CS732/DS732: Data Visualization – Datathon 04

Kunika Valecha

October 14, 2020

Contents

1 Problem Statement

To study different types of Matrix seriation techniques on COVID-19 data set.

2 Data Description

The data source is COVID-19 dataset.

- **COVID-19 Metadata**: The tabular datasets published by the World Health Organization to create new networks (similarity networks, correlation networks, etc.) and visualize network communities.
- The data is in tabular format and is contained inside multiple csv files in multiple formats.
- One of the formats is the time-series format in which the number of cases(deaths, recovered, and confirmed) are being recorded day-wise and stored in accordance with there positions in world coordinates.
- The time-series data for United states particularly is also available.
- The data has been recorded from 22nd January, 2020 to 23rd September,2020.
- Three categories of cases are recorded in three separate files viz.
  - time_series_covid_19_deaths.csv,
  - time_series_covid_19_recovered.csv, and
  - time_series_covid_19_confirmed.csv.

3 Data Reading and Cleaning

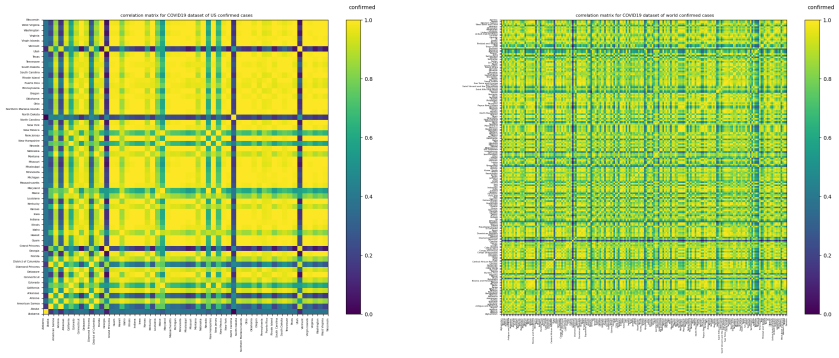
- Since the dates of data are represented by columns and rows represent the States and their respective countries worldwide, so the data has been transposed in order to make an adjacency matrix on relations of countries.
- For some of the observations the information of the particular state is absent, leaving a NaN value in the data frame, this is why the countries are considered to be a better choice in order to represent the labels on the nodes.
- As the the rows of the dataset are unique from each other due to State identity, but as we consider the countries for the basis of our labelling then there might occur repetitions as several state can have a common country. In order to avoid this repetitions, we have taken the sum of all the values belonging to a particular country on a specific date.

4 Matrix Seriation

Matrix Seriation is the process of reordering matrix elements to find patterns. This approach can be used to reorganize rows or columns of a dataset so as to enumerate them in an appropriate order. An enumeration of rows or columns that best expresses resemblance relationships between the elements is sought.

4.1 Adjacency Matrix

An adjacency matrix is a square matrix used to represent a finite graph. The elements of the matrix indicate whether pairs of vertices are adjacent or not in the graph. Here we have an unweighted and undirected graph where the adjacency matrix is a zero diagonal symmetric matrix with zeros representing no edge and ones representing the edge. The adjacency matrix before seriation looks like below, in fig. 1.



(a) US (b) World

Figure 1: Fig.(a) and (b) showing Adjacency matrix for confirmed cases in US and World respectively

4.2 Seriating Adjacency Matrix

There are several ways in which a matrix can be seriated. These ways are dependent on the type of clustering applied to the matrix. A library named **scipy.cluster.hierarchy.linkage** is used to perform hierarchical/agglomerative clustering.

Input to linkage function is a condensed distance matrix. A condensed distance matrix is a flat array containing the upper triangular of the distance matrix. This is the form that pdist from scipy returns. Alternatively, a collection of m observation vectors in n dimensions may be passed as an m by n array. All elements of the condensed distance matrix must be finite, i.e., no NaNs or infs.

As said, The input y may be either a 1-D condensed distance matrix or a 2-D array of observation vectors. If y is a 1-D condensed distance matrix, then y must be a sized vector, where n is the number of original observations paired in the distance matrix. The behavior of this function is very similar to the MATLAB linkage function.

The following linkage methods are used to compute the distance between two clusters and . The algorithm begins with a forest of clusters that have yet to be used in the hierarchy being formed. When two clusters s and t from this forest are combined into a single cluster u, s and t are removed from the forest, and u is added to the forest. When only one cluster remains in the forest, the algorithm stops, and this cluster becomes the root.

A distance matrix is maintained at each iteration. The d[i,j] entry corresponds to the distance between cluster i and j in the original forest.

At each iteration, the algorithm must update the distance matrix to reflect the distance of the newly formed cluster u with the remaining clusters in the forest.

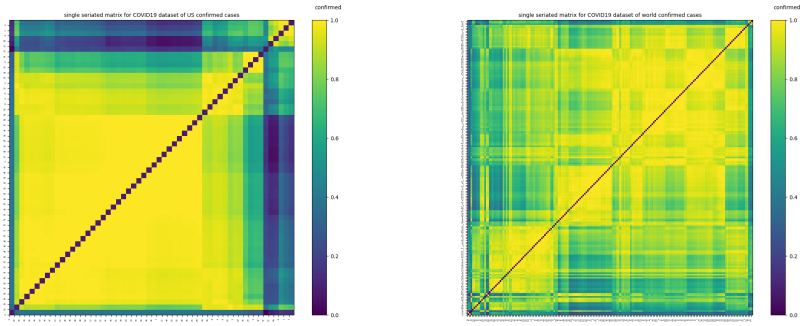
The following are methods for calculating the distance between the newly formed cluster u and each v.

4.2.1 Types of Matrix Seriation

- method='single' assigns

  d(u,v) = min(dist(u[i, v[j])

  for all points i in cluster u and j in cluster v. This is also known as the Nearest Point Algorithm.



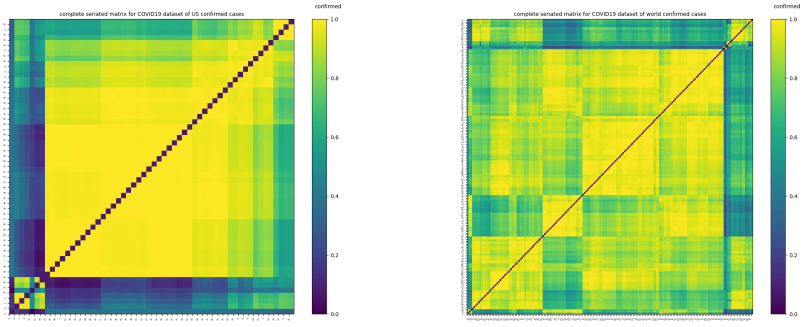
(a) US (b) World

Figure 2: Fig.(a) and (b) showing ordered Adjacency matrix in single linkage clustering for confirmed cases in US and World respectively

- method='complete' assigns

d(u,v) = max(dist(u[i, v[j])

for all points i in cluster u and j in cluster v. This is also known by the Farthest Point Algorithm or Voor Hees Algorithm.



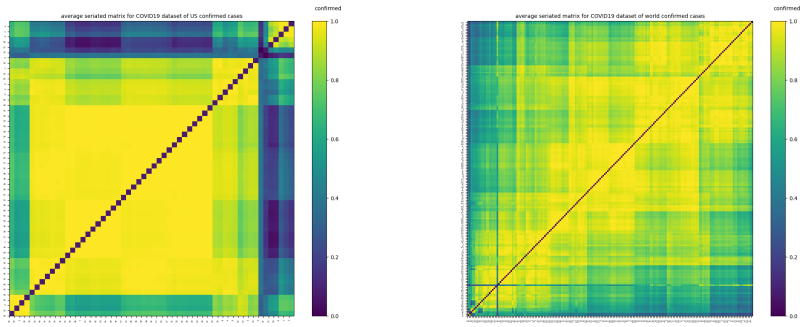
(a) US (b) World

Figure 3: Fig.(a) and (b) showing ordered Adjacency matrix in complete linkage of clustering for confirmed cases in US and World respectively

- method='average' assigns

  d(u,v)=$\sum_{ij} \frac{d(u[i],v[j])}{(|u|*|v|)}$

  for all points i and j where |u| and |v| are the cardinalities of clusters u and v respectively. This is also called the UPGMA algorithm.



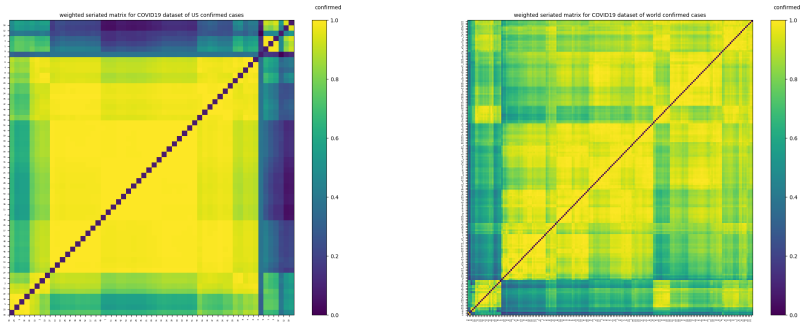
(a) US (b) World

Figure 4: Fig.(a) and (b) showing Ordered Adjacency matrix in average linkage clustering for confirmed cases in US and World respectively

- method='weighted' assigns

  $\mathbf{dist(u,\ v) = (dist(s,\ v) + dist(t,\ v))/2}$

  where cluster u was formed with cluster s and t and v is a remaining cluster in the forest (also called WPGMA).



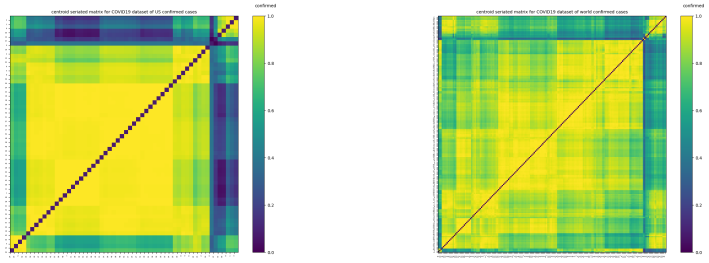
(a) US (b) World

Figure 5: Fig.(a) and (b) showing ordered Adjacency matrix in weighted linkage clustering for confirmed cases in US and World respectively

- method='centroid' assigns

  $\mathbf{dist(s,\ t) = \|c_s - c_t\|_2}$

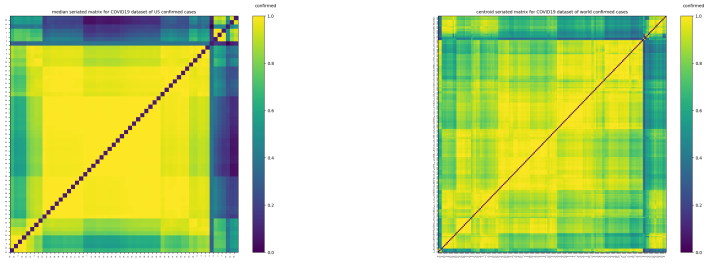
  where c_s and c_t are the centroids of clusters s and t, respectively. When two clusters s and t are combined into a new cluster u, the new centroid is computed over all the original objects in clusters s and t. The distance then becomes the Euclidean distance between the centroid of and the centroid of a remaining cluster in the forest. This is also known as the UPGMC algorithm.



(a) US (b) World

Figure 6: Fig.(a) and (b) showing ordered Adjacency matrix in centroid linkage clustering for confirmed cases in US and World respectively

- method='median' assigns d(s,t) like the centroid method. When two clusters s and t are combined into a new cluster u, the average of centroids s and t give the new centroid u. This is also known as the WPGMC algorithm.



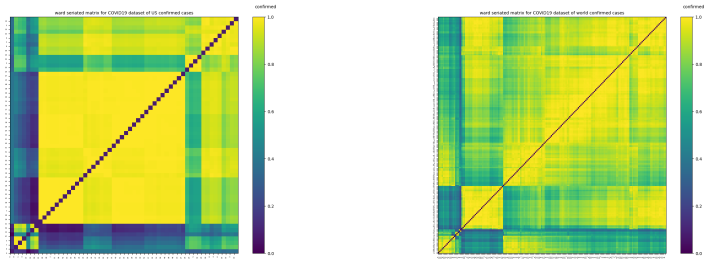
(a) US (b) World

Figure 7: Fig.(a) and (b) showing ordered Adjacency matrix in median linkage of clustering for confirmed cases in US and World respectively

- method='ward' uses the Ward variance minimization algorithm. The new entry is computed as follows,

  $$\mathbf{d(u,v)} = \sqrt{\frac{|v|+|s|}{T}d(v,s)^2 + \frac{|v|+|t|}{T}d(v,t)^2 - \frac{|v|}{T}d(s,t)^2}$$

  where u is the newly joined cluster consisting of clusters s and t, v is an unused cluster in the forest, T = |v| + |s| + |t| , and * is the cardinality of its argument. This is also known as the incremental algorithm.



(a) US (b) World

Figure 8: Fig.(a) and (b) showing ordered Adjacency matrix in ward linkage of clustering for confirmed cases in US and World respectively

5 Best suited Seriation method

The best suited for the world COVID-19 data set are ***complete*** and ward layout as they depicts the clusters properly which other seriation techniques fail to portray. However, the US dataset being lesser dense can be analysed through other methods as well.

6 Technologies Used

- fastcluster library python
- matplotlib python