", y="state", fill="rate") \
  + geom_tile(color = "grey") \
```

```
  + scale_x_continuous(expand=(0,0)) \
  + scale_fill_gradient(high="#8B0000",low="#FF7F7F", trans = "sqrt") \
  + geom_vline(xintercept=1963, color = "blue") \
  + theme_minimal() \
  + labs(y='Cases per 10000', x='year') \
  + theme(text= element_text(colour = "white", face = "bold")
          , strip_text = element_text(colour = "black", face = "bold")
          )
```