In [7]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from bokeh.plotting import figure, show
from bokeh.io import output_notebook, output_file, curdoc
#from bokeh.layouts import row, column, widgetbox
#from bokeh.models import Button, CustomJS, ColumnDataSource, CheckboxButtonGroup, Panel, Tabs, Ch
eckboxGroup, Range1d
import chart_studio.plotly as cs
import pandas as pd
import plotly
from plotly import tools
#import plotly.plotly as py
import plotly.graph_objs as go
#plotly.offline.iplot()
plotly.subplots.make_subplots
#plotly.offline.init_notebook_mode(connected=True)
import plotly.offline as py
#py.signin('sanjaysan648', '0NshM3TvkbHRyz133ss4')
```

```
C:\Users\SANJAY\Anaconda3\lib\importlib\_bootstrap.py:219: RuntimeWarning: numpy.ufunc size
changed, may indicate binary incompatibility. Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
```

In [9]:

```
df = pd.read_csv(r'F:\san files\bank challan\dataset\president_general_polls_2016.csv')
df.head()
```

Out[9]:

	cycle	branch	type	matchup	forecastdate	state	startdate	enddate	pollster	grade	...	adjpoll_clinton	adjpoll_trump
0	2016	President	polls-plus	Clinton vs. Trump vs. Johnson	11-08-2016	U.S.	11-03-2016	11-06-2016	ABC News/Washington Post	A+	...	45.20163	41.72430
1	2016	President	polls-plus	Clinton vs. Trump vs. Johnson	11-08-2016	U.S.	11-01-2016	11-07-2016	Google Consumer Surveys	B	...	43.34557	41.21439
2	2016	President	polls-plus	Clinton vs. Trump vs. Johnson	11-08-2016	U.S.	11-02-2016	11-06-2016	Ipsos	A-	...	42.02638	38.81620
3	2016	President	polls-plus	Clinton vs. Trump vs. Johnson	11-08-2016	U.S.	11-04-2016	11-07-2016	YouGov	B	...	45.65676	40.92004
4	2016	President	polls-plus	Clinton vs. Trump vs. Johnson	11-08-2016	U.S.	11-03-2016	11-06-2016	Gravis Marketing	B-	...	46.84089	42.33184

5 rows × 27 columns

In [10]:

```
categories = ['type', 'state', 'enddate', 'pollster', 'grade', 'samplesize',
              'population', 'adjpoll_clinton', 'adjpoll_trump',
              'adjpoll_johnson', 'adjpoll_mcmullin', 'poll_id']
df2 = df.loc[:, categories]
df_po = df2[df2.loc[:,'type']=='polls-only']
df_po = df_po.reset_index(drop=True)
df_po.loc[:,'enddate'] = pd.to_datetime(df_po.loc[:,'enddate'])
```

In [11]:

```
fig = {
    'data':[
        #Clinton data
        {
            'name':"Clinton",
            'x':df_po.loc[:,'enddate'],
            'y':df_po.loc[:,'adjpoll_clinton'],
            'mode':'markers', 'marker':{'color':'blue', 'opacity':0.1}},
        #Trump data
        {
            'name':"Trump",
            'x':df_po.loc[:,'enddate'],
            'y':df_po.loc[:,'adjpoll_trump'],
            'mode':'markers', 'marker':{'color':'red', 'opacity':0.1}},
        #Johnson data
        {
            'name':"Johnson",
            'x':df_po.loc[:,'enddate'],
            'y':df_po.loc[:,'adjpoll_johnson'],
            'mode':'markers', 'marker':{'color':'gold', 'opacity':0.1}},
        #McMullin data
        {
            'name':"McMullin",
            'x':df_po.loc[:, 'enddate'],
            'y':df_po.loc[:,'adjpoll_mcmullin'],
            'mode':'markers', 'marker':{'color':'green', 'opacity':0.1}}
    ],
    #set graph layout
    'layout':{
        'title':"Adjusted Poll Data",
        #set x-axis default range from one month before the first observation, until one month
after the last observation
        'xaxis':{'title':"Date",
                 'range':[min(df_po.loc[:, 'enddate']) - pd.DateOffset(months=1),
                                max(df_po.loc[:, 'enddate']) + pd.DateOffset(months=1)]},
        #set y-axis default range from 0 to 100, with tick marks every 10 percentage points
        'yaxis':{'title':"Percentage",
                 'range':[0, 100], 'tick0':0, 'dtick':10},
        #set background color
        'plot_bgcolor':'ghostwhite', 'paper_bgcolor':'ghostwhite'}
}

#plot the data
py.iplot(fig, filename='Election Poll Data')
```

In [21]:

```
df_po.sort_values(by='enddate', inplace=True)
clinton_upper = go.Scatter(
    name="Clinton Upper Bound",
    x=df_po.loc[:,'enddate'],
    y=df_po.loc[:,'adjpoll_clinton'] + np.std(df_po.loc[:,'adjpoll_clinton']),
    mode='lines', marker=dict(color='blue'),
    line=dict(width=0),
    fillcolor='rgba(0, 0, 255, 0.2)',
    fill='tonexty')
clinton = go.Scatter(
    name="Clinton Mean",
    x=df_po.loc[:,'enddate'],
    y=np.mean(df_po.loc[:,'adjpoll_clinton']),
    mode='lines', marker=dict(color='blue'),
    line=dict(color='blue'),
    fillcolor='rgba(0,0,255,0.2)',
    fill='tonexty')
clinton_lower = go.Scatter(
    name="Clinton Lower Bound",
    x=df_po.loc[:,'enddate'],
    y=df_po.loc[:,'adjpoll_clinton'] - np.std(df_po.loc[:,'adjpoll_clinton']),
    mode='lines', marker=dict(color='blue'),
    line=dict(width=0),
    fillcolor='rgba(0,0,255,0.2)',
    fill='tonexty')

trump_upper = go.Scatter(
    name="Trump Upper Bound",
    x=df_po.loc[:,'enddate'],
    y=df_po.loc[:,'adjpoll_trump'] + np.std(df_po.loc[:,'adjpoll_trump']),
    mode='lines', marker=dict(color='red'),
    line=dict(width=0),
    fillcolor='rgba(255, 0, 0, 0.2)',
    fill='tonexty'
)
trump = go.Scatter(
    name="Trump Mean",
    x=df_po.loc[:,'enddate'],
    y=np.mean(df_po.loc[:,'adjpoll_trump']),
    mode='lines', marker=dict(color='red'),
    line=dict(color='red'),
    fillcolor='rgba(255, 0, 0, 0.2)',
    fill='tonexty'
)
trump_lower = go.Scatter(
    name="Trump Lower Bound",
    x=df_po.loc[:,'enddate'],
    y=df_po.loc[:,'adjpoll_trump'] - np.std(df_po.loc[:,'adjpoll_trump']),
    mode='lines', marker=dict(color='red'),
    line=dict(width=0),
    fillcolor='rgba(255, 0, 0, 0.2)'
)

data = [clinton_lower, clinton, clinton_upper, trump_lower, trump, trump_upper]
layout = go.Layout(
    title="Election Poll Means with Standard Deviation",
    xaxis=dict(range=(min(df_po.loc[:, 'enddate']) - pd.DateOffset(months=1),
                      max(df_po.loc[:, 'enddate']) + pd.DateOffset(months=1))),
    yaxis=dict(title="Percentage", range=[0,100], tick0=0, dtick=10),
    plot_bgcolor='ghostwhite', paper_bgcolor='ghostwhite')

fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='Election Poll Data, Continuous Error Chart')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-21-df62eb0ffd6a> in <module>()
     15     line=dict(color='blue'),
     16     fillcolor='rgba(0,0,255,0.2)',
```

```
---> 17     fill='tonexty')
     18 clinton_lower = go.Scatter(
     19     name="Clinton Lower Bound",

~\Anaconda3\lib\site-packages\plotly\graph_objs\__init__.py in __init__(self, arg, cliponaxis,
connectgaps, customdata, customdatasrc, dx, dy, error_x, error_y, fill, fillcolor, groupnorm, hove
rinfo, hoverinfosrc, hoverlabel, hoveron, hovertemplate, hovertemplatesrc, hovertext,
hovertextsrc, ids, idssrc, legendgroup, line, marker, meta, metasrc, mode, name, opacity,
orientation, r, rsrc, selected, selectedpoints, showlegend, stackgaps, stackgroup, stream, t,
text, textfont, textposition, textpositionsrc, textsrc, tsrc, uid, uirevision, unselected,
visible, x, x0, xaxis, xcalendar, xsrc, y, y0, yaxis, ycalendar, ysrc, **kwargs)
  39602         self["xsrc"] = xsrc if xsrc is not None else _v
  39603         _v = arg.pop("y", None)
> 39604         self["y"] = y if y is not None else _v
  39605         _v = arg.pop("y0", None)
  39606         self["y0"] = y0 if y0 is not None else _v

~\Anaconda3\lib\site-packages\plotly\basedatatypes.py in __setitem__(self, prop, value)
   3347             # ### Handle simple property ###
   3348             else:
-> 3349                 self._set_prop(prop, value)
   3350
   3351         # Handle non-scalar case

~\Anaconda3\lib\site-packages\plotly\basedatatypes.py in _set_prop(self, prop, val)
   3630                 return
   3631             else:
-> 3632                 raise err
   3633
   3634         # val is None

~\Anaconda3\lib\site-packages\plotly\basedatatypes.py in _set_prop(self, prop, val)
   3625         validator = self._validators.get(prop)
   3626         try:
-> 3627             val = validator.validate_coerce(val)
   3628         except ValueError as err:
   3629             if self._skip_invalid:

~\Anaconda3\lib\site-packages\_plotly_utils\basevalidators.py in validate_coerce(self, v)
    387             v = to_scalar_or_list(v)
    388         else:
--> 389             self.raise_invalid_val(v)
    390         return v
    391

~\Anaconda3\lib\site-packages\_plotly_utils\basevalidators.py in raise_invalid_val(self, v, inds)
    281                 typ=type_str(v),
    282                 v=repr(v),
--> 283                 valid_clr_desc=self.description(),
    284             )
    285         )

ValueError:
    Invalid value of type 'builtins.float' received for the 'y' property of scatter
        Received value: 43.3225174952472

    The 'y' property is an array that may be specified as a tuple,
    list, numpy array, or pandas Series
```

In [22]:

```
df_date = df_po[df_po.loc[:,'state']=='U.S.']
df_date = df_date.loc[:,['enddate', 'adjpoll_clinton', 'adjpoll_trump']]
df_date = df_date.reset_index(drop=True)

for index in range(len(df_date)):
    d = df_date.loc[index, 'enddate']
    if (d.strftime("%d") <= '05'):
        df_date.loc[index, 'enddate']=d.replace(day=1)
    elif (d.strftime("%d") <= '10'):
        df_date.loc[index, 'enddate']=d.replace(day=6)
    elif (d.strftime("%d") <= '15'):
        df_date.loc[index, 'enddate']=d.replace(day=11)
    elif (d.strftime("%d") <= '20'):
        df_date.loc[index, 'enddate']=d.replace(day=16)
    elif (d.strftime("%d") <= '25'):
```

```
        df_date.loc[index, 'enddate']=d.replace(day=21)
    elif (d.strftime("%d") <= '31'):
        df_date.loc[index, 'enddate']=d.replace(day=26)

df_bins = pd.DataFrame(columns=('date', 'mean_c', 'mean_t', 'sd_c', 'sd_t'))
while (len(df_date) != 0):
    row0 = df_date.iloc[0,:]
    d = row0.loc['enddate']
    mean_c=np.mean(df_date.loc[:,'adjpoll_clinton'][df_date.loc[:,'enddate']==d])
    mean_t=np.mean(df_date.loc[:,'adjpoll_trump'][df_date.loc[:,'enddate']==d])
    sd_c = np.std(df_date.loc[:,'adjpoll_clinton'][df_date.loc[:,'enddate']==d])
    sd_t = np.std(df_date.loc[:,'adjpoll_trump'][df_date.loc[:,'enddate']==d])
    df_bins.loc[df_bins.shape[0]] = [d, mean_c, mean_t, sd_c, sd_t]
    df_date = df_date[df_date.loc[:,'enddate']!=d]
    df_po = df_po.reset_index(drop=True)

df_bins.sort_values(by='date', inplace=True)
df_bins.sort_values(by='date', inplace=True)
df_bins.head()
```

Out[22]:

	date	mean_c	mean_t	sd_c	sd_t
0	2015-11-16	43.585107	44.713320	1.307381	1.470704
1	2015-11-26	47.270880	40.884840	0.000000	0.000000
2	2015-12-01	48.883163	41.659147	1.055642	2.413191
3	2015-12-06	46.479255	44.982312	3.397378	2.493830
4	2015-12-11	47.555180	43.771135	0.487900	0.180635

In [19]:

```
clinton_upper = go.Scatter(
    name="Clinton Upper Bound",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_c'] + df_bins.loc[:,'sd_c'],
    mode='lines', marker=dict(color='blue'),
    line=dict(width=0),
    fillcolor='rgba(0, 0, 255, 0.2)',
    fill='tonexty')
clinton = go.Scatter(
    name="Clinton Mean",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_c'],
    mode='lines', marker=dict(color='blue'),
    line=dict(color='blue'),
    fillcolor='rgba(0,0,255,0.2)',
    fill='tonexty')
clinton_lower = go.Scatter(
    name="Clinton Lower Bound",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_c'] - df_bins.loc[:,'sd_c'],
    mode='lines', marker=dict(color='blue'),
    line=dict(width=0),
    fillcolor='rgba(0,0,255,0.2)')

trump_upper = go.Scatter(
    name="Trump Upper Bound",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_t'] + df_bins.loc[:,'sd_t'],
    mode='lines', marker=dict(color='red'),
    line=dict(width=0),
    fillcolor='rgba(255, 0, 0, 0.2)',
    fill='tonexty')
trump = go.Scatter(
    name="Trump Mean",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_t'],
    mode='lines', marker=dict(color='red'),
    line=dict(color='red'),
    fillcolor='rgba(255, 0, 0, 0.2)',
    fill='tonexty')
trump_lower = go.Scatter(
```

```
    name="Trump Lower Bound",
    x=df_bins.loc[:,'date'],
    y=df_bins.loc[:,'mean_t'] - df_bins.loc[:,'sd_t'],
    mode='lines', marker=dict(color='red'),
    line=dict(width=0),
    fillcolor='rgba(255, 0, 0, 0.2)')

data = [clinton_lower, clinton, clinton_upper, trump_lower, trump, trump_upper]
layout = go.Layout(
    title="Election Poll Means with Standard Deviation (Adjusted)",
    xaxis=dict(range=(min(df_po.loc[:, 'enddate']) - pd.DateOffset(months=1),
                      max(df_po.loc[:, 'enddate']) + pd.DateOffset(months=1))),
    yaxis=dict(title="Percentage", range=[0,100], tick0=0, dtick=10),
    plot_bgcolor='ghostwhite', paper_bgcolor='ghostwhite')

fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='Election Poll Data, Continuous Error Chart (Adjusted)')
```

In [14]:

```
df_grade = df_po
for index in range(len(df_grade)):
    grade = df_grade.loc[index, 'grade']
    if (grade=='A-'):                      #change A- grades to A
        df_grade.loc[index, 'grade']='A'
    elif (grade=='B+' or  grade=='B-'):    #change B+ and B- grades to B
        df_grade.loc[index, 'grade']='B'
    elif (grade=='C+' or grade=='C-'):
        df_grade.loc[index, 'grade']='C'  #change C+ and C- grades to C
df_grade.loc[:, 'grade'][df_grade.loc[:, 'grade'].isnull()] = 'NA'     #change empty grades ('nan')
to string 'NA'

df_grade.grade.unique()
```

```
C:\Users\SANJAY\Anaconda3\lib\site-packages\ipykernel_launcher.py:10: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
```

Out[14]:

```
array(['A+', 'B', 'A', 'NA', 'C', 'D'], dtype=object)
```

In [17]:

```
trace_CAp=go.Scatter()
trace_CA=go.Scatter()
trace_CB=go.Scatter()
trace_CC=go.Scatter()
trace_CD=go.Scatter()
trace_CNA=go.Scatter()
trace_TAp=go.Scatter()
trace_TA=go.Scatter()
trace_TB=go.Scatter()
trace_TC=go.Scatter()
trace_TD=go.Scatter()
trace_TNA=go.Scatter()
tracesC = [trace_CAp, trace_CA, trace_CB, trace_CC, trace_CD, trace_CNA]
tracesT = [trace_TAp, trace_TA, trace_TB, trace_TC, trace_TD, trace_TNA]
candidates = {'clinton':tracesC, 'trump':tracesT}
grades = ['A+', 'A', 'B', 'C', 'D', 'NA']

#Assigning attributes for each trace
def gradeSubsetX(grade):
    return df_grade.loc[:, 'enddate'][df_grade.loc[:, 'grade']==grade]
def gradeSubsetY(grade, name):
    return df_grade.loc[:, 'adjpoll_'+name][df_grade.loc[:, 'grade']==grade]

for name, traceList in candidates.items():
    i=0
    for trace in traceList:
        trace.name = name.title() + ' ' + grades[i]
        trace.x = gradeSubsetX(grades[i])
        trace.y = gradeSubsetY(grades[i], name)
        trace.mode = 'markers'
        trace.marker = dict(color='blue' if name=='clinton' else 'red',
                            opacity=0.1)
        i+=1

#Creating subplots and appending traces to each subplot
fig = tools.make_subplots(rows=3, cols=2, subplot_titles=('A+', 'A', 'B',
                                                          'C', 'D', 'NA'))

i=0
for j in range(1,4):
    for k in range(1, 3):
        fig.append_trace(tracesC[i], j, k)
        fig.append_trace(tracesT[i], j, k)
        i += 1

#Organizing layout for each subplot
xaxis = dict(range=(min(df_grade.loc[:, 'enddate']) - pd.DateOffset(months=1),
                    max(df_grade.loc[:, 'enddate']) + pd.DateOffset(months=1)))
yaxis = dict(range=[0,100], tick0=0, dtick=25)
layout = go.Layout(
    title="Clinton and Trump, By Pollster Grade",
    xaxis1=xaxis,
    xaxis2=xaxis,
    xaxis3=xaxis,
    xaxis4=xaxis,
    xaxis5=xaxis,
    xaxis6=xaxis,
    yaxis1=yaxis,
    yaxis2=yaxis,
    yaxis3=yaxis,
    yaxis4=yaxis,
    yaxis5=yaxis,
    yaxis6=yaxis,
    plot_bgcolor='ghostwhite', paper_bgcolor='ghostwhite')
fig['layout'].update(layout)

py.iplot(fig, filename='Election Poll Data, By Grade')
```

```
C:\Users\SANJAY\Anaconda3\lib\site-packages\plotly\tools.py:465: DeprecationWarning:
```

```
plotly.tools.make_subplots is deprecated, please use plotly.subplots.make_subplots instead
```

In [39]:

```
df_states = df_po.sort_values(by='state')

trace0 = go.Scatter(
        x = df_states.loc[:,'state'],
        y = df_states.loc[:,'adjpoll_clinton'],
        mode = 'markers',
        name = 'Clinton',
        marker = dict(color = 'blue', opacity = 0.1))

trace1 = go.Scatter(
        x = df_states.loc[:,'state'],
        y = df_states.loc[:,'adjpoll_trump'],
        mode = 'markers',
        name = 'Trump',
        marker = dict(color = 'red', opacity = 0.1))

title = "Adjusted Poll Data by State"
x_title = "State"
y_title = "Percentage"

layout = go.Layout(
    title=title,
    xaxis=go.XAxis( title=x_title),
    yaxis=go.YAxis(title=y_title),
    hovermode = 'closest'
)


data = go.Data([trace0,trace1])

fig = go.Figure(data=data, layout=layout)
py.iplot(fig)
```

```
C:\Users\SANJAY\Anaconda3\lib\site-packages\plotly\graph_objs\_deprecations.py:550:
DeprecationWarning:

plotly.graph_objs.XAxis is deprecated.
Please replace it with one of the following more specific types
  - plotly.graph_objs.layout.XAxis
```

```
  - plotly.graph_objs.layout.scene.XAxis


C:\Users\SANJAY\Anaconda3\lib\site-packages\plotly\graph_objs\_deprecations.py:578:
DeprecationWarning:

plotly.graph_objs.YAxis is deprecated.
Please replace it with one of the following more specific types
  - plotly.graph_objs.layout.YAxis
  - plotly.graph_objs.layout.scene.YAxis


C:\Users\SANJAY\Anaconda3\lib\site-packages\plotly\graph_objs\_deprecations.py:40:
DeprecationWarning:

plotly.graph_objs.Data is deprecated.
Please replace it with a list or tuple of instances of the following types
  - plotly.graph_objs.Scatter
  - plotly.graph_objs.Bar
  - plotly.graph_objs.Area
  - plotly.graph_objs.Histogram
  - etc.
```

In [40]:

```
state_dict = {'Alabama':'AL', 'Alaska':'AK', 'Arizona':'AZ', 'Arkansas':'AR',
                'California':'CA', 'Colorado':'CO', 'Connecticut':'CT',
                'Delaware':'DE', 'District of Columbia':'DC', 'Florida':'FL',
                'Georgia':'GA','Hawaii':'HI', 'Idaho':'ID', 'Illinois':'IL',
                'Indiana':'IN', 'Iowa':'IA', 'Kansas':'KS', 'Kentucky':'KY',
                 'Louisiana':'LA', 'Maine':'ME', 'Maryland':'MD',
                 'Massachusetts':'MA', 'Michigan':'MI', 'Minnesota':'MN',
                'Mississippi':'MS', 'Missouri':'MO', 'Montana':'MT',
                'Nebraska':'NE', 'Nevada':'NV', 'New Hampshire':'NH',
                'New Jersey':'NJ', 'New Mexico':'NM', 'New York':'NY',
                'North Carolina':'NC', 'North Dakota':'ND','Ohio':'OH',
                'Oklahoma':'OK', 'Oregon':'OR', 'Pennsylvania':'PA',
                'Rhode Island':'RI', 'South Carolina':'SC', 'South Dakota':'SD',
                'Tennessee':'TN', 'Texas':'TX', 'Utah':'UT', 'Vermont':'VT',
                'Virginia':'VA', 'Washington':'WA', 'West Virginia':'WV',
                'Wisconsin':'WI', 'Wyoming':'WY'}
```

```
data_list = []
for state, code in state_dict.items():
    dict1 = {}
    samplesize = df_po[df_po.loc[:,'state']==state].loc[:,'samplesize']
    clinton = df_po[df_po.loc[:,'state']==state].loc[:,'adjpoll_clinton']
    trump = df_po[df_po.loc[:,'state']==state].loc[:,'adjpoll_trump']
    dict1['state'] = state
    dict1['code'] = code
    dict1['samplesize'] = samplesize.sum()
    dict1['clinton'] = (clinton * samplesize).sum()
    dict1['trump'] = (trump * samplesize).sum()
    data_list.append(dict1)

df_map = pd.DataFrame(data_list, columns=['code', 'state', 'samplesize', 'clinton', 'trump'])
df_map.loc[:,'clinton_pct'] = df_map.loc[:,'clinton']/df_map.loc[:,'samplesize']
df_map.loc[:,'trump_pct'] = df_map.loc[:,'trump']/df_map.loc[:,'samplesize']
df_map.head()
```

Out[40]:

	code	state	samplesize	clinton	trump	clinton_pct	trump_pct
0	AL	Alabama	27153.0	8.992856e+05	1.570689e+06	33.119197	57.845873
1	AK	Alaska	12240.0	4.309978e+05	5.495426e+05	35.212235	44.897274
2	AZ	Arizona	69177.0	2.919496e+06	3.068696e+06	42.203272	44.360056
3	AR	Arkansas	17965.0	6.353103e+05	9.472248e+05	35.363781	52.726123
4	CA	California	103154.0	5.588077e+06	3.374816e+06	54.172181	32.716288

In [43]:

```
results = pd.read_csv(r'F:\san files\bank challan\dataset\results.csv')
results = results.sort_values('state')
results = results.reset_index(drop=True)
results.head()
```

Out[43]:

	state	clinton	trump
0	Alabama	34.4	62.1
1	Alaska	36.6	51.3
2	Arizona	45.1	48.7
3	Arkansas	33.7	60.6
4	California	61.7	31.6

In [44]:

```
df_map['clinton_diff'] = results['clinton'] - df_map['clinton_pct']
df_map['trump_diff'] = results['trump'] - df_map['trump_pct']
df_map.head()
```

Out[44]:

	code	state	samplesize	clinton	trump	clinton_pct	trump_pct	clinton_diff	trump_diff
0	AL	Alabama	27153.0	8.992856e+05	1.570689e+06	33.119197	57.845873	1.280803	4.254127
1	AK	Alaska	12240.0	4.309978e+05	5.495426e+05	35.212235	44.897274	1.387765	6.402726
2	AZ	Arizona	69177.0	2.919496e+06	3.068696e+06	42.203272	44.360056	2.896728	4.339944
3	AR	Arkansas	17965.0	6.353103e+05	9.472248e+05	35.363781	52.726123	-1.663781	7.873877
4	CA	California	103154.0	5.588077e+06	3.374816e+06	54.172181	32.716288	7.527819	-1.116288

In [45]:

```
scl_c = [[0.0, 'red'], [0.5, 'white'], [1.0, 'blue']]
```

```
data = [dict(
        type='choropleth',
        colorscale = scl_c, autocolorscale = False,
        locations = df_map['code'],
        z = df_map['clinton_diff'],
        zmin = -10, zmax =  10,
        locationmode = 'USA-states',
        #text = df['text'],
        marker = dict(
            line = dict (
                color = 'rgb(255,255,255)',
                width = 2
            ) ),
        colorbar = dict(
            title = "Percentage difference")
        ) ]

layout = dict(
        title = 'Clinton Poll + Result difference',
        geo = dict(
            scope='usa',
            projection=dict( type='albers usa' ),
            showlakes = True,
            lakecolor = 'rgb(255, 255, 255)'),
             )

fig = dict( data=data, layout=layout )
py.iplot( fig, filename='Clinton Poll + Result difference' )
```

In [46]:

```
scl_t = [[0.0, 'blue'], [0.5, 'white'], [1.0, 'red']]

data = [dict(
        type='choropleth',
        colorscale = scl_t, autocolorscale = False,
        locations = df_map['code'],
        z = df_map['trump_diff'],
        zmin = -10, zmax =  10,
        locationmode = 'USA-states',
        #text = df['text'],
        marker = dict(
            line = dict (
                color = 'rgb(255,255,255)',
```

```
                width = 2
            ) ),
        colorbar = dict(
            title = "Percentage difference")
        ) ]

layout = dict(
        title = 'Trump Poll + Result difference',
        geo = dict(
            scope='usa',
            projection=dict( type='albers usa' ),
            showlakes = True,
            lakecolor = 'rgb(255, 255, 255)'),
             )

fig = dict( data=data, layout=layout )
py.iplot( fig, filename='Trump Poll + Result difference' )
```

In [ ]:

```

```