SI671 Homework2

Name: Junqi Chen

Uniqname: junqich

UMID: 03846505

In [78]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

1. Load & Transform the Data

1. a)

In [79]:
```
def load_data(filename = './time_series_covid19_confirmed_global.csv'):
    covid_raw = pd.read_csv(filename, index_col=1)
    covid_raw = covid_raw.drop(columns=['Province/State', 'Lat', 'Long'])

    df = covid_raw.groupby('Country/Region').sum().T
    df.set_index(pd.to_datetime(df.index), inplace = True)

    top_5_countries = df.sort_values(by = df.index[-1], axis = 1, ascending = False).columns[0:5]
    new_case_df = df.diff()
    return new_case_df[top_5_countries].dropna()

daily_new_case_df = load_data()
daily_new_case_df
```

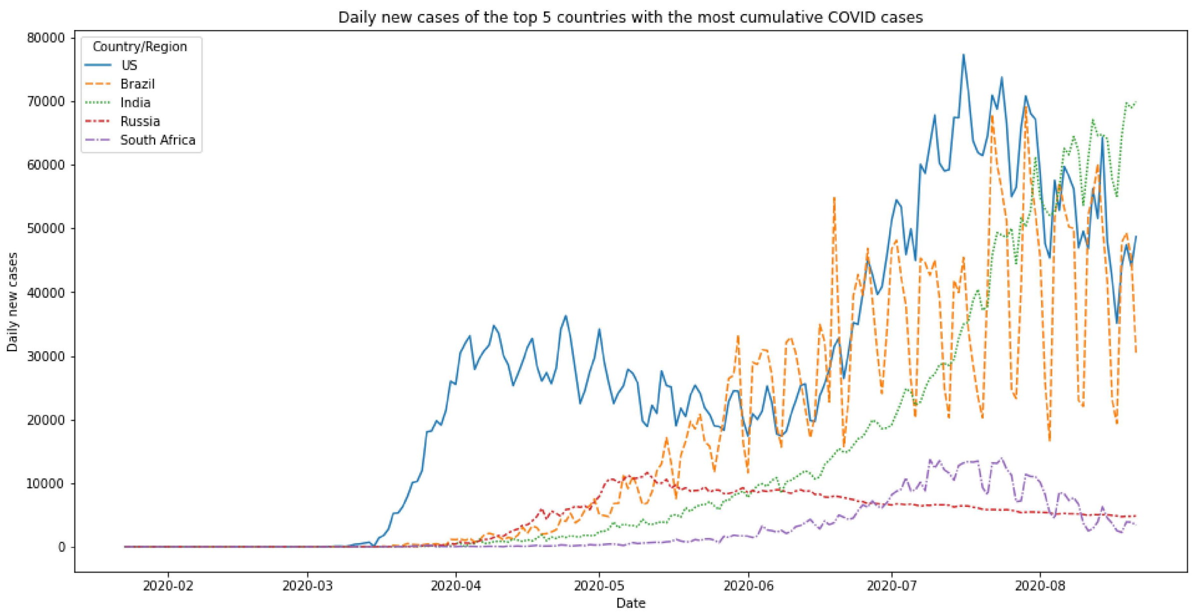
Country/Region	US	Brazil	India	Russia	South Africa
2020-01-23	0.0	0.0	0.0	0.0	0.0
2020-01-24	1.0	0.0	0.0	0.0	0.0
2020-01-25	0.0	0.0	0.0	0.0	0.0
2020-01-26	3.0	0.0	0.0	0.0	0.0
2020-01-27	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...
2020-08-17	35112.0	19373.0	55018.0	4839.0	2541.0
2020-08-18	44091.0	47784.0	64572.0	4718.0	2258.0
2020-08-19	47408.0	49298.0	69672.0	4790.0	3916.0
2020-08-20	44023.0	45323.0	68900.0	4767.0	3880.0
2020-08-21	48693.0	30355.0	69876.0	4838.0	3398.0

212 rows × 5 columns

1. b)

In [80]:

```
plt.figure(figsize=(16, 8))
ax = sns.lineplot(data=daily_new_case_df)
ax.set(
    xlabel="Date",
    ylabel="Daily new cases",
    title="Daily new cases of the top 5 countries with the most cumulative COVID cases"
)
plt.show()
```



2. Extract Seasonal Components

2. a)

In [81]:

```python
from statsmodels.tsa.seasonal import seasonal_decompose


def sea_decomp(df = daily_new_case_df):
    seasonal_components = [
        seasonal_decompose(df[country], model='additive').seasonal
        for country in df.columns
    ]
    new_df = pd.DataFrame(seasonal_components).T
    new_df.columns = df.columns
    return new_df

seasonal_df = sea_decomp()
seasonal_df
```

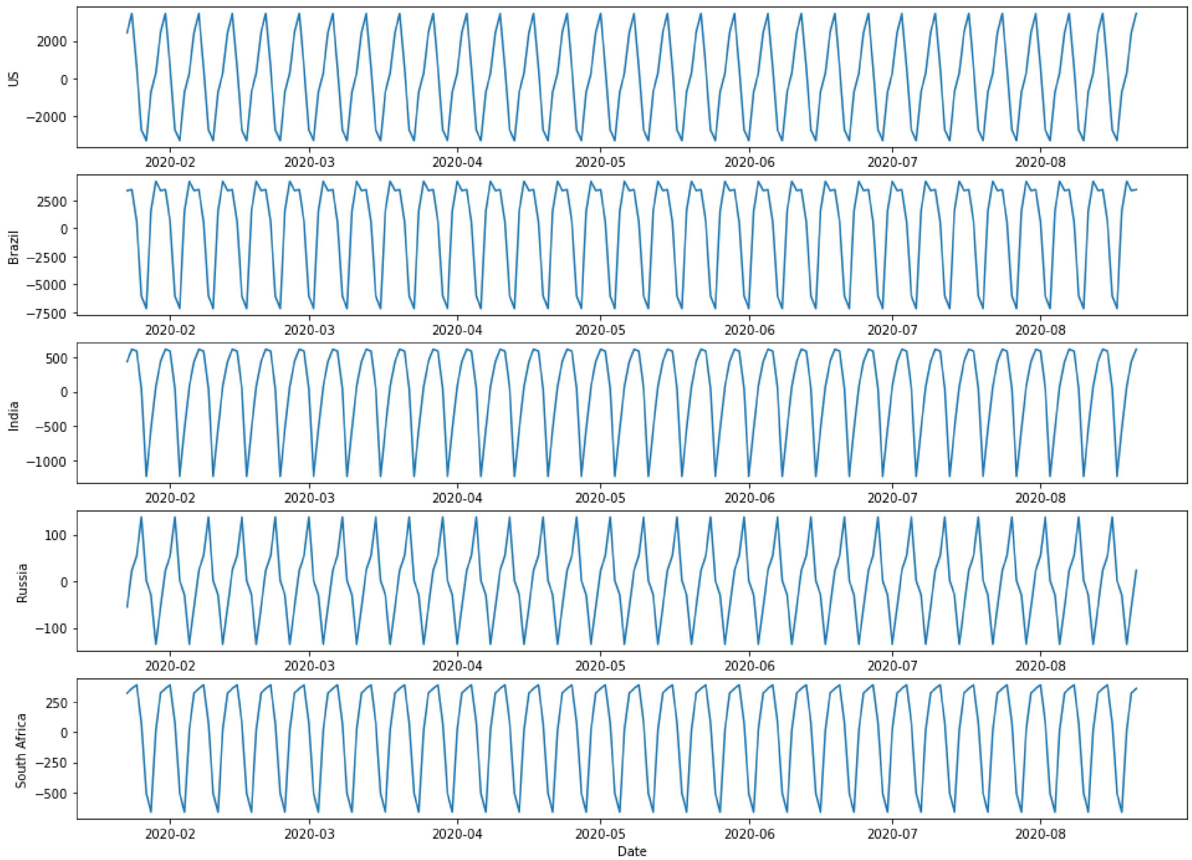
Country/Region	US	Brazil	India	Russia	South Africa
2020-01-23	2431.761670	3380.626554	441.179428	-54.886371	322.986535
2020-01-24	3446.796153	3457.641332	621.396176	23.689984	362.434811
2020-01-25	578.564626	586.665963	594.066127	55.034811	391.346141
2020-01-26	-2728.454422	-6031.950950	46.655454	137.908703	76.880131
2020-01-27	-3293.854422	-7144.674760	-1234.673118	1.842036	-507.496059
...	...	...	...	...	...
2020-08-17	-3293.854422	-7144.674760	-1234.673118	1.842036	-507.496059
2020-08-18	-719.521088	1549.577621	-544.749308	-28.929392	-662.877011
2020-08-19	284.707483	4202.114239	76.125240	-134.659770	16.725452
2020-08-20	2431.761670	3380.626554	441.179428	-54.886371	322.986535
2020-08-21	3446.796153	3457.641332	621.396176	23.689984	362.434811

212 rows × 5 columns

2. b)

In [82]:
```
fig, ax = plt.subplots(5, figsize=(16, 12))
for i, country in enumerate(seasonal_df.columns):
    sns.lineplot(data=seasonal_df, x=seasonal_df.index, y=country, ax=ax[i])

plt.xlabel("Date")
plt.show()
```



3. Time Series Similarities

3.1 Euclidean Distance

In [83]:
```
def calc_euclidean_dist(df):
    euclidean_dist = [
        [np.linalg.norm(df[i] - df[j]) for i in df.columns]
        for j in df.columns
    ]
    new_df = pd.DataFrame(euclidean_dist, index=list(df.columns), columns=list(df.columns))
    return new_df
```

3.1. a)

In [84]:
```
calc_euclidean_dist(daily_new_case_df)
```

	US	Brazil	India	Russia	South Africa
US	0.000000	233760.757213	272344.138927	433638.331574	436238.175972
Brazil	233760.757213	0.000000	178779.663740	306032.283923	304919.698741
India	272344.138927	178779.663740	0.000000	316862.767630	303936.538967
Russia	433638.331574	306032.283923	316862.767630	0.000000	67392.593681
South Africa	436238.175972	304919.698741	303936.538967	67392.593681	0.000000

3.1. b)

In [85]:
```
calc_euclidean_dist(seasonal_df)
```

	US	Brazil	India	Russia	South Africa
US	0.000000	37616.752035	27032.887714	33986.305519	30084.544171
Brazil	37616.752035	0.000000	57583.437987	63663.896821	60839.376478
India	27032.887714	57583.437987	0.000000	9102.412727	4490.020448
Russia	33986.305519	63663.896821	9102.412727	0.000000	5658.222387
South Africa	30084.544171	60839.376478	4490.020448	5658.222387	0.000000

3.2 Cosine Similarity

In [86]:
```
def calc_cos_sim(df):
    def cos_sim(a, b):
        return np.dot(a, b) / (np.linalg.norm(a) * np.linalg.norm(b))

    cos_dist = [
        [cos_sim(df[i], df[j]) for i in df.columns]
        for j in df.columns
    ]
    new_df = pd.DataFrame(cos_dist, index=list(df.columns), columns=list(df.columns))
    return new_df
```

3.2. a)

In [87]:
```python
calc_cos_sim(daily_new_case_df)
```

	US	Brazil	India	Russia	South Africa
US	1.000000	0.898664	0.847160	0.804740	0.884909
Brazil	0.898664	1.000000	0.878452	0.763523	0.871214
India	0.847160	0.878452	1.000000	0.590388	0.809944
Russia	0.804740	0.763523	0.590388	1.000000	0.638246
South Africa	0.884909	0.871214	0.809944	0.638246	1.000000

3.2. b)

In [88]:
```python
calc_cos_sim(seasonal_df)
```

	US	Brazil	India	Russia	South Africa
US	1.000000	0.868859	0.783851	-0.325065	0.664261
Brazil	0.868859	1.000000	0.632741	-0.629987	0.403198
India	0.783851	0.632741	1.000000	0.092292	0.917529
Russia	-0.325065	-0.629987	0.092292	1.000000	0.174437
South Africa	0.664261	0.403198	0.917529	0.174437	1.000000

4. Dynamic Time Warping (DTW) Cost

4.1 Define a Function to Calculate DTW Cost

In [126]:
```python
def calc_pairwise_dtw_cost(A, B, ret_matrix = True):
    def dist(a, b):
        return (a - b) ** 2

    ret_mat = np.zeros((len(A), len(B)))
    for i in range(len(A)):
        for j in range(len(B)):
            if i == 0:
                ret_mat[i][j] = dist(A[i], B[j]) + ret_mat[i][j-1]
            elif j == 0:
                ret_mat[i][j] = dist(A[i], B[j]) + ret_mat[i-1][j]
            else:
                ret_mat[i][j] = dist(A[i], B[j]) + min(ret_mat[i][j-1], ret_mat[i-1][j], ret_mat[i-1][j-1])

    if ret_matrix:
        return ret_mat
    else:
        return ret_mat[len(A)-1][len(B)-1]
```

In [127]:
```
calc_pairwise_dtw_cost([0,1,1,2,3,4,3,2,1,1], [1,2,3,4,3,2,1,1,1,2])
```

```
array([[ 1.,  5., 14., 30., 39., 43., 44., 45., 46., 50.],
       [ 1.,  2.,  6., 15., 19., 20., 20., 20., 20., 21.],
       [ 1.,  2.,  6., 15., 19., 20., 20., 20., 20., 21.],
       [ 2.,  1.,  2.,  6.,  7.,  7.,  8.,  9., 10., 10.],
       [ 6.,  2.,  1.,  2.,  2.,  3.,  7., 11., 13., 11.],
       [15.,  6.,  2.,  1.,  2.,  6., 12., 16., 20., 15.],
       [19.,  7.,  2.,  2.,  1.,  2.,  6., 10., 14., 15.],
       [20.,  7.,  3.,  6.,  2.,  1.,  2.,  3.,  4.,  4.],
       [20.,  8.,  7., 12.,  6.,  2.,  1.,  1.,  1.,  2.],
       [20.,  9., 11., 16., 10.,  3.,  1.,  1.,  1.,  2.]])
```

4.2 Compute Pairwise DTW Cost

In [128]:
```
def calc_dtw_cost(df):
    dtw_dist = [
        [calc_pairwise_dtw_cost(df[i], df[j], ret_matrix=False) for i in df.columns]
        for j in df.columns
    ]
    new_df = pd.DataFrame(dtw_dist, index=list(df.columns), columns=list(df.columns))
    return new_df
```

4.2. a)

In [129]:
```
calc_dtw_cost(daily_new_case_df)
```

	US	Brazil	India	Russia	South Africa
US	0.000000e+00	9.575974e+09	5.187397e+09	1.740747e+11	1.395159e+11
Brazil	9.575974e+09	0.000000e+00	1.430988e+10	8.361811e+10	6.542703e+10
India	5.187397e+09	1.430988e+10	0.000000e+00	9.927626e+10	8.728950e+10
Russia	1.740747e+11	8.361811e+10	9.927626e+10	0.000000e+00	1.638671e+08
South Africa	1.395159e+11	6.542703e+10	8.728950e+10	1.638671e+08	0.000000e+00

4.2. b)

In [130]:
```
calc_dtw_cost(seasonal_df).apply(np.sqrt)
```

	US	Brazil	India	Russia	South Africa
US	0.000000	31878.178988	23565.948799	32327.414867	28016.515162
Brazil	31878.178988	0.000000	53400.789074	61868.013722	57143.198022
India	23565.948799	53400.789074	0.000000	7687.627537	4463.425362
Russia	32327.414867	61868.013722	7687.627537	0.000000	4259.096369
South Africa	28016.515162	57143.198022	4463.425362	4259.096369	0.000000

Answer:

1. According to the DTW similarity matrix above, we can find out that Russia and South Africa has the most similar seasonal pattern. We can also find out that (definitely!) each countries has the exactly the same seasonal parttern as itself. We can also compare the similarity between different countries with the distance metric (including DTW, cos and euclidean).
2. Comparing to the Euclidean Distance and Cosine Similarity, DTW can depict more inner relationship (with time shifting) in between. For example, we can find out that the values of DTW distance (after taking the square root) are smaller than the corresponding Euclidean distance,

indicating that it depicts more similarity by shifting the time series. So I do not think they are telling the same story. DTW distance can be a more advanced measurement for similarity on time series.

(The similarity matrix of Euclidean Distance and Cosine Similarity are attached below)

In [93]:
```
display(calc_euclidean_dist(seasonal_df))
display(calc_cos_sim(seasonal_df))
```

	US	Brazil	India	Russia	South Africa
US	0.000000	37616.752035	27032.887714	33986.305519	30084.544171
Brazil	37616.752035	0.000000	57583.437987	63663.896821	60839.376478
India	27032.887714	57583.437987	0.000000	9102.412727	4490.020448
Russia	33986.305519	63663.896821	9102.412727	0.000000	5658.222387
South Africa	30084.544171	60839.376478	4490.020448	5658.222387	0.000000

	US	Brazil	India	Russia	South Africa
US	1.000000	0.868859	0.783851	-0.325065	0.664261
Brazil	0.868859	1.000000	0.632741	-0.629987	0.403198
India	0.783851	0.632741	1.000000	0.092292	0.917529
Russia	-0.325065	-0.629987	0.092292	1.000000	0.174437
South Africa	0.664261	0.403198	0.917529	0.174437	1.000000

In [ ]: