In [19]:
```
import pandas as pd
import numpy as np

import geopandas as gpd
import json

import statsmodels

import plotly
import plotly.express as px
import plotly.io as pio
# pio.renderers.default='jupyter'
# pio.renderers.default='jupyterlab'

# allow full interactivity offlice
plotly.offline.init_notebook_mode()
```

Load Data

Giffords Gun Law Scorecard

Load the Giffords Gun Law Scorecard for each State. Note, original scores have been coerced into a 5-point Likert scale and a numeric grade has also been assigned to each letter grades.

In [20]:
```
scorecard_df = pd.read_csv('giffords_gun_law_scorecard.csv')
scorecard_df.head(n=10)
```

Out[20]:

	state_abbrev	giffords_grade	grade
0	NY	A	4.0
1	NJ	A	4.0
2	MD	A	4.0
3	MA	A	4.0
4	IL	A	4.0
5	HI	A	4.0
6	CT	A	4.0
7	CA	A	4.0
8	WA	B	3.0
9	VA	B	3.0

Firearm Related Deaths

Load the firearm related mortality data for each state from the CDC. Data was acquired using the Socrate API and wrangled for this presentation. See DataWrangling.ipynb for details.

In [21]:
```
firearm_related_deaths_df = pd.read_csv('tidy-489q-934x-firearm-related-injury.csv')
firearm_related_deaths_df.tail(n=10)
```

Out[21]:

	year_and_quarter	state	state_abbrev	mortality_per_100k
540	2022 Q3	South Dakota	SD	16.7
541	2022 Q3	Tennessee	TN	21.6
542	2022 Q3	Texas	TX	16.1
543	2022 Q3	Utah	UT	12.4
544	2022 Q3	Vermont	VT	13.4
545	2022 Q3	Virginia	VA	15.3
546	2022 Q3	Washington	WA	13.0
547	2022 Q3	West Virginia	WV	17.9
548	2022 Q3	Wisconsin	WI	14.5
549	2022 Q3	Wyoming	WY	23.6

Make state, state abbreviation, and Giffords grade columns categorical.

In [22]:
```
firearm_related_deaths_df = firearm_related_deaths_df.astype({'state': 'category', 'state_abbrev': 'category'})
print(firearm_related_deaths_df.dtypes)

scorecard_df = scorecard_df.astype({'state_abbrev': 'category', 'giffords_grade' : 'category'})
print(scorecard_df.dtypes)
```

```
year_and_quarter        object
state                 category
state_abbrev          category
mortality_per_100k     float64
dtype: object
state_abbrev      category
giffords_grade    category
grade              float64
dtype: object
```

Retain only the latest data for firearm related deaths.

In [23]:
```
deaths_q3_2022 = firearm_related_deaths_df[firearm_related_deaths_df['year_and_quarter'] == '2022 Q3']
deaths_q3_2022.head()
```

Out[23]:

	year_and_quarter	state	state_abbrev	mortality_per_100k
500	2022 Q3	Alabama	AL	25.6
501	2022 Q3	Alaska	AK	23.3
502	2022 Q3	Arizona	AZ	20.9
503	2022 Q3	Arkansas	AR	22.4
504	2022 Q3	California	CA	9.1

Merge firearm related deaths and Giffords scrore datasets.

In [24]:
```
df = deaths_q3_2022.merge(scorecard_df, on='state_abbrev')
df = df.sort_values(by=['giffords_grade'], ascending=True)
df = df.reset_index(drop=True)
df.head()
```

Out[24]:

	year_and_quarter	state	state_abbrev	mortality_per_100k	giffords_grade	grade
0	2022 Q3	Illinois	IL	14.7	A	4.0
1	2022 Q3	Maryland	MD	13.5	A	4.0
2	2022 Q3	New Jersey	NJ	5.3	A	4.0
3	2022 Q3	Hawaii	HI	4.3	A	4.0
4	2022 Q3	Massachusetts	MA	3.9	A	4.0

Choropleth map of Firearm Mortality

Load GeoJSON data that was previously downloaded from here.

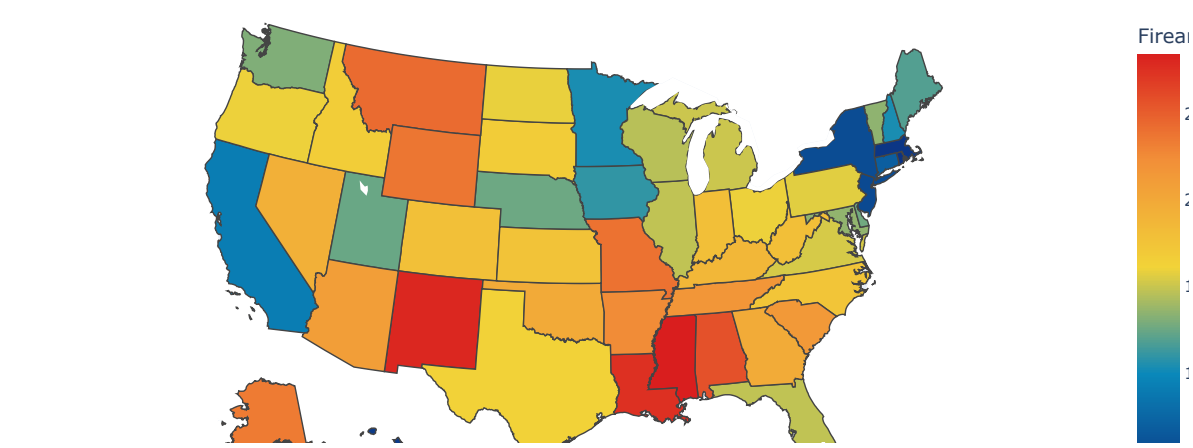
In [25]:
```
us_states = json.load(open('states.geojson', 'r'))
```

Add a new property 'id' to features - 'id' is the default name of column that is used to map values from the dataset ('state') to the corresponding State in GerJSON data.

In [26]:
```
for feat in us_states['features']:
    feat['id'] = feat['properties']['NAME']
```

Display Firearm Mortality by State on map.

In [27]:
```
fig = px.choropleth(df, locations='state', scope='usa', geojson=us_states,
                    color='mortality_per_100k',
                    hover_name='state',
                    hover_data=['giffords_grade'],
                    color_continuous_scale='Portland',
                    color_continuous_midpoint=np.average(df['mortality_per_100k']),
                    labels={
                     "mortality_per_100k": "Firearm Mortality per 100,000 persons",
                     "giffords_grade" : "Giffords Gun Law Scorecard",
                     "state" : "State"
                    },
                    title='Firearm Mortality and Gun Law Scorecard by State (Hover over map for details)')


fig.show()
```

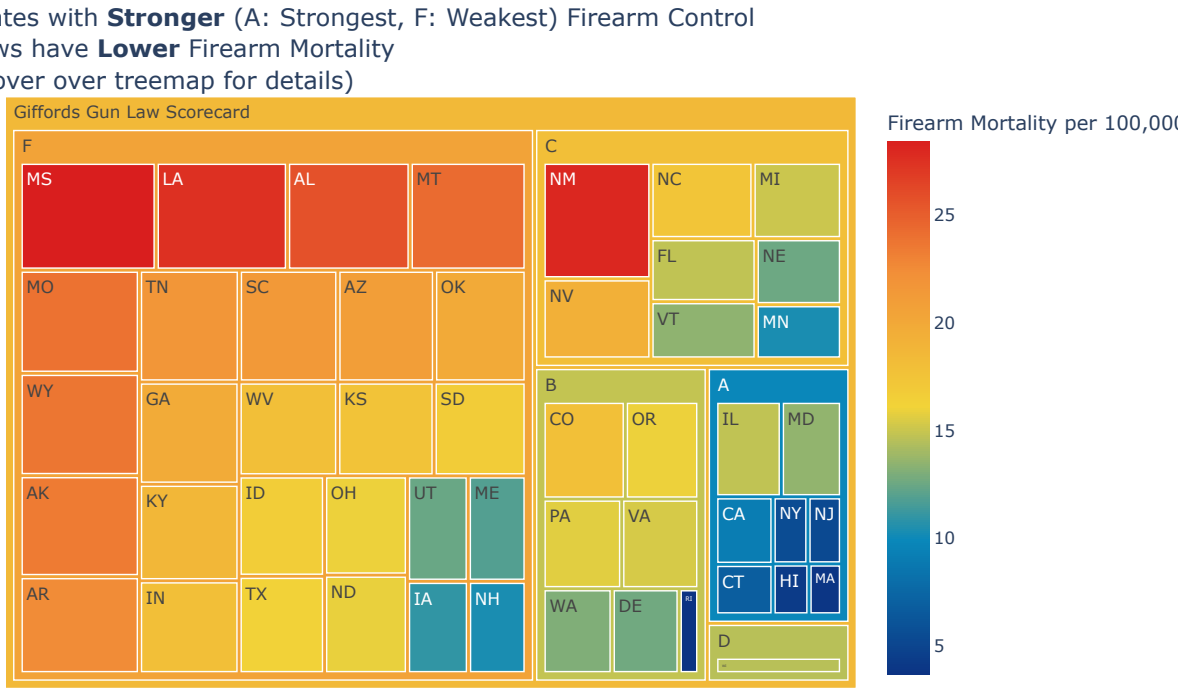


Heatmap (Treemap) of Firearm Mortality grouped by Giffords Gun Law Scorecard

The Heatmap makes reveals the following trends:

- States with the strictest gun laws have the lowest firearm related mortality rates.
- Outliers: e.g. New Mexico, which has extremely high firearm related mortality despite a non-failing gun law score. Other outliers include Utah, Maine, Iowa, and New Hampshire. Explaining outliers is not in scope for this presentation.

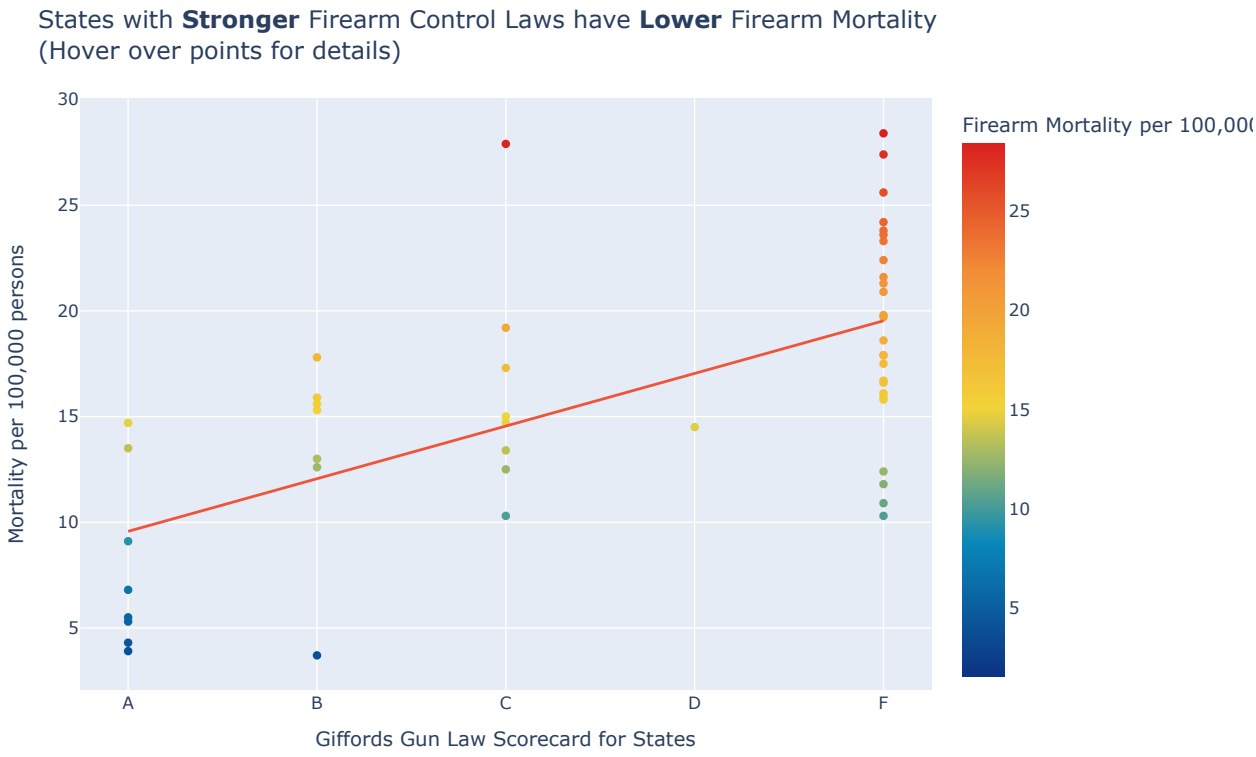
In [28]:
```
import numpy as np
fig = px.treemap(df, path=[px.Constant('Giffords Gun Law Scorecard'), 'giffords_grade', 'state_abbrev'], values='m
                 color='mortality_per_100k',
                #  labels={"mortality_per_100k": "Firearm Mortality per 100,000 persons"},
                 labels={
                     "mortality_per_100k": "Firearm Mortality per 100,000 persons",
                     "state" : "State",
                     "grade" : 'Giffords Gun Law Scorecard',
                     "giffords_grade" : 'Giffords Gun Law Scorecard'
                 },

                 height=600,
                 width=1000,
                 title='''
                 States with <b>Stronger</b> (A: Strongest, F: Weakest) Firearm Control
                 <br>Laws have <b>Lower</b> Firearm Mortality
                 <br>(Hover over treemap for details)
                 ''',
                #hover_name='label',
                #  hover_data=['mortality_per_100k'],
                 color_continuous_scale='Portland',
                 color_continuous_midpoint=np.average(df['mortality_per_100k'])
                )

fig.show()
```



Scatterplot with trend line using Ordinary Least Squares

In [29]:
```
fig = px.scatter(df, x="grade", y="mortality_per_100k",
                 color="mortality_per_100k",
                 labels={
                     "mortality_per_100k": "Firearm Mortality per 100,000 persons",
                     "state" : "State",
                     "grade" : 'Giffords Gun Law Scorecard',
                     "giffords_grade" : 'Giffords Gun Law Scorecard'
                 },
                 hover_name='state',
                 hover_data=['giffords_grade', 'mortality_per_100k'],

                 color_continuous_scale=px.colors.diverging.Portland,
                 color_continuous_midpoint=15,
```

```
                height=600,
                width=1000,

                trendline='ols',
                title='''
                States with <b>Stronger</b> Firearm Control Laws have <b>Lower</b> Firearm Mortality
                <br>(Hover over points for details)</br>
                '''
            )

fig.update_xaxes(type='category')
fig.update_xaxes(tickmode='array', tickvals=df['grade'], ticktext=df['giffords_grade'])
fig.update_layout(
          xaxis_title='Giffords Gun Law Scorecard for States',
          yaxis_title="Mortality per 100,000 persons"
)

#fig.update_traces(visible=False, selector=dict(mode="markers"))
fig.show()
```

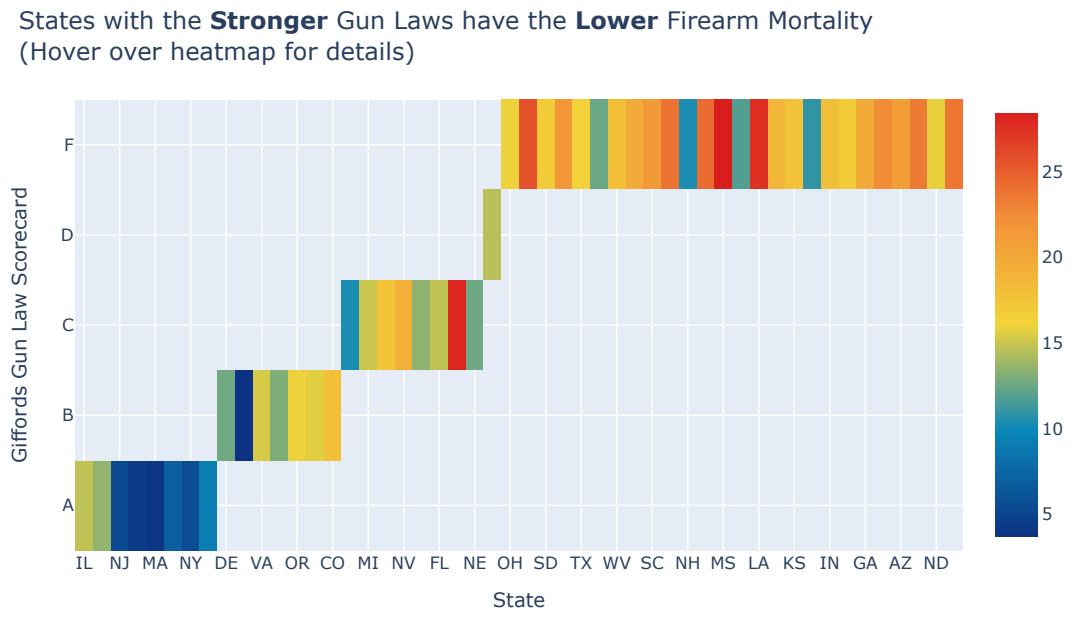


Heatmap showing Gun Mortality by State and Gun Law Grade

In [30]:

```
import plotly.graph_objects as go
import datetime
import numpy as np
np.random.seed(1)

states = list(df['state_abbrev'])
grades = list(df['giffords_grade'])
mortality = list(df['mortality_per_100k'])

base = datetime.datetime.today()
dates = base - np.arange(180) * datetime.timedelta(days=1)

fig = go.Figure(data=go.Heatmap(
        z=mortality,
        x=states,
        y=grades,
        # hoverinfo=['x', 'z'],
        hovertemplate='State: %{x}<br>Grade: %{y}<br>Firearm Mortality: %{z}<extra></extra>',
```

```
        colorscale='Portland'))

fig.update_layout(
    title='States with the <b>Stronger</b> Gun Laws have the <b>Lower</b> Firearm Mortality<br>(Hover over heatmap
    xaxis_nticks=36)

fig.update_layout(
    xaxis_title="State",
    yaxis_title="Giffords Gun Law Scorecard",
    legend_title={'text': "Firearm Mortality per 100,000 Persons"}
)

fig.layout.height = 500
fig.layout.width = 800

fig.show()
```



In [ ]: