Second Dataset

February 21, 2021

[12]:
```
import numpy as np
import pandas as pd
import pickle
from statsmodels import tsa
import statsmodels.api as sm
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels import multivariate
from statsmodels import regression
import scipy.stats as stats
from statsmodels.sandbox.regression import gmm
from statsmodels.sandbox.regression.gmm import GMM
import statsmodels.stats.diagnostic as smd
from statsmodels.tsa.adfvalues import mackinnonp, mackinnoncrit
from statsmodels.tsa.stattools import adfuller
import statsmodels.tsa.api as smt
from statsmodels.tsa.vector_ar.hypothesis_test_results import␣
 ↪CausalityTestResults
from statsmodels.tsa.vector_ar.var_model import VAR, VARProcess, VARResults
from statsmodels.tsa.vector_ar.vecm import VECM, coint_johansen, select_order
import statsmodels.tsa.arima_model as am
from statsmodels.regression.rolling import RollingOLS

from tabulate import tabulate

import datetime as dt
from dateutil.relativedelta import relativedelta
from datetime import timedelta



import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.dates import DateFormatter, MinuteLocator
from matplotlib.ticker import PercentFormatter


import os
import warnings
```

```
from scipy.optimize import minimize, brute
from arch import arch_model


warnings.filterwarnings("ignore")
```

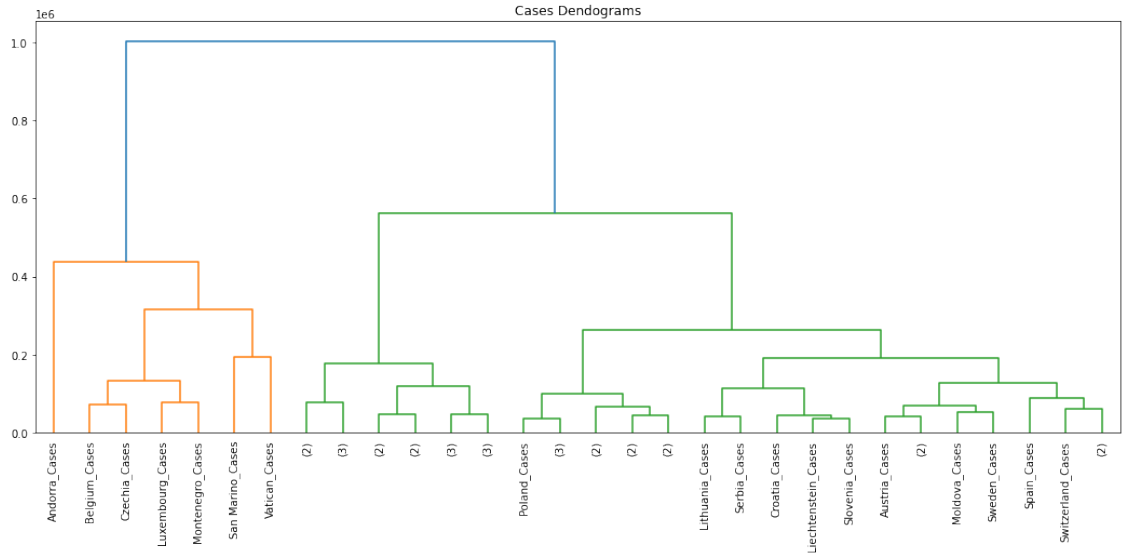
[13]:
```
covid_data = pd.read_csv('/Users/garik/Datathon/owid-covid-data.csv')
eu = covid_data[covid_data.continent == 'Europe']
```

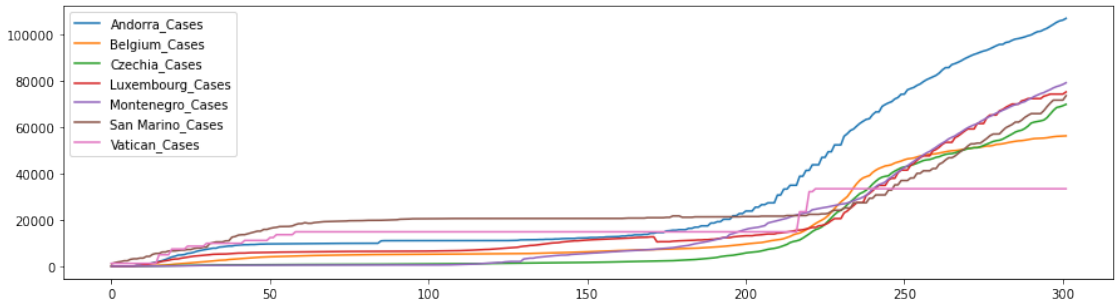
[14]:
```
countries = eu.location.unique()
deaths = []
cases = []
dates = []
for c in countries:
    temp_df = eu[eu.location == c]
    dates.append(temp_df.date.values)
    cases.append(temp_df.total_cases_per_million.values)
    deaths.append(temp_df.total_deaths_per_million.values)
```

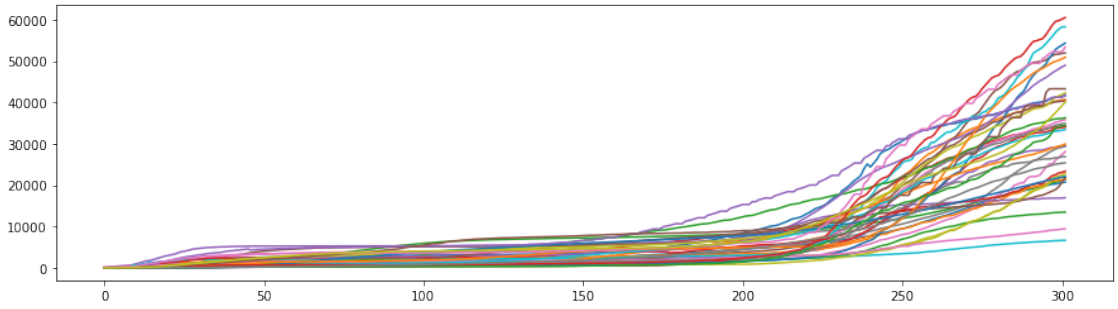
[15]:
```
ans1 = pd.DataFrame([dates[0],cases[0],deaths[0]]).T
ans1.columns = ['Date',countries[0] + '_Cases',countries[0] + '_Deaths']


for i in range(1,len(countries)):
    ans2 = pd.DataFrame([dates[i],cases[i],deaths[i]]).T
    ans2.columns = ['Date',countries[i] + '_Cases',countries[i] + '_Deaths']
    ans2.index = ans2['Date']
    ans1 = ans1.join(ans2,on = 'Date',rsuffix = 'r')

ans1 = ans1.drop(['Dater'],axis = 1)
ans1.iloc[0] = ans1.iloc[0].fillna(0)
ans1 = ans1.ffill()
```

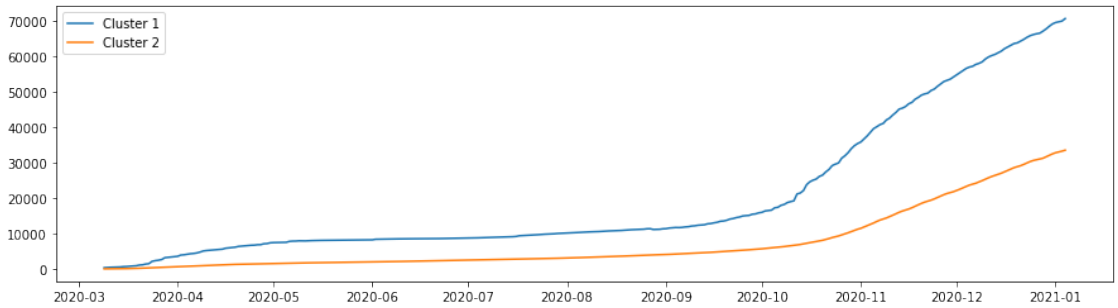
[16]:
```
cl = ans1.iloc[:,1::2]
import scipy.cluster.hierarchy as shc

plt.figure(figsize=(18, 7))
plt.title("Cases Dendograms")
dend = shc.dendrogram(shc.linkage(cl.T, method='ward'),truncate_mode='lastp',
                      orientation='top', show_leaf_counts=True,
                     no_labels = False,labels = cl.columns,leaf_rotation =90)

cluster_1 = [1,4,9,24,28,36,45]
cl_1 = list(np.array(countries[cluster_1]) + "_Cases")

cluster_2 = np.arange(0,cl.shape[1])
```

```
cluster_2= cluster_2[list(map(lambda x: x not in cluster_1,cluster_2))]
cl_2 = list(np.array(countries[cluster_2]) + "_Cases")

series_1 = ans1[list(np.array(countries[cluster_1]) + "_Cases")]

plt.figure(figsize = (15,4))
plt.plot(series_1)
plt.legend(cl_1)
plt.show()

series_2 = ans1[list(np.array(countries[cluster_2]) + "_Cases")]
plt.figure(figsize = (15,4))
plt.plot(series_2)
plt.show()

series_1.index = pd.to_datetime(ans1.Date)
series_2.index = pd.to_datetime(ans1.Date)

plt.figure(figsize = (15,4))
plt.plot(series_1.mean(axis = 1))
plt.plot(series_2.mean(axis = 1))
plt.legend(["Cluster 1","Cluster 2"])
plt.show()
```









[22]:
```
ans1
```

[22]:

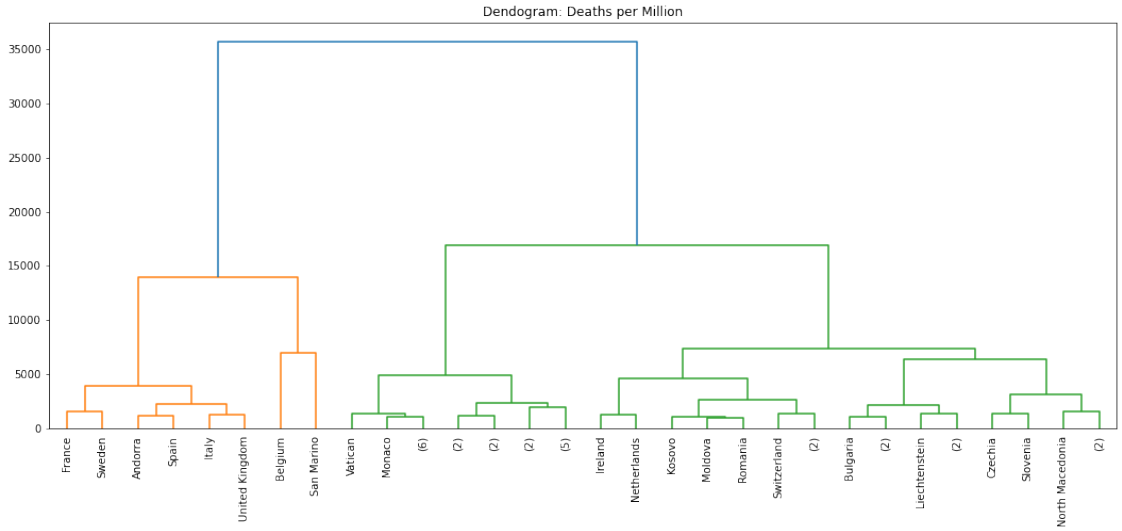
	Date	Albania_Cases	Albania_Deaths	Andorra_Cases	Andorra_Deaths	\
0	2020-03-09	0.695	0.000	12.942	0.000	
1	2020-03-10	3.475	0.000	12.942	0.000	
2	2020-03-11	4.170	0.347	12.942	0.000	
3	2020-03-12	7.992	0.347	12.942	0.000	
4	2020-03-13	11.467	0.347	12.942	0.000	
..	…	…	…	…	…	
297	2020-12-31	20264.091	410.383	104173.947	1087.168	
298	2021-01-01	20264.091	410.383	105054.035	1087.168	
299	2021-01-02	20498.645	413.510	105688.216	1087.168	

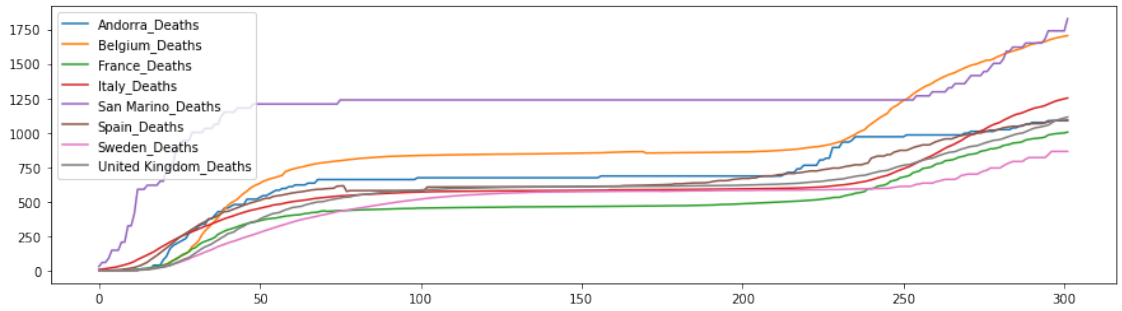
```
300  2021-01-03      20653.972         414.553     106024.720        1087.168
301  2021-01-04      20718.257         416.638     106762.441        1087.168

     Austria_Cases  Austria_Deaths  Belarus_Cases  Belarus_Deaths  \
0           14.545           0.000          0.635           0.000
1           20.208           0.000          0.952           0.000
2           27.314           0.000          0.952           0.000
3           33.532           0.111          1.270           0.000
4           55.960           0.111          2.857           0.000
..             …               …              …               …
297      40062.067         690.842      20560.631         150.699
298      40294.790         695.172      20765.831         151.651
299      40449.236         696.727      20967.115         152.604
300      40612.009         702.167      21161.520         153.556
301      40794.324         705.831      21359.312         154.614

     Belgium_Cases  …  Sweden_Cases  Sweden_Deaths  Switzerland_Cases  \
0           20.622  …        32.280          0.000             43.214
1           23.038  …        41.983          0.099             56.733
2           27.093  …        61.391          0.099             75.336
3           27.093  …        76.342          0.198             75.336
4           48.233  …        91.393          0.297            131.606
..             …  …             …              …                …
297      55782.349  …     43307.982        864.122          52260.654
298      55937.056  …     43307.982        864.122          52260.654
299      56012.986  …     43307.982        864.122          52260.654
300      56085.637  …     43307.982        864.122          52260.654
301      56161.222  …     43307.982        864.122          53377.399

     Switzerland_Deaths  Ukraine_Cases  Ukraine_Deaths  United Kingdom_Cases  \
0                 0.231          0.023           0.000                 9.280
1                 0.347          0.023           0.000                13.095
2                 0.462          0.023           0.000                19.164
3                 0.462          0.023           0.000                26.368
4                 1.271          0.069           0.023                33.438
..                  …              …               …                    …
297             883.343      24854.872         440.872             36770.923
298             890.161      25080.282         444.439             37558.390
299             893.049      25203.779         445.834             38410.598
300             895.129      25315.386         448.852             39223.092
301             914.887      25418.144         450.727             40091.062

     United Kingdom_Deaths  Vatican_Cases  Vatican_Deaths
0                    0.044       1236.094               0
1                    0.103       1236.094               0
2                    0.103       1236.094               0
3                    0.133       1236.094               0
```

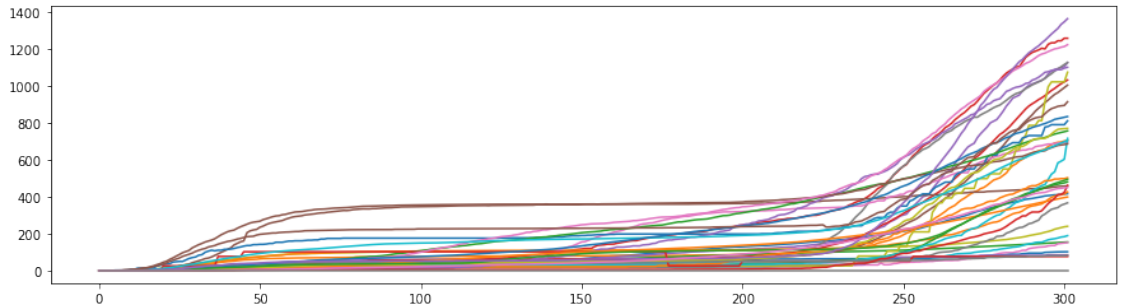
```
4                    0.147       1236.094                0
..                     …           …                   …
297                1084.495      33374.536               0
298                1093.554      33374.536               0
299                1100.109      33374.536               0
300                1106.811      33374.536               0
301                1112.851      33374.536               0

[302 rows x 93 columns]
```

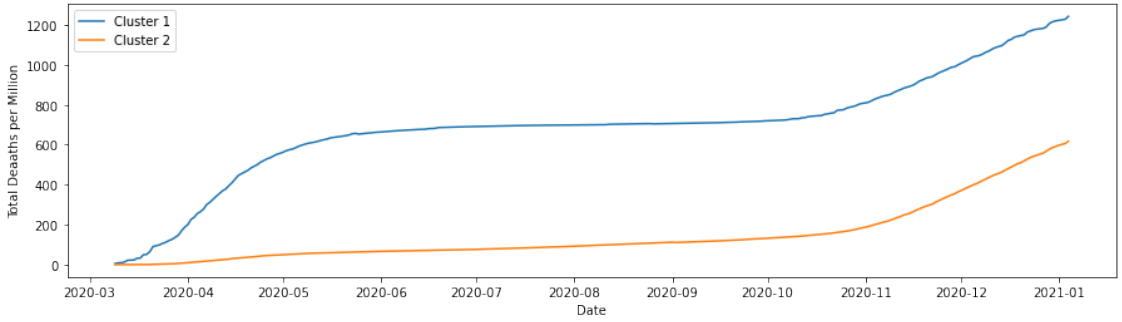
[67]:
```
cl = ans1.iloc[:,2::2]
import scipy.cluster.hierarchy as shc
from scipy.cluster.hierarchy import fcluster


plt.figure(figsize=(18, 7))
plt.title("Dendogram: Deaths per Million")

Z = shc.linkage(cl.T, method='ward')
dend = shc.dendrogram(Z,truncate_mode='lastp',
                      orientation='top', show_leaf_counts=True,
                     no_labels = False,leaf_rotation =90,labels =␣
 ↪list(map(lambda x: x[:-7],cl.columns)))

k = 2
names_cl = fcluster(Z, k, criterion='maxclust')

cluster_1 = cl.columns[names_cl == 1]
cl_1 = list(cluster_1)
cluster_2 = cl.columns[names_cl == 2]
cl_2 =  list(cluster_2)


series_1 = ans1[list(cluster_1)]

plt.figure(figsize = (15,4))
plt.plot(series_1)
plt.legend(cl_1)
plt.show()

series_2 = ans1[list(cluster_2)]
plt.figure(figsize = (15,4))
plt.plot(series_2)
plt.show()

series_1.index = pd.to_datetime(ans1.Date)
series_2.index = pd.to_datetime(ans1.Date)
```

```
plt.figure(figsize = (15,4))
plt.plot(series_1.mean(axis = 1))
plt.plot(series_2.mean(axis = 1))
plt.legend(["Cluster 1","Cluster 2"])
plt.ylabel("Total Deaaths per Million")
plt.xlabel("Date")
plt.show()
```









[63]:
```
data = dict(type = 'choropleth',
            locations = pd.Series(list(map(lambda x: x[:-7],cl.columns))),
            locationmode = 'country names',
            colorscale= 'Portland',
            text= ['IND','NEP','CHI','PAK','BAN','BHU', 'MYN','SLK'],
            z=pd.Series(names_cl),
            colorbar = {'title':'Country Clusters', 'len':200,'lenmode':
 ↪'pixels' })
```

[56]:
```
names_cl
```

[56]:
```
array([2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 1, 1, 2, 2,
       1, 2], dtype=int32)
```

[60]:
```
import plotly.graph_objs as gobj
import pandas as pd
from plotly.offline import download_plotlyjs,init_notebook_mode,plot,iplot
init_notebook_mode(connected=True)
```

[66]:
```
data = dict(type = 'choropleth',
            locations = pd.Series(list(map(lambda x: x[:-7],cl.columns))),
            locationmode = 'country names',
            colorscale= 'Portland',
            text= ['IND','NEP','CHI','PAK','BAN','BHU', 'MYN','SLK'],
            z=pd.Series(names_cl),
            colorbar = {'title':'Country Clusters', 'len':200,'lenmode':
 ↪'pixels' })

layout = dict(geo = {'scope':'europe'})
col_map = gobj.Figure(data = [data],layout = layout)
```

```
iplot(col_map)
```

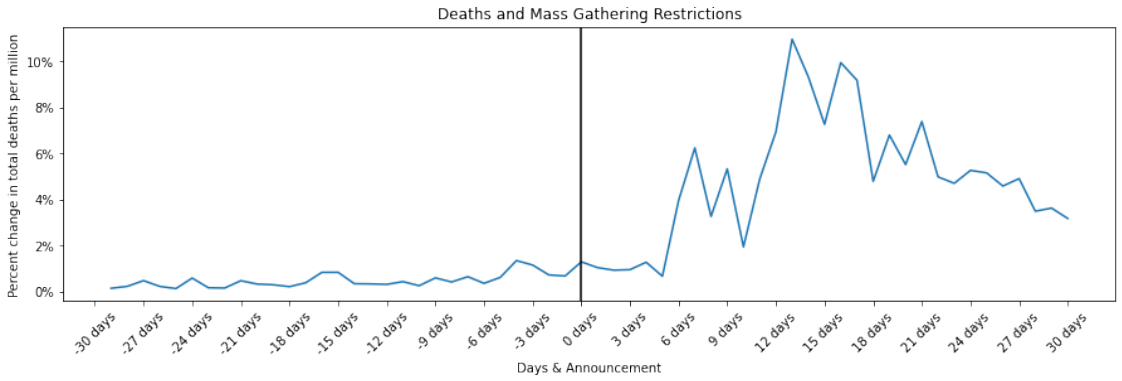
0.0.1 -30,-29,...29,30 days, graphs

[68]:
```
ans1.index = pd.to_datetime(ans1.Date)

cases_data = ans1.iloc[:,1::2]
cols = cases_data.columns
cols_new = list(map(lambda x: x[:-6],cols))
cases_data.columns = cols_new

death_data = ans1.iloc[:,2::2]
cols = death_data.columns
cols_new = list(map(lambda x: x[:-7],cols))
death_data.columns = cols_new


response = pd.read_csv('country_response_measures.csv',parse_dates=True)
response.name = 'response'

response.date_start = pd.to_datetime(response.date_start)
response.date_end = pd.to_datetime(response.date_end)

response = response[response.date_start > min(cases_data.index)]


def compare_measures(response,by_data = death_data, m = 'MassGatherAll',du =␣
↪30,dd = 30):

    mass = response[response.Response_measure == m]
    mass_countries = mass.Country.unique()

    policy_matrix = pd.DataFrame(columns = mass_countries, index= pd.
↪to_datetime(by_data.index)).fillna(0)

    for c in mass_countries:
        df_temp = mass[mass.Country == c]
        try:
            policy_matrix.loc[df_temp.date_start.iloc[0],c] = 1
        except:
            pass
        try:
            policy_matrix.loc[df_temp.date_end.iloc[0],c] = -1
        except:
            pass
```

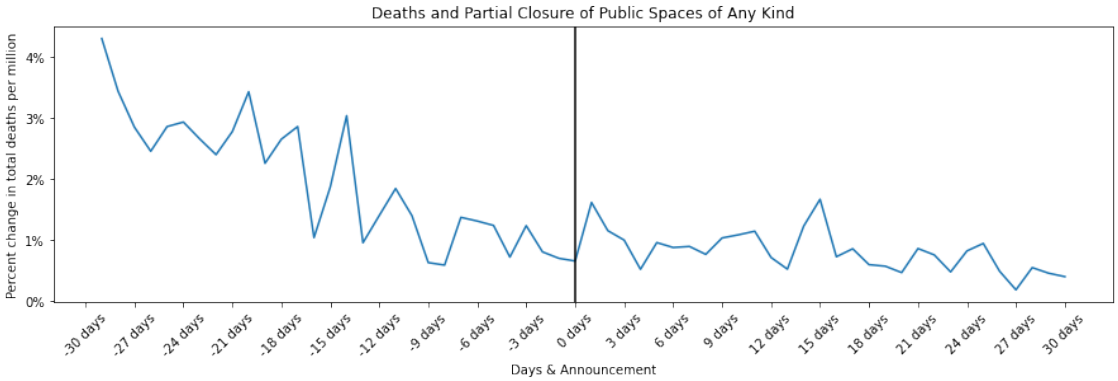
```
    policy_matrix = policy_matrix.dropna()
    changes_before_after = pd.DataFrame(index = np.arange(-dd,du+1),columns =␣
↪policy_matrix.columns)
    for c in policy_matrix.columns:
        row = policy_matrix[c]
        pdate = row[row == 1].index[0]
        temp_change = by_data.loc[pdate - relativedelta(days = du):
                                  pdate + relativedelta(days = dd),c].
↪pct_change()*100

        if(temp_change.shape[0] <= du+dd):
            changes_before_after.loc[0 - temp_change[:pdate].shape[0]:-1,c] =␣
↪temp_change[:pdate].values

            changes_before_after.loc[0: 0 + temp_change[pdate:].shape[0],c] =␣
↪temp_change[pdate:].values
        else:

            changes_before_after.loc[0 - temp_change[:pdate].shape[0]:
                                     -1,c] = temp_change[:
↪pdate-relativedelta(days = 1)].values

            changes_before_after.loc[0: 0 + temp_change[pdate:].shape[0],c] =␣
↪temp_change[pdate:].values

    return changes_before_after.replace([np.inf, -np.inf], np.nan)
```

[101]:
```
ev_name = 'MassGatherAll'
title = 'Mass Gathering Restrictions'
dd = 30
du = 30


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
```

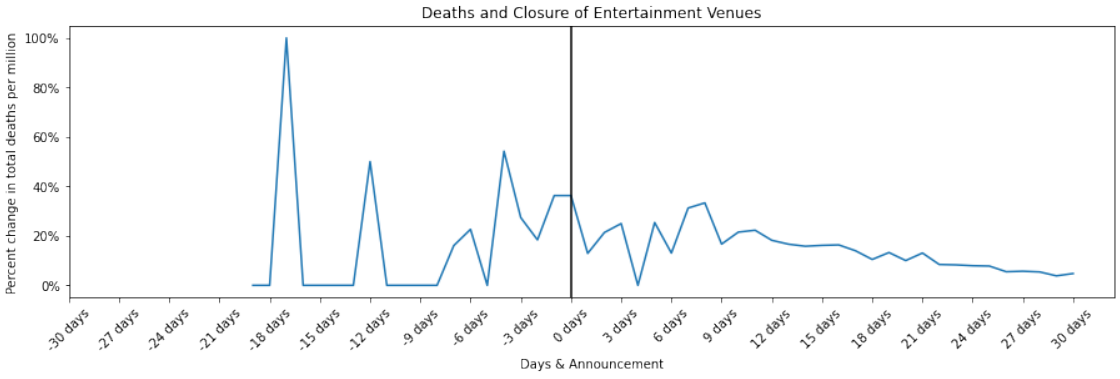
```
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```



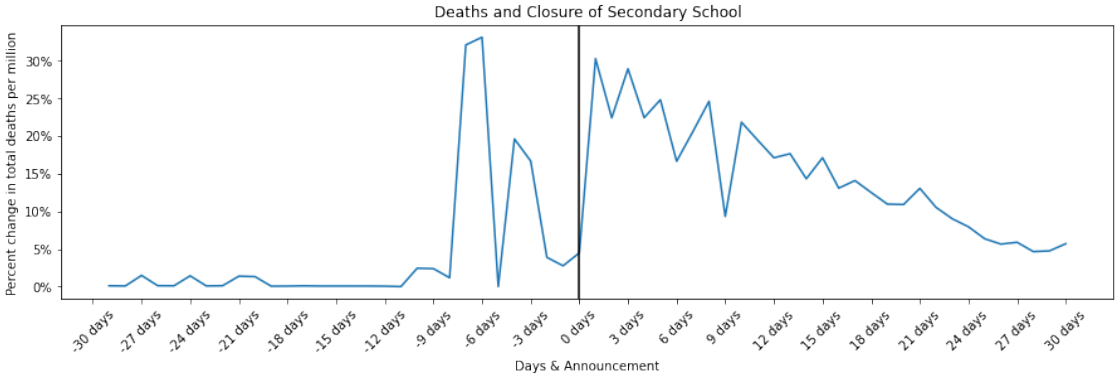
[103]:
```
ev_name = 'ClosPubAnyPartial'
title = 'Partial Closure of Public Spaces of Any Kind'
dd = 30
du = 30


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```



[104]:
```
ev_name = 'EntertainmentVenues'
title = 'Closure of Entertainment Venues'



plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```

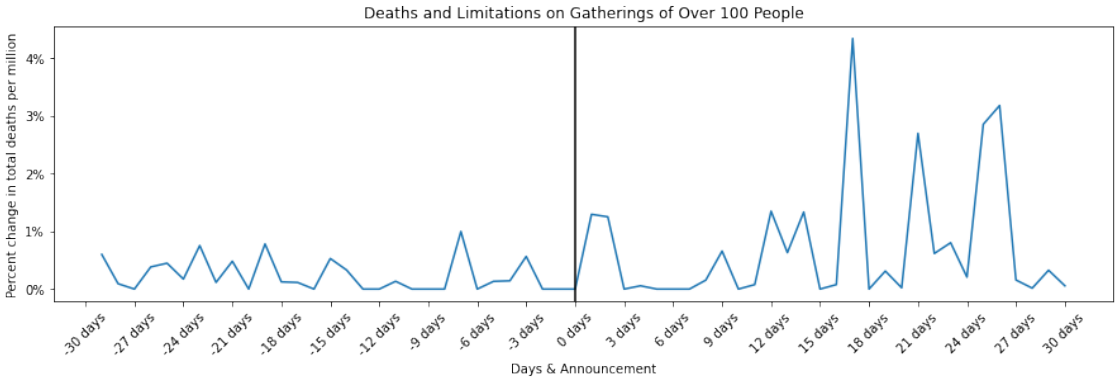


[105]:
```
ev_name = 'ClosSec'
title = 'Closure of Secondary School'


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```



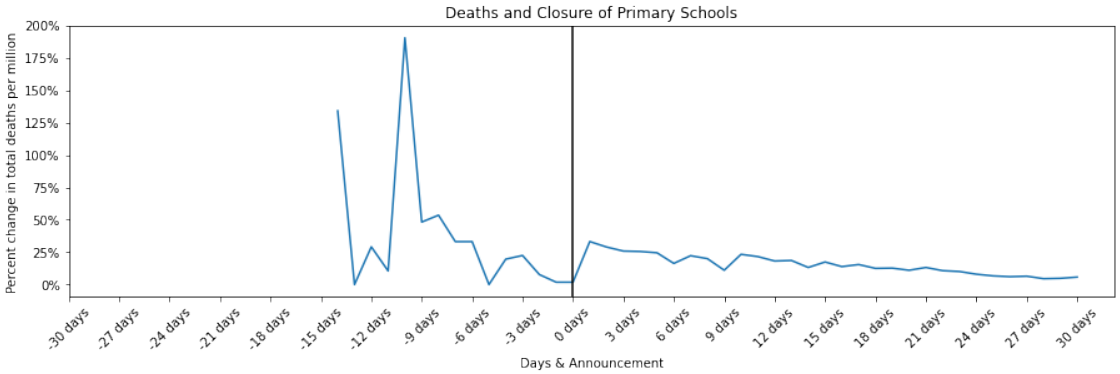
[114]:
```
ev_name = 'IndoorOver100'
title = 'Limitations on Gatherings of Over 100 People'


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()
```

```
plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
           #'50% quantile',
           #'90% quantile'])
plt.show()
```

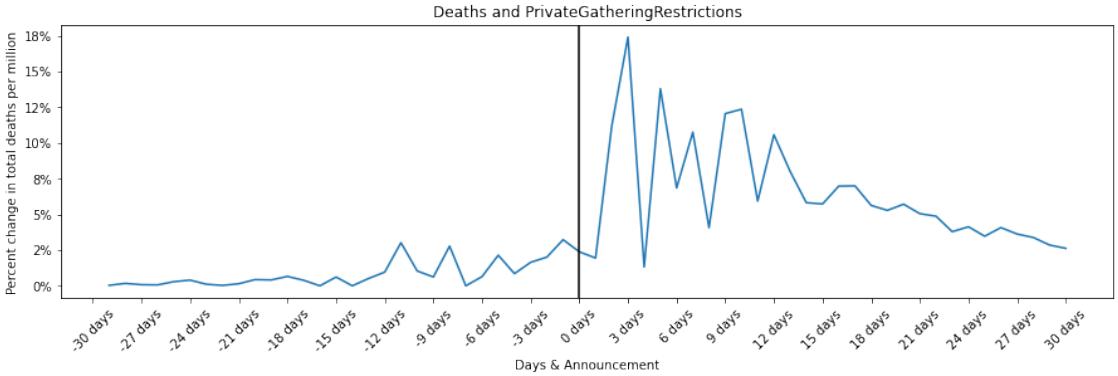


[115]:
```
ev_name = 'ClosPrim'
title = 'Closure of Primary Schools'


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
           #'50% quantile',
           #'90% quantile'])
```

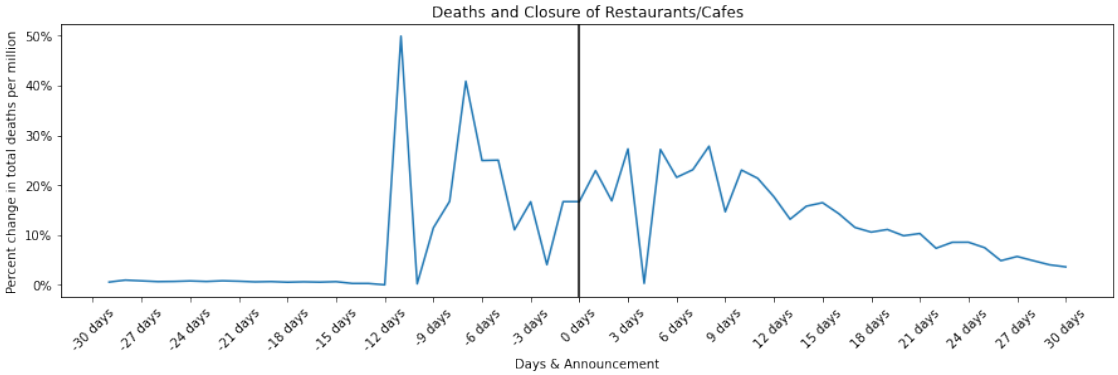
```
plt.show()
```



[117]:
```
ev_name = 'PrivateGatheringRestrictions'
title = 'PrivateGatheringRestrictions'


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```



[119]:
```
ev_name = 'RestaurantsCafes'
title = 'Closure of Restaurants/Cafes'


plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()

plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```



0.1 Clusters & Measures

[106]:
```
response['days_between'] = list(map(lambda x: x.
 ↪days,(response['date_end']-response['date_start'])))
```

[107]:
```
cluster_1_response = response[response['Country'].isin(c1)]
cluster_2_response = response[response['Country'].isin(c2)]
```

[108]:
```
analysis = (pd.DataFrame(cluster_1_response.Response_measure.value_counts()/
 ↪len(c1)).
 join(cluster_2_response.Response_measure.value_counts()/len(c2),rsuffix = '2'))
analysis.columns = ['Cluster 1 av measure','Cluster 2 av measure']

analysis = analysis.join(cluster_1_response.groupby('Response_measure').
 ↪days_between.mean())
analysis = analysis.join(cluster_2_response.groupby('Response_measure').
 ↪days_between.mean(),rsuffix = '2')
analysis.columns = ['Cluster 1 av measure','Cluster 2 av measure','Cluster 1 av␣
 ↪days','Cluster 2 av days']
```

[109]:
```
analysis = analysis.sort_values(by = 'Cluster 2 av measure',ascending = False)
analysis['diff_measure'] = analysis['Cluster 2 av measure'] - analysis['Cluster␣
 ↪1 av measure']
analysis['diff_days'] = analysis['Cluster 2 av days'] - analysis['Cluster 1 av␣
 ↪days']
```

[110]:
```
display(analysis.sort_values(by = 'diff_measure',ascending = False)[:10])
display(analysis.sort_values(by = 'diff_measure',ascending = False)[-10:])
```

```
                           Cluster 1 av measure  Cluster 2 av measure  \
IndoorOver100                             0.125              0.578947
OutdoorOver500                            0.125              0.552632
IndoorOver50                              0.250              0.526316
ClosPrim                                  0.625              0.894737
MassGatherAllPartial                      0.375              0.578947
ClosHigh                                  0.750              0.921053
StayHomeGen                               0.250              0.421053
MassGather50Partial                       0.250              0.368421
BanOnAllEventsPartial                     0.125              0.236842
PrivateGatheringRestrictions              0.875              0.973684

                           Cluster 1 av days  Cluster 2 av days  \
IndoorOver100                      19.000000          46.100000
```

```
OutdoorOver500                      123.000000          44.578947
IndoorOver50                         74.000000          53.933333
ClosPrim                             98.600000          82.615385
MassGatherAllPartial                 66.333333          81.312500
ClosHigh                            116.333333          89.000000
StayHomeGen                           8.000000          40.785714
MassGather50Partial                  26.500000          57.400000
BanOnAllEventsPartial                23.000000          55.833333
PrivateGatheringRestrictions         98.333333          56.550000

                            diff_measure  diff_days
IndoorOver100                   0.453947  27.100000
OutdoorOver500                  0.427632 -78.421053
IndoorOver50                    0.276316 -20.066667
ClosPrim                        0.269737 -15.984615
MassGatherAllPartial            0.203947  14.979167
ClosHigh                        0.171053 -27.333333
StayHomeGen                     0.171053  32.785714
MassGather50Partial             0.118421  30.900000
BanOnAllEventsPartial           0.111842  32.833333
PrivateGatheringRestrictions    0.098684 -41.783333

                            Cluster 1 av measure  Cluster 2 av measure  \
MasksMandatoryClosedSpaces                 0.625              0.421053
OutdoorOver1000                            0.500              0.263158
MasksVoluntaryClosedSpaces                 0.375              0.131579
BanOnAllEvents                             0.875              0.552632
NonEssentialShopsPartial                   0.875              0.526316
SocialCircle                               0.500              0.078947
StayHomeOrder                              0.875              0.394737
Teleworking                                1.000              0.447368
StayHomeOrderPartial                       0.875              0.289474
RegionalStayHomeOrderPartial               0.250                   NaN

                            Cluster 1 av days  Cluster 2 av days  \
MasksMandatoryClosedSpaces          24.000000         119.333333
OutdoorOver1000                     95.000000          73.000000
MasksVoluntaryClosedSpaces          34.666667          81.500000
BanOnAllEvents                      78.500000          78.818182
NonEssentialShopsPartial            26.200000          51.625000
SocialCircle                              NaN                NaN
StayHomeOrder                       47.428571          42.250000
Teleworking                         70.250000          72.875000
StayHomeOrderPartial                27.666667          16.000000
RegionalStayHomeOrderPartial        26.500000                NaN

                            diff_measure  diff_days
MasksMandatoryClosedSpaces      -0.203947  95.333333
```

```
OutdoorOver1000                  -0.236842 -22.000000
MasksVoluntaryClosedSpaces       -0.243421  46.833333
BanOnAllEvents                   -0.322368   0.318182
NonEssentialShopsPartial         -0.348684  25.425000
SocialCircle                     -0.421053        NaN
StayHomeOrder                    -0.480263  -5.178571
Teleworking                      -0.552632   2.625000
StayHomeOrderPartial             -0.585526 -11.666667
RegionalStayHomeOrderPartial           NaN        NaN
```

[111]:
```
display(analysis.sort_values(by = 'diff_days',ascending = False)[:10])
display(analysis.sort_values(by = 'diff_days',ascending = False)[-10:])
```

```
                                 Cluster 1 av measure  Cluster 2 av measure  \
MasksMandatoryClosedSpaces                      0.625              0.421053
IndoorOver1000                                  0.250              0.263158
ClosureOfPublicTransportPartial                 0.125              0.157895
MasksVoluntaryClosedSpaces                      0.375              0.131579
BanOnAllEventsPartial                           0.125              0.236842
StayHomeGen                                     0.250              0.421053
MassGather50Partial                             0.250              0.368421
RegionalStayHomeOrder                           0.125              0.052632
IndoorOver100                                   0.125              0.578947
MassGatherAll                                   1.000              0.947368

                                 Cluster 1 av days  Cluster 2 av days  \
MasksMandatoryClosedSpaces           24.000000         119.333333
IndoorOver1000                        7.000000          65.800000
ClosureOfPublicTransportPartial      28.000000          78.000000
MasksVoluntaryClosedSpaces           34.666667          81.500000
BanOnAllEventsPartial                23.000000          55.833333
StayHomeGen                           8.000000          40.785714
MassGather50Partial                  26.500000          57.400000
RegionalStayHomeOrder                16.000000          46.000000
IndoorOver100                        19.000000          46.100000
MassGatherAll                        53.000000          79.350000

                                 diff_measure  diff_days
MasksMandatoryClosedSpaces          -0.203947  95.333333
IndoorOver1000                       0.013158  58.800000
ClosureOfPublicTransportPartial      0.032895  50.000000
MasksVoluntaryClosedSpaces          -0.243421  46.833333
BanOnAllEventsPartial                0.111842  32.833333
StayHomeGen                          0.171053  32.785714
MassGather50Partial                  0.118421  30.900000
RegionalStayHomeOrder               -0.072368  30.000000
IndoorOver100                        0.453947  27.100000
MassGatherAll                       -0.052632  26.350000
```

```
                                Cluster 1 av measure  Cluster 2 av measure  \
ClosPubAnyPartial                              1.000              1.078947
PlaceOfWorshipPartial                          0.500              0.394737
OutdoorOver500                                 0.125              0.552632
StayHomeGenPartial                             0.125              0.131579
TeleworkingPartial                             0.375              0.263158
AdaptationOfWorkplacePartial                   0.250              0.105263
MasksVoluntaryAllSpacesPartial                 0.125              0.105263
SocialCircle                                   0.500              0.078947
StayHomeRiskGPartial                           0.125              0.026316
RegionalStayHomeOrderPartial                   0.250                   NaN

                                Cluster 1 av days  Cluster 2 av days  \
ClosPubAnyPartial                           141.0          71.875000
PlaceOfWorshipPartial                       119.0          46.750000
OutdoorOver500                              123.0          44.578947
StayHomeGenPartial                          112.0          28.750000
TeleworkingPartial                          151.0          16.000000
AdaptationOfWorkplacePartial                115.0                NaN
MasksVoluntaryAllSpacesPartial                NaN          70.666667
SocialCircle                                  NaN                NaN
StayHomeRiskGPartial                         26.0                NaN
RegionalStayHomeOrderPartial                 26.5                NaN

                                diff_measure   diff_days
ClosPubAnyPartial                   0.078947  -69.125000
PlaceOfWorshipPartial              -0.105263  -72.250000
OutdoorOver500                      0.427632  -78.421053
StayHomeGenPartial                  0.006579  -83.250000
TeleworkingPartial                 -0.111842 -135.000000
AdaptationOfWorkplacePartial       -0.144737         NaN
MasksVoluntaryAllSpacesPartial     -0.019737         NaN
SocialCircle                       -0.421053         NaN
StayHomeRiskGPartial               -0.098684         NaN
RegionalStayHomeOrderPartial             NaN         NaN
```

[120]:
```
ev_name = 'MasksMandatoryClosedSpaces'
title = 'Mandatory Masks'
dd = 30
du = 30



plt.figure(figsize = (15,4))
md = compare_measures(response,by_data = death_data, m = ev_name)
y = md.ffill().quantile(0.5,axis = 1)
y.plot()
```

```
plt.axvline(x = 0 , c = 'black')
plt.title("Deaths and "+ title)
plt.ylabel('Percent change in total deaths per million')
plt.xlabel("Days & Announcement")
plt.xticks(ticks = np.arange(-dd,du+3,3),
           labels = list(map(lambda x: str(x)+ ' days',np.arange(-dd,du+3,3))),
          rotation=45)
plt.gca().set_yticklabels(['{:.0f}%'.format(x) for x in plt.gca().get_yticks()])
#plt.legend(['10% quantile',
            #'50% quantile',
            #'90% quantile'])
plt.show()
```