



# Data Analytics and Visualization 21/22

## Assignment

## Graph Analysis of Air Transport Network

Claire DELGOVE – S359263

*Number of words  $\approx$  2500 without problem statement, data description, titles, and code.*

# Contents

1.	<b>CONTEXT .....</b>	<b>3</b>
2.	<b>DATA PRE-ANALYSIS .....</b>	<b>4</b>
1.	GEODATA DATASET .....	4
2.	FLIGHTDATA DATASET .....	5
3.	<b>TRANSFORMATION OF THE DATA INTO NETWORK FORM .....</b>	<b>6</b>
1.	GRAPH VISUALIZATION .....	6
2.	GEOPLOT VISUALIZATION .....	8
4.	<b>MULTI-SCALE ANALYSIS .....</b>	<b>9</b>
1.	WEIGHTED DEGREE DISTRIBUTION .....	9
2.	DEGREE VS BETWEENNESS DISTRIBUTION .....	11
3.	ASSORTATIVITY (DEGREE CORRELATION) .....	13
4.	CORE COMMUNITY SIZE .....	15
5.	<b>IMPACTS OF FUTURE AIRCRAFTS .....</b>	<b>17</b>
6.	<b>EXTRA-WORK .....</b>	<b>18</b>
1.	WEIGHTED DEGREE CUMULATIVE DISTRIBUTION .....	18
2.	NORMALIZED BETWEENNESS VS NORMALIZED DEGREE .....	19
3.	COMMUNITIES' DETECTION .....	20
7.	<b>CONCLUSION .....</b>	<b>23</b>
8.	<b>CONTENTS OF FIGURES .....</b>	<b>24</b>
9.	<b>BIBLIOGRAPHY .....</b>	<b>25</b>
10.	<b>CODE .....</b>	<b>26</b>

# 1 Context

Nowadays, the air transport is predominant in the ways of life. Due to the globalization and the fourth industrial revolution, people met the need to travel more and more for working, but also for their well-being. These societies' changes led to increase interactions.

The purpose of this project is to lead an air transport network data analysis across different scales for four countries: the United States, China, the United Kingdom and Australia. For this, different feature-based graph analysis methods have been carefully selected to complete the airline network study in these countries on a macro-scale, a meso-scale, and node-level. To enable effective visualization regarding the computation time, the representation will be done in a low data dimension, selecting data for one chosen month of flight data across the countries.

The data used will be analysed as a whole, before demonstrating multi-scale graph analysis for the four given countries. At the end of each method, the selection and the application of the metric used will be analysed with performance considerations. Thereafter, the potential impact of results on the design of future aircrafts in the different countries will be discussed. Finally, the extra-work of this project will be briefly presented. All the codes will be done using MATLAB.

# 2 Data Pre-Analysis

## 2.1 GeoData Dataset

**GeoData** (Figure 1) is the dataset contains information about the geographical positions in latitude and longitude for all airports (whose id, name and country are specified). Since it is just needed to know the positions for the studied countries which are the USA (Figure 2), the UK, China and Australia, extractions from this dataset have been performed per country in order to search for the positions more efficiently.

	1	2	3	4	5
	id	label	country	Lat	Lon
1	'0A7'	'Henderso...	'United Sta...	35.3167	-82.4500
2	'AAA'	'Anaa'	'French Pol...	-17.4167	-145.5000
3	'AAB'	'Arrabury'	'Australia'	-26.7500	141
4	'AAC'	'Al Arish In...	'Egypt'	31.0733	33.8358
5	'AAD'	'Ad-Dabbah'	'Sudan'	17.5928	33.9592
6	'AAE'	'Annaba'	'Algeria'	36.9200	7.7600
7	'AAG'	'Arapoti'	'Brazil'	-24.1667	-49.6667
8	'AAH'	'Aachen/M...	'Germany'	50.8231	6.1861
9	'AAI'	'Arraias'	'Brazil'	-12.9167	-46.9333
10	'AAJ'	'Cayana Air...	'Suriname'	3.8967	-55.5792
11	'AAK'	'Aranuka'	'Kiribati'	0.2167	174

Figure 1: GeoData Dataset

	1	2	3	4	5
	id	label	country	Lat	Lon
1	'0A7'	'Henderso...	'United Sta...	35.3167	-82.4500
2	'AAP'	'Andrau Air...	'United Sta...	29.7239	-95.5883
3	'ABE'	'Allentown/...	'United Sta...	40.6522	-75.4403
4	'ABI'	'Abilene'	'United Sta...	32.4167	-99.6833
5	'ABL'	'Ambler'	'United Sta...	67.0865	-157.8514
6	'ABQ'	'Albuquerq...	'United Sta...	35.1050	-106.6413
7	'ABR'	'Aberdeen ...	'United Sta...	45.4500	-98.4333
8	'ABY'	'Albany (U...	'United Sta...	31.5333	-84.2000
9	'ACB'	'Antrim Co...	'United Sta...	44.9833	-85.2028
10	'ACK'	'Nantucket'	'United Sta...	41.2667	-70.0667
11	'ACT'	'Waco'	'United Sta...	31.6167	-97.2333

Figure 2: GeoData Dataset extraction for the USA

This dataset shows there are 609 listed airports in the USA, 242 in China, 135 in the UK and 1974 in the USA (Figure 3).

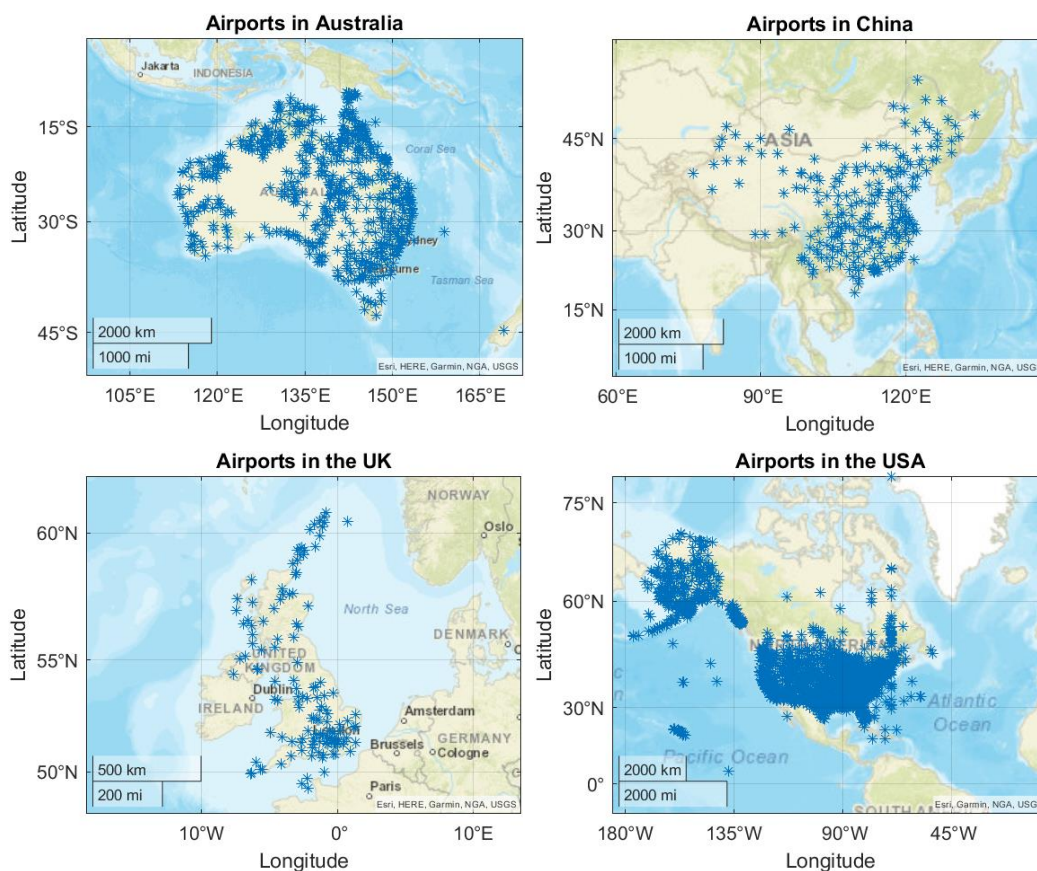


Figure 3: Airports' visualization for each country

## 2.2 FlightData Dataset

**FlightData** (Figure 4) is the dataset indicated the flights routes from a source airport (id, city and country specified), and the weight associated with each line. This weighted quantifies the passenger volume.

In the dataset, two sheets are provided. For this project, only the first sheet with data from 2003 and 2009 will be studied. More, a time series is indicated corresponding when a line has been listed. For simplification, only one month data across the fourth different countries for one month will be used for the following analysis. The arbitrary date of the 01 July 2009 is chosen up to this point.

In the same way, since only the lines with sources and targets stucked in the same country are interested in this project, extractions from this dataset will be performed per country.

	1	2	3	4	5	6	7	8
	Source	SourceCity	SourceCountry	Target	TargetCity	TargetCountry	Weight	TimeSeries
1	'FNC'	'Funchal'	'Portugal'	'PXO'	'Porto Santo'	'Portugal'	9864	01-Jul-2003
2	'PXO'	'Porto Santo'	'Portugal'	'FNC'	'Funchal'	'Portugal'	9864	01-Jul-2003
3	'AEP'	'Buenos Aires'	'Argentina'	'MVD'	'Montevideo'	'Uruguay'	1463	01-Jul-2003
4	'MVD'	'Montevideo'	'Uruguay'	'AEP'	'Buenos Air...	'Argentina'	1463	01-Jul-2003
5	'AEP'	'Buenos Aires'	'Argentina'	'ROS'	'Rosario (AR)'	'Argentina'	2261	01-Jul-2003
6	'ROS'	'Rosario (AR)'	'Argentina'	'AEP'	'Buenos Air...	'Argentina'	2337	01-Jul-2003
7	'AEP'	'Buenos Aires'	'Argentina'	'PDP'	'Punta del E...	'Uruguay'	2413	01-Jul-2003
8	'PDP'	'Punta del E...	'Uruguay'	'AEP'	'Buenos Air...	'Argentina'	2337	01-Jul-2003
9	'SFN'	'Santa Fe (A...	'Argentina'	'ROS'	'Rosario (AR)'	'Argentina'	76	01-Jul-2003

Figure 4: FlightData Dataset

# 3 Transformation of the data into network form

## 3.1 Graph Visualization

To construct an adequate network, airports will be represented by nodes, and flight routes by weighted edges. Before visualizing the flights routes with edges on a graph, it is necessary to clean the whole data. Indeed, flights routes whose source or target airports don't have their geographical positions listed in **GeoData** are deleted. Regarding the edges (airline routes), they are all kept since they represent the different routes for different airlines with weights associated. Several weights are sometimes present for the same route due to the multiple airlines. To simplify, it could have been possible to sum the weights to represent the whole passenger flow for each route. But for authenticity, it has been decided to keep them all, and visualize the important routes in the future by lines superposition.

Thereafter, the datasets contain the following numbers of routes (edges) and airports (nodes). The returns routes are counting and hold the same weight as their accurate single route:

	United States	China	United Kingdom	Australia
Number of Nodes	642	150	64	119
Number of Edges	12556	4695	698	894

On *Figure 5*, directed graphs are plotted to visualize the different edges organizations for the four countries. Weights for each edge aren't indicated to avoid overloaded graphs.

It is noticeable that airports are quite all connected with each other for every country, excepted for some in the UK and in the USA. Those which are not connected to the core graph are airports located in countries far from major urban centres which aren't well served by transports, such as DUT (*Dutch Harbor*) in Alaska, EEA (*Eagle*) in Colorado, or KKI (*Kentland*) in Indiana.

On the zoom square for China, it is observable that graphs are multi-edge graphs. Indeed, there are several loops from the same source and target.

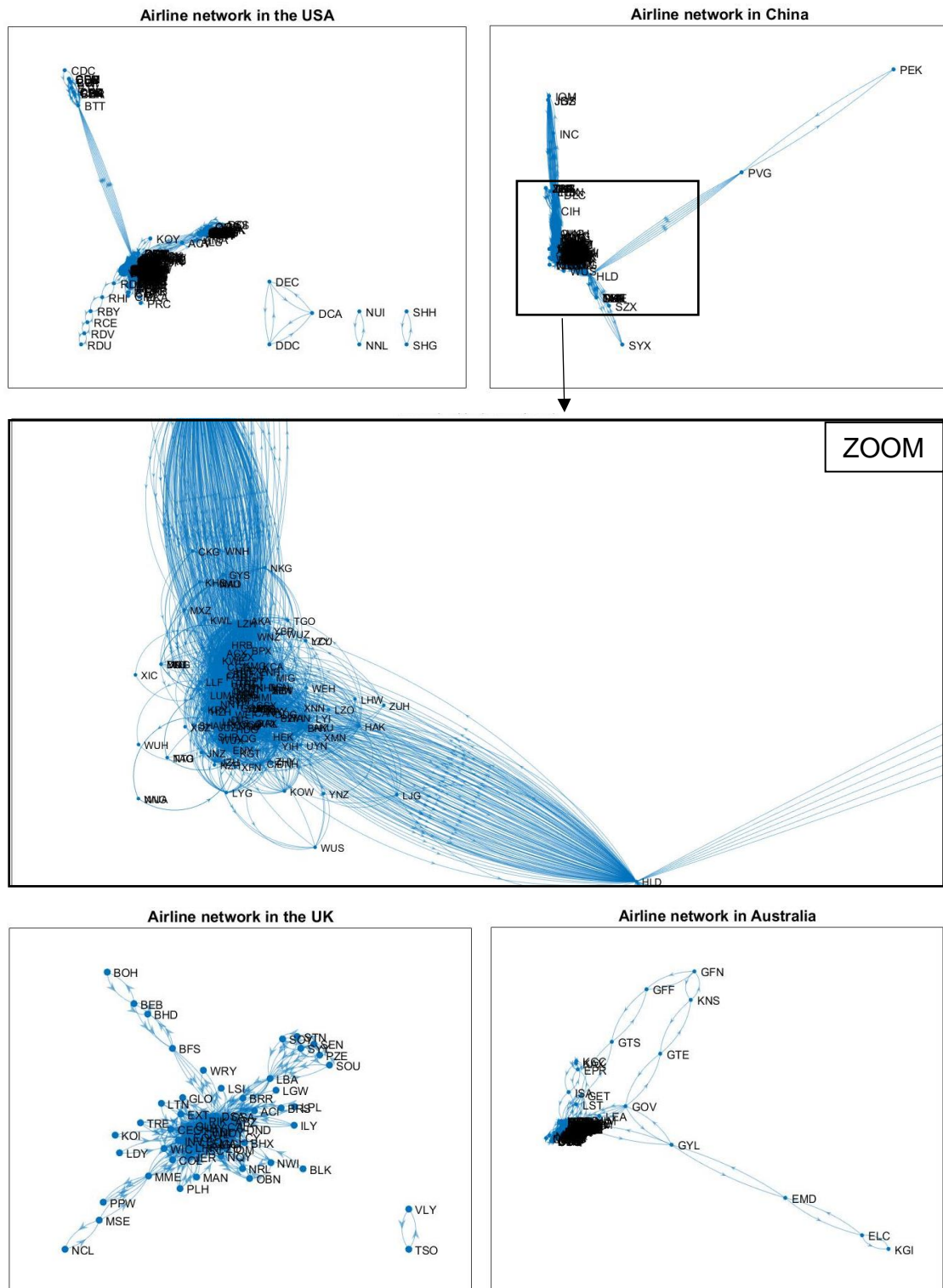


Figure 5: Airline network visualization for each country



## 3.2 GeoPlot Visualization

To improve the visualization, same graphs are plotted using *GeoPlot* on MATLAB on a geographic plan collecting the needed positions listed in **GeoData** (Figure 6). Only airports involved in internal flights are considered for each network.

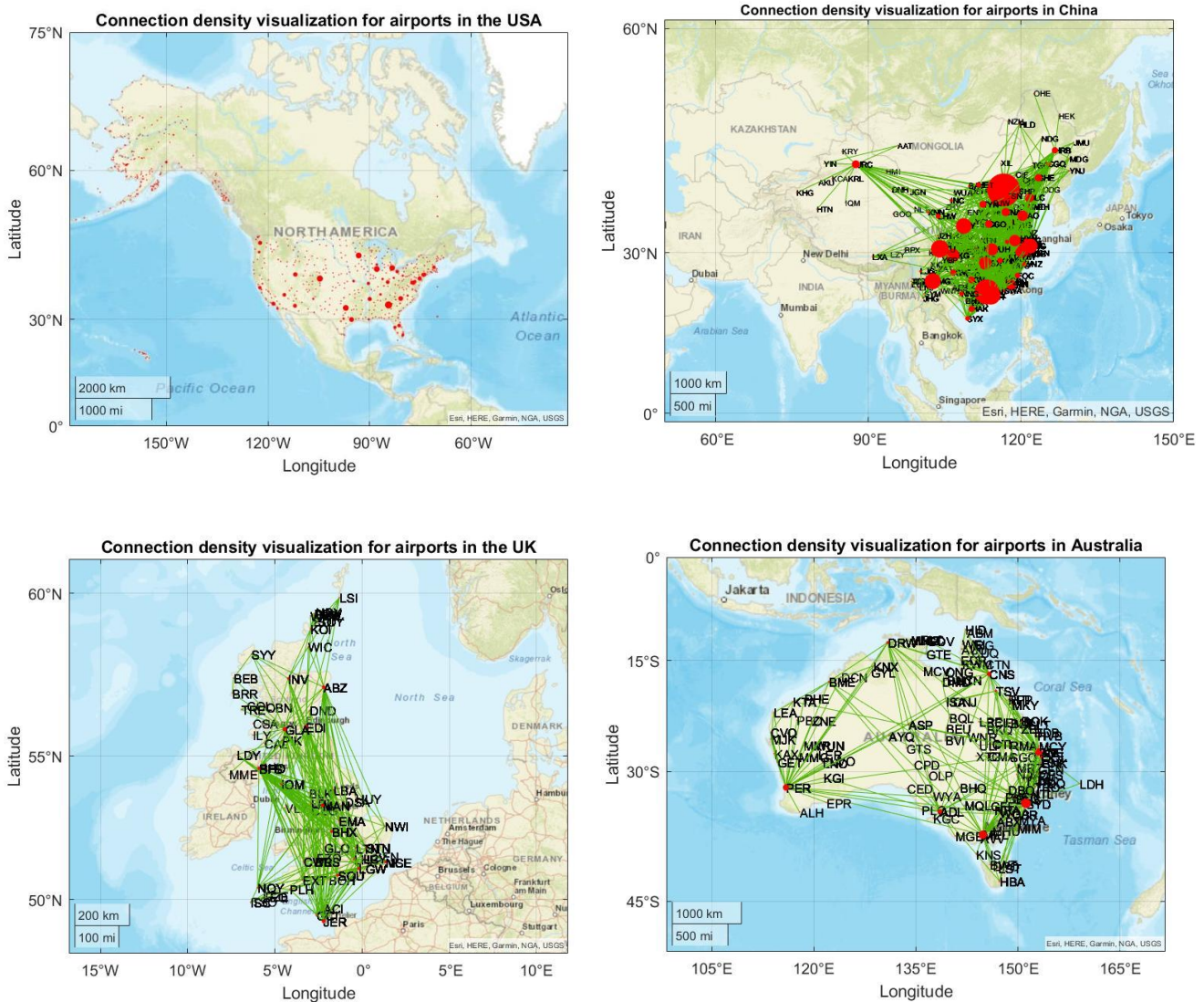


Figure 6: Airline network visualization for each country by GeoPlot

Each airport is represented by a red point as big as its degree (number of connections it has with other airports). The flight routes are represented in green. Since all lines have been kept, it is observable the importance of a route regarding its weight (passenger volume) by the superposition of lines. Lines for the USA are not plotted due to an important computation time.

In the USA, big airports are located a little to the east, such as China, because there is more activity in these areas. In the UK, they are quite evenly distributed. For Australia, they are essentially located near to the coasts, which makes sense since the centre of Australia is only a huge desert.



# 4 Multi-scale analysis

## 4.1 Weighted degree distribution

The degree distribution helps to study the connections' infrastructure of the whole network. To plot the distribution, the weighted degree for each node is computed. It is the number of edges (in-edges and out-edges for a directed graph) for a node, weighted by the weight of each edge. In other words, it is the sum of the weight of the edges.

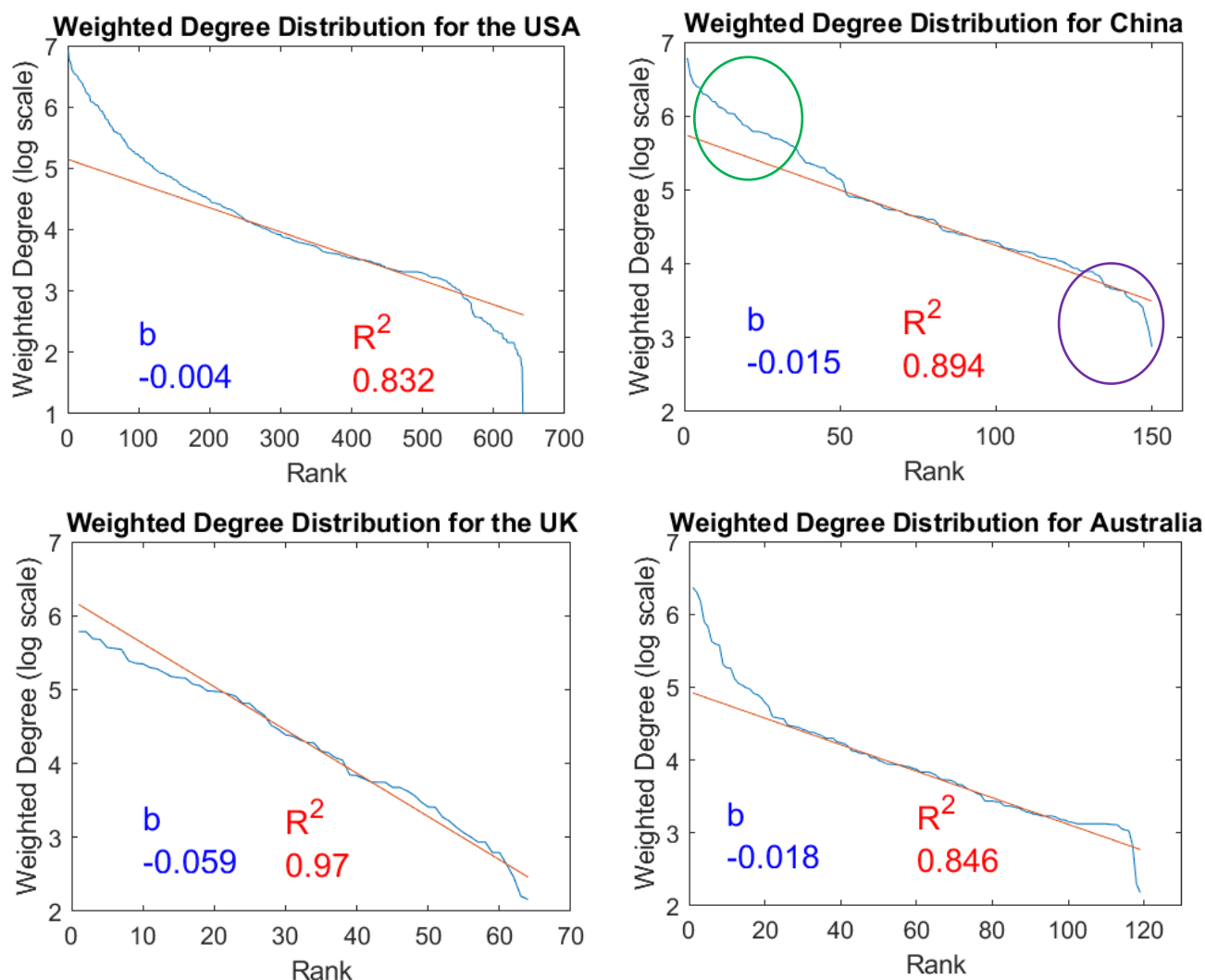


Figure 7: Weighted Degree Distribution for each country



CAT 1: Airports with a higher number of connections than expected (major airports)



CAT 2: Airports with a lower number of connections than expected (private airports)

It is observable (Figure 7) that the first ranked cities have a degree higher than expected. It can be said that there are over-connected. At the opposite, the last ranked cities have a lower degree than expected. These exceptions don't follow the linear trend of slope  $b$ . These over-levels are qualified as the King Pauper effect. It is observable that influential international airports formed connections with a higher magnitude than the power of law, compared to less important airports, which formed connections poorly connected. The power law for degree might suggest that countries with these characteristics have several very large airports with loads of connections (making them hubs), in contrast to the majority of smaller airports with very few connections.

More important airports with a high flow of passengers are also those with a high population. Indeed, passenger flows and population are correlated.

Meanwhile, the determination coefficient measures how much airports follow the linear trend. For the UK, the  $R^2$  of 0.97 is high and quite close to 1. Then, a few airports are over or under degrees expected, but the majority follows the linear trend. It is not the same situation in the USA where they are many over-levelled airports since the  $R^2$  is the less near to 1.

Generally, this method is adequate to visualize the trend of the whole network, and how airports prioritize their connections. This is perfectly applicable for these networks since they are large and complex. However, the weighted degree distribution isn't sufficient to fully analyse the network since it gives only an overview, without focusing on the attributes of airports.

## 4.2 Degree VS Betweenness distribution

Where the degree centrality is a measure of how many connections a node has, the betweenness centrality measures the number of shortest paths passing through a node. In this study of the airline's networks, the betweenness refers to the number of random traveller pathways through an airport. Consequently, it measures the importance of an airport regarding its connections with others.

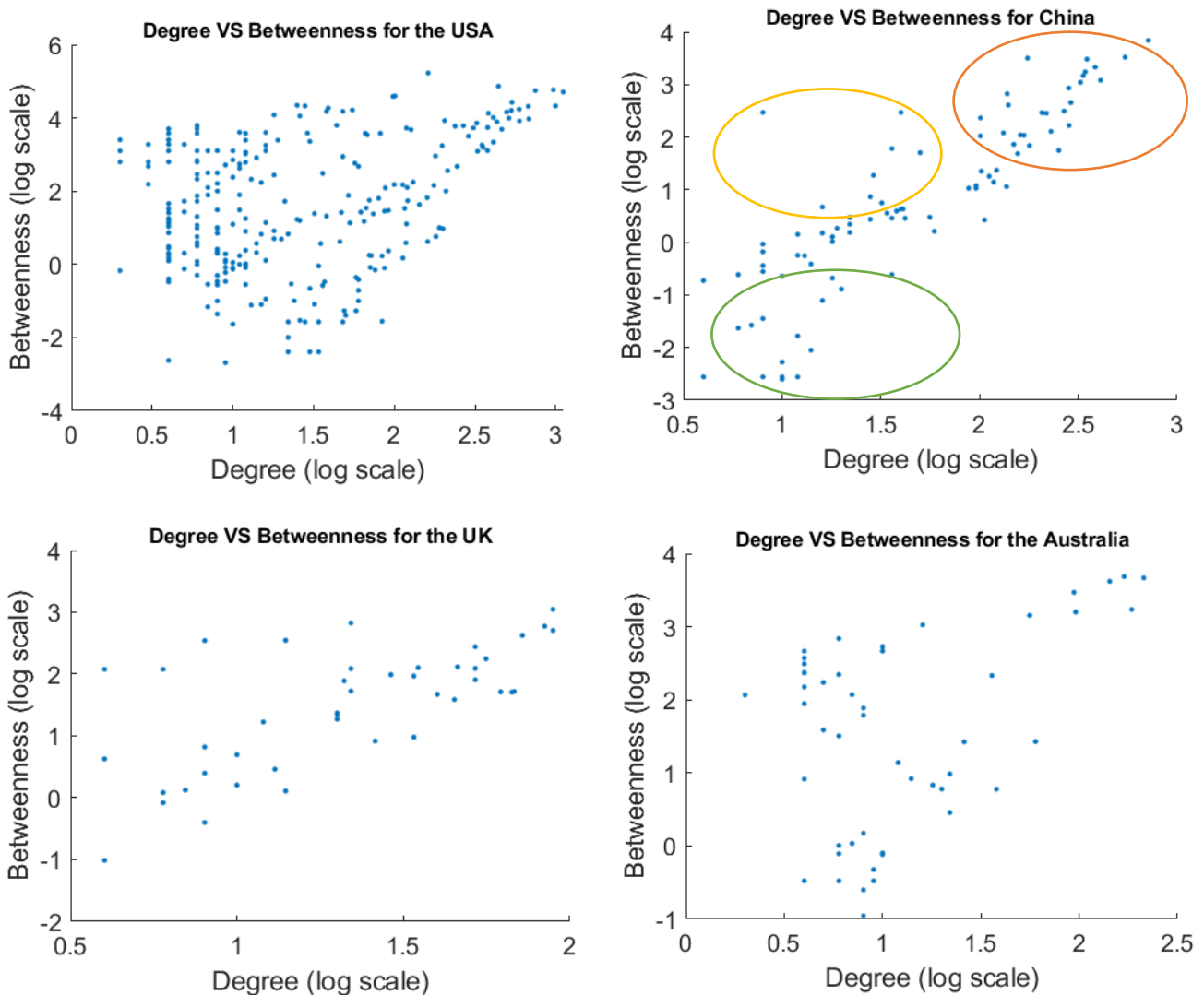





Figure 8: Degree VS Betweenness graph for each country

-  CAT 1: Well-connected airports with a high betweenness (major points)
-  CAT 2: Not well-connected airports with a low betweenness (local points)
-  CAT 3: Not well-connected airports but with a high betweenness (intermediary points)

The graphs (Figure 8) show a correlation between degree and betweenness. Indeed, the betweenness increases as degree increases. This means highly connected airports (with a high degree) are also the ones with a high betweenness as well. In other words, they are the shortest paths for the routes with stops.

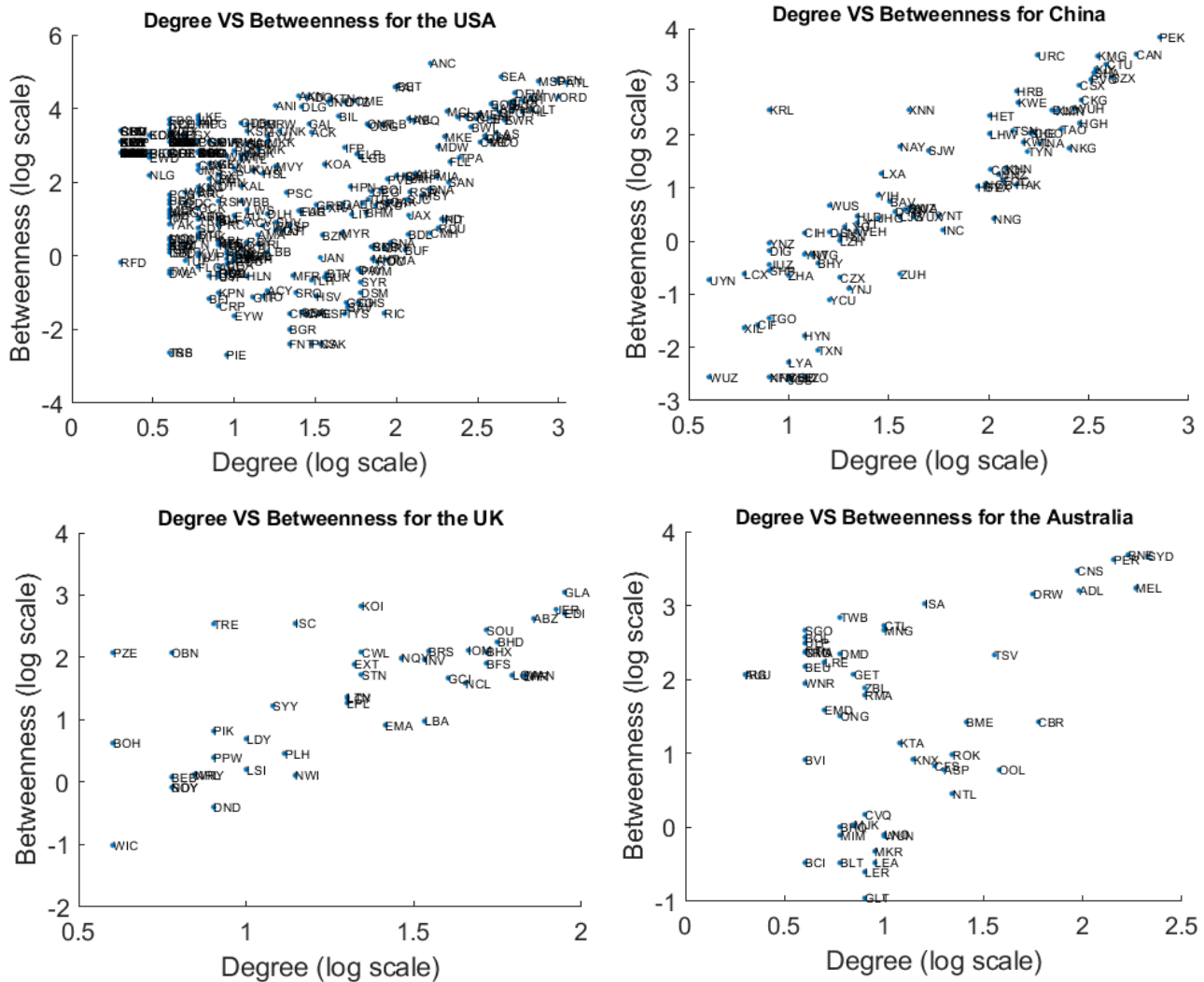


Figure 9: Degree VS Betweenness graph for each country (with labels)

It makes sense to find airports such as Seattle in the USA, Pekin in China, Sydney in Australia, or Glasgow in the UK at the top right hand-corner (CAT 1) since they are the most connected airports in big cities, connecting many people from anywhere to anywhere else. These well-connected airports have a high correlation of degree and betweenness. Anchorage for instance, has the highest betweenness centrality. Even if it is in Alaska, it is a main intermediary point for diverse routes. More, in China, the distribution is near to linear. Thus, the betweenness is positively correlated with the degree, which indicates that more important airports regarding their degree are also the most used as intermediary points.

At the opposite, at the bottom left hand-corner (CAT 2) are the most isolated locate airports.

In CAT 3 for the USA, the airports are mainly local aerodromes, which are not well connected but can serve as intermediary points in a longer trip.

Generally, this method is adequate to visualize the importance of airports and their main function (if they are central or more used as a stop point). This can be applicable for large or small network. For large networks, the whole trend of the distribution can be more easily distinguished. However, for smaller networks, the analysis focusses more on each airport individually.

## 4.3 Assortativity (degree correlation)

Assortativity is the correlation between degree of nodes. It measures how diverse the collaborations are. It can be computed as a general coefficient (*Figure 10*) or as an adjacency matrix plotting the different existing relations between the airports in each country (*Figure 11*).

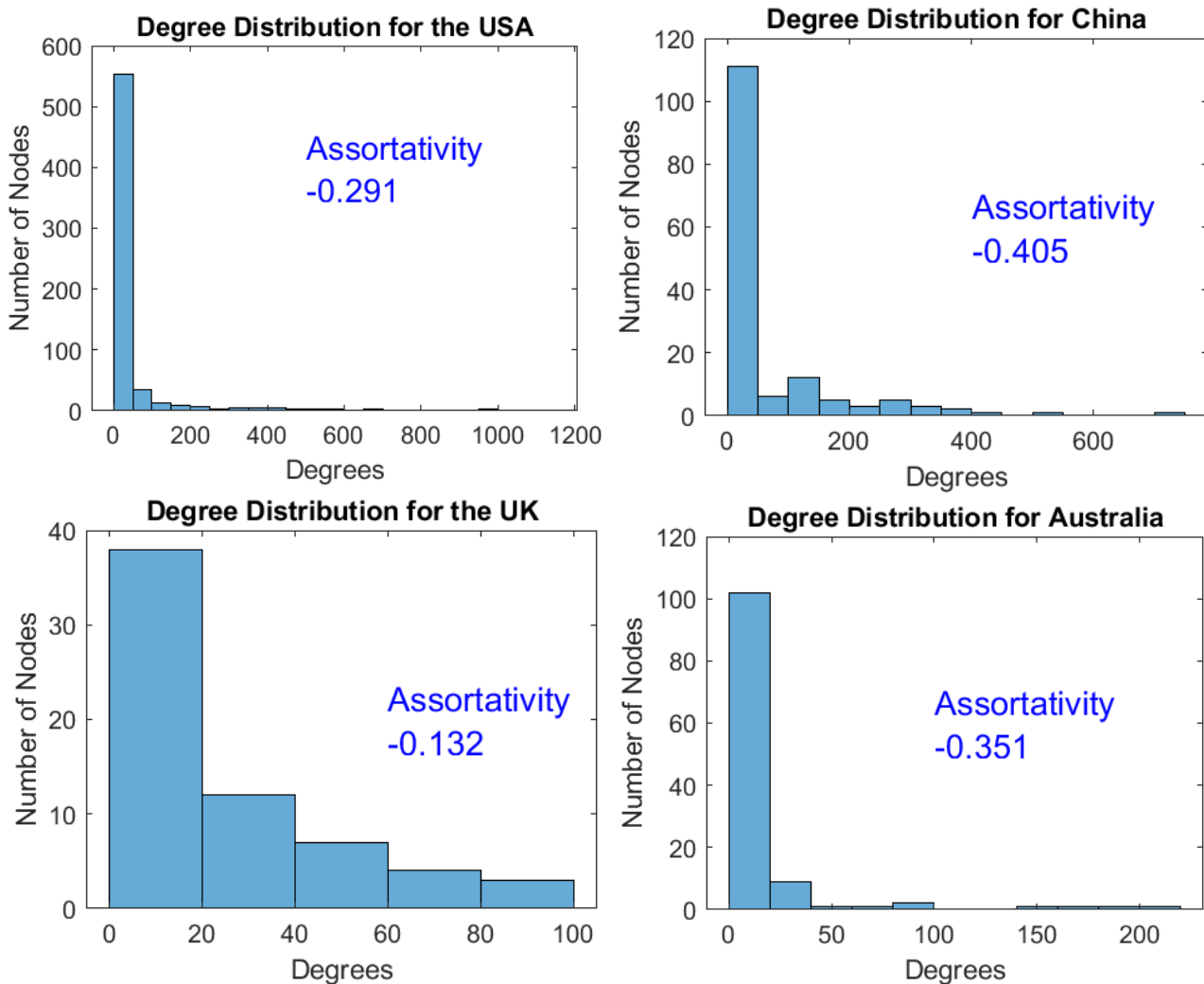


Figure 10: Degree distribution histogram for each country

All the assortativity coefficients are negative and reveal disassortative networks. It means that high degree nodes tend to link with low degree nodes. In other words, smaller airports have a tendency to connect to high degree airports. It shows diverse collaborations between airports due to relationships between nodes of different degrees. The more disassortative country is China since the assortativity coefficient is the closest to -1. But all coefficients are quite all similar. This could be explained by the globalization since the 20<sup>th</sup> century. Networks tend to have the most diverse paths to connect outlying areas to major cities.

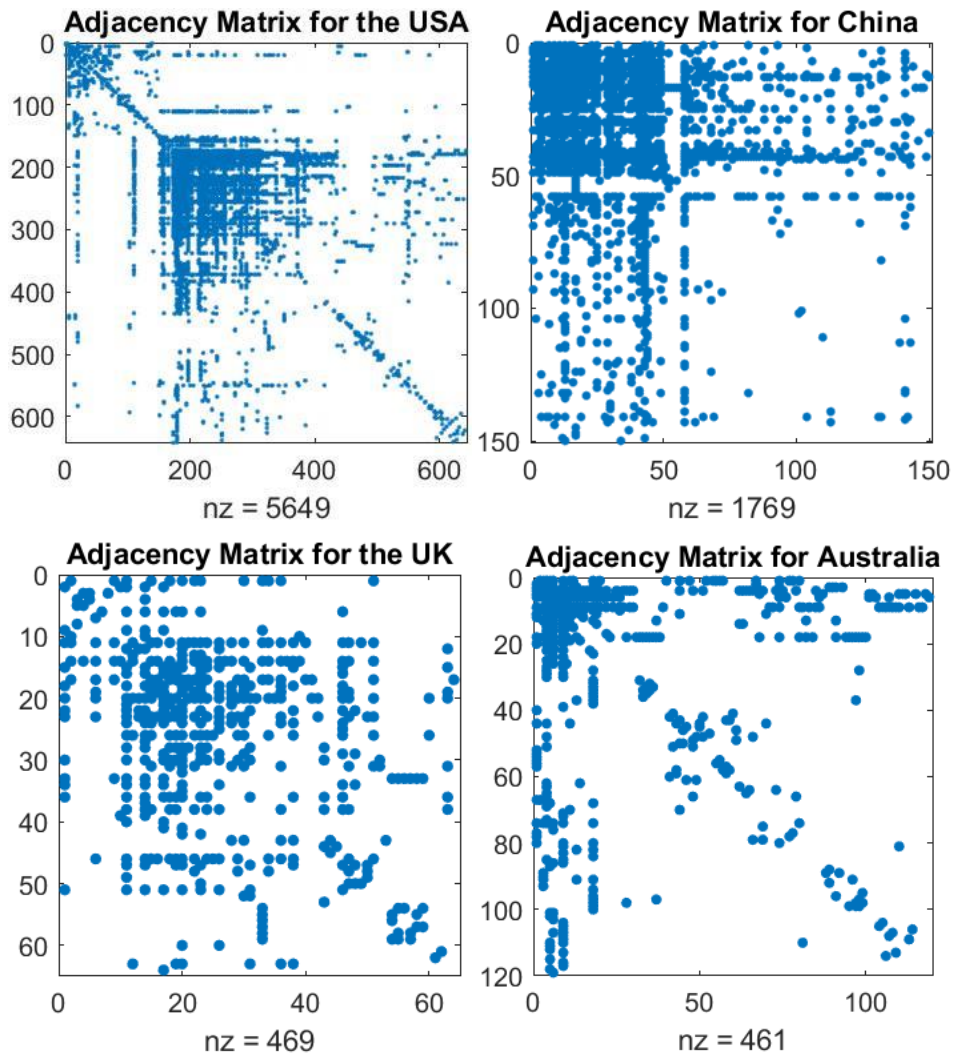


Figure 11: Adjacency matrix for each country

Regarding the degree correlation matrix for each country, the blue points indicate whether pair of vertices have an edge between them (or in other words if they are adjacent). For all the countries excepting Australia, a main core network can be observed (kind of square). These cores are coherent since there is at least one main concentration of airports for each country (Figure 5). The label 'nz' corresponds to the total number of different routes (without multi-edges). It is remarkable that for the UK, most of the routes belong to the core network. But it does not mean inevitably that other routes are associated with outlying airports.

The method could have been expended plotting the degree correlation matrix. To do that, it would have been needed to capture the probability of finding a node with degrees  $i$  and  $j$  (for  $i$  and  $j$  from  $\min(\text{degree})$  to  $\max(\text{degree})$ ) at the two ends of a randomly selected links.



## 4.4 Core community size

The core periphery structure can be analysed to check where the limit is between the core nodes (which are densely connected among themselves) and the periphery nodes.

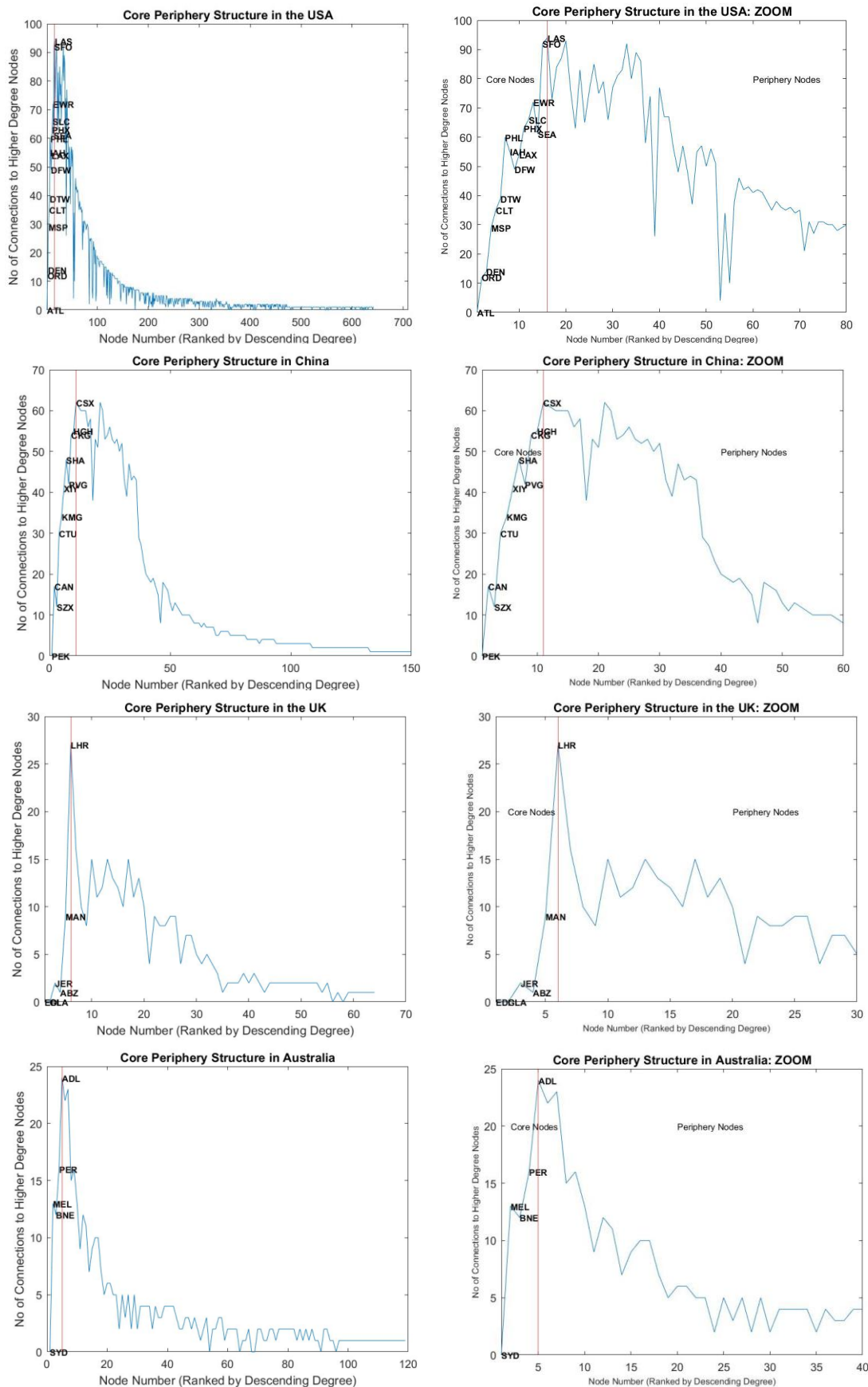


Figure 12: Core periphery structure for each country

The air transport networks on *Figure 12* contain cores whose sizes are listed in the table below:

Country	United States	China	United Kingdom	Australia
<b>Core community size</b>	16	11	6	5
<b>Core community proportion</b> (over the number of total nodes)	2,5 %	7,3 %	9,4 %	4,2 %

Core community sizes founded are coherent regarding the previous *GeoPlot* (*Figure 6*). For the USA and Australia, the cores don't represent even 10% of their network, which is very diversified for each. It makes sense since the USA is a vast country and needs to serve far areas of the country. Such small core sizes are a proof of a quite heterogenous air transport network tending to diversify routes. However, not enough cohesive cores could have an issue regarding the stability against potential perturbations of the network.

Generally, this method is adequate to visualize the importance of the network's core and if it is the main part of the network or not. If it is, the network is very centred. The method is performant to know more about the main core size but doesn't study the potential sub cores and their associated sizes in the network.

# 5 Impacts of future aircrafts

In this part, the impact of the design of future aircrafts in the networks are analysed.

Different hypotheses which could impact airline networks and how they would bring changes are listed below:

## 1. Reducing flight costs

It can be assumed that reducing flight costs would increase the influence of an airport or an area. Some flights would gain weights and become more influential. Consequently, specific airports could become more important and balance the current core community size. Obviously, increasing flight costs would create the opposite tend.

## 2. Modification of the attractivity of a zone

Routes could be added due to the increasing attractivity of some areas for some causes. These routes would have a weight associated with their new attractivity and disrupt the different degree distributions. More, the whole assortative of the network could be modified. But to change the assortativity in a subsistent way, many modifications would be necessary, to eventually give an assortative network. This is almost impossible in the absolute, unless all the network was upset.

Instead, some flight routes could be deleted due to some bad events such as environmental or political troubles in a country.

## 3. Fuel price disturbance

How would a graph change in the future if the fuel price increases?

At first sight, it seems that this increasing would increase the cost of the flight and decrease the passenger volume. In the worst case, some flights could be temporarily deleted due to the lack of passengers. This would affect the network and reduce the number of flights. But this impact can be verified generating random graphs (*Figure 13*) (other method in **NB\***) and analysing the changes in the distance penalty. Indeed, if the fuel price goes high, the distance penalty associated would increase to point the difficulty to travel since the flight cost would be more expensive.

But how the distance penalty would affect the real air network?

For a high value of the distance penalty, the graph would be less consistent (with less edges), without correlation between the degree and the betweenness. The airline transport would become less relevant since high degree airports wouldn't be the most important hubs (with high betweenness) anymore.

At the opposite, for a low distance penalty (if the fuel price goes down), hub airports would be the most important ones due to the strong correlation between the degree and the betweenness.

To conclude on this part, if the cost of travel is high, the routes couldn't take advantage of airports as hubs anymore. This network configuration prefers small airports. But when the cost goes down, the structure of the network moves to a more integrated and connected one, with more hub airports. Consequently, the first impression is confirmed: reducing fuel price (flight cost in the same way) tend to increase the importance of the hub airports in the network.

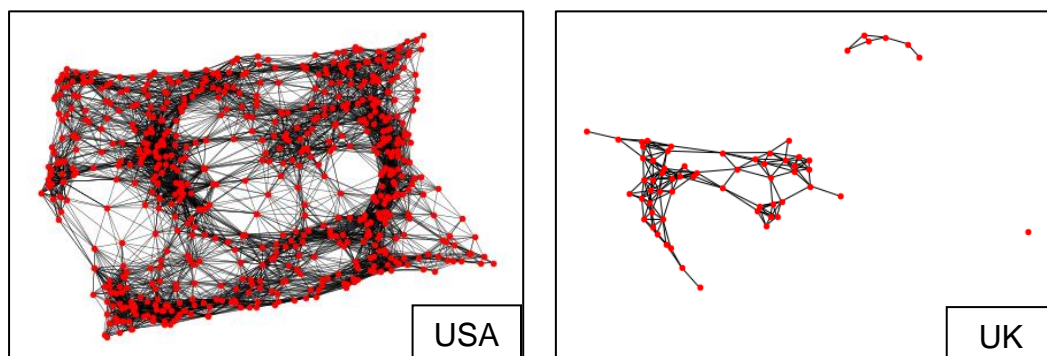


Figure 13: Examples of random graphs for the USA and the UK

## 4. Adaptation for developing countries

The evolution of population and economic factors such as the GDP, in developing countries could increase the flight route demand. Routes could be added, or airports could be expanded and consolidated, or even created if they are inexistant. Places of airports would be selected according of the population and distances at stake.

# 6 Extra-works

## 6.1 Weighted degree cumulative distribution

In the same way as the weighted degree distribution presented before, the degree has a cumulative distribution that obeys a power of law, visible for the higher degrees which don't follow the linear relationship.

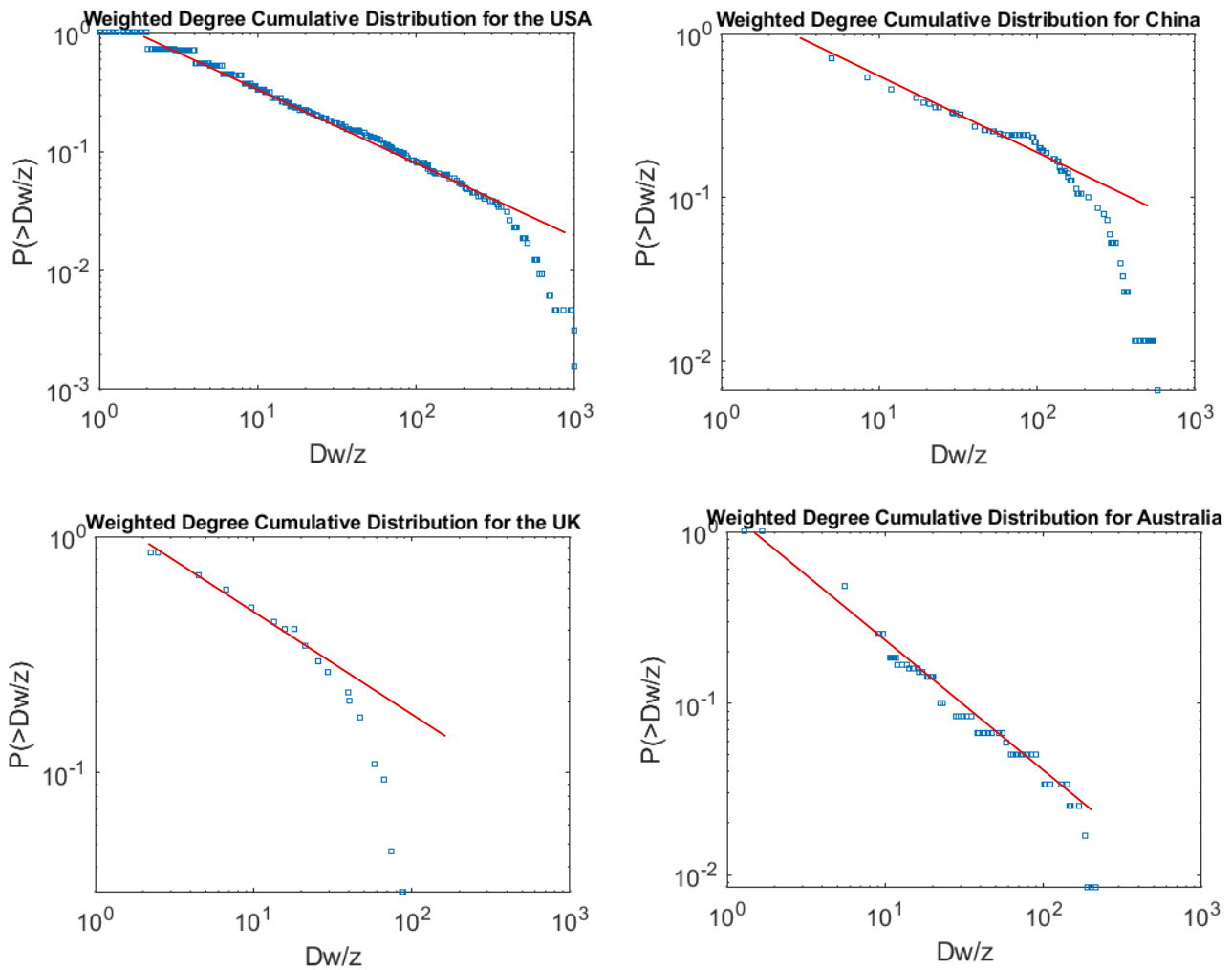


Figure 14: Weighted Degree Cumulative Distribution for each country

## 6.2 Normalized betweenness VS normalized degree

Normalized betweenness centrality helps to find the different communities in a meso-scale. They can be analysed in the same way as the weighted degree distribution regarding the role of airports, to know if they are stop points, or central points in the network.

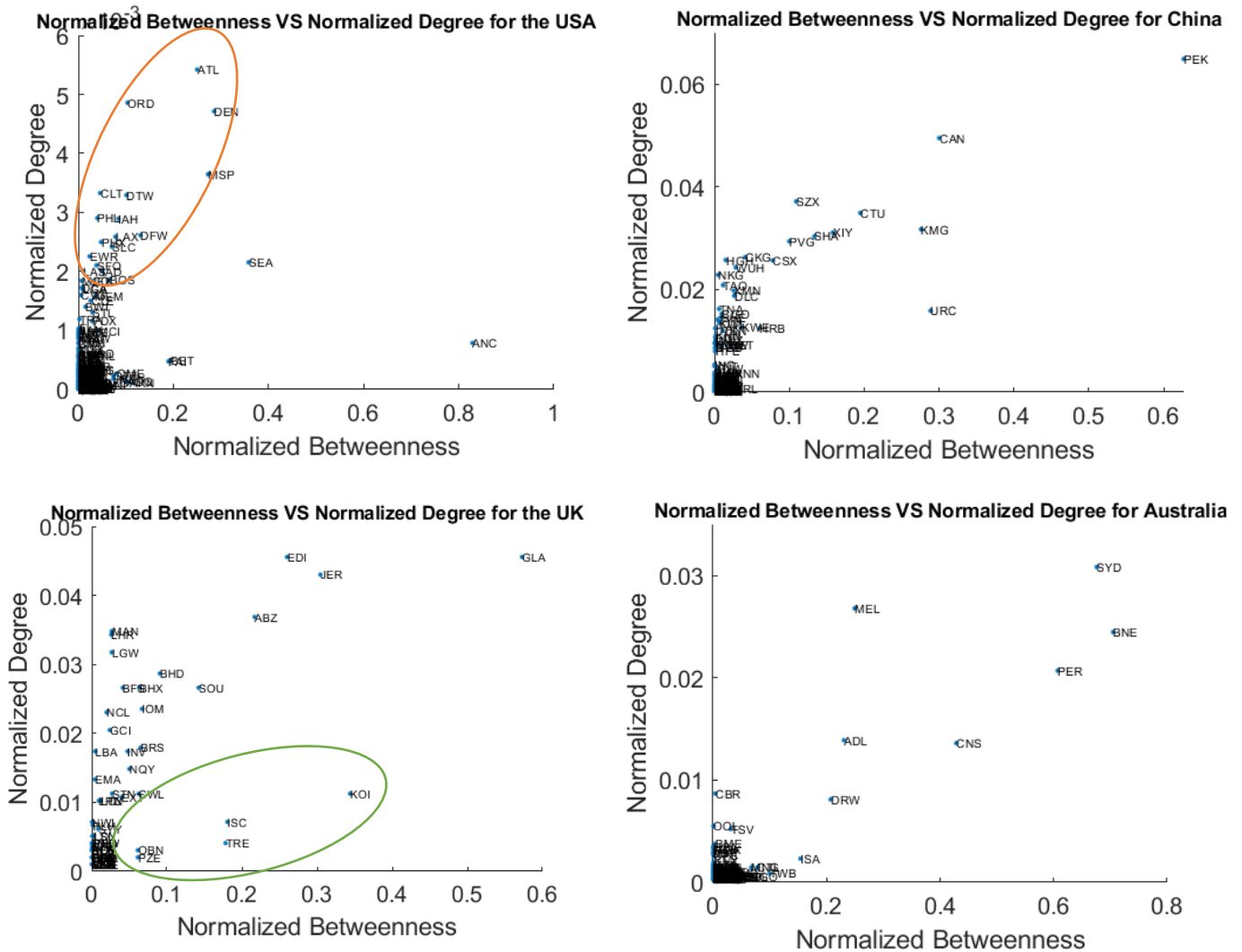


Figure 15: Normalized Betweenness VS Normalized Degree for each country

- CAT 1: Low distance cost (producing high degree airports)
- CAT 2: High distance cost (producing Hub-Spoke)

## 6.3 Communities' detection

Communities' detection methods are used to identify patterns such as grouped areas structures. In this case, the method selected is the *Modularity* method since it uses the higher link density very suitable for large networks. In more details, it computes the density of connections within a community. This method is used especially because it was preferable to use a method which doesn't require the number of communities to be performed as they are initially unknown.

For each country, the first step is to perform the algorithm and analyse the different clusters found on *Figure 16*.

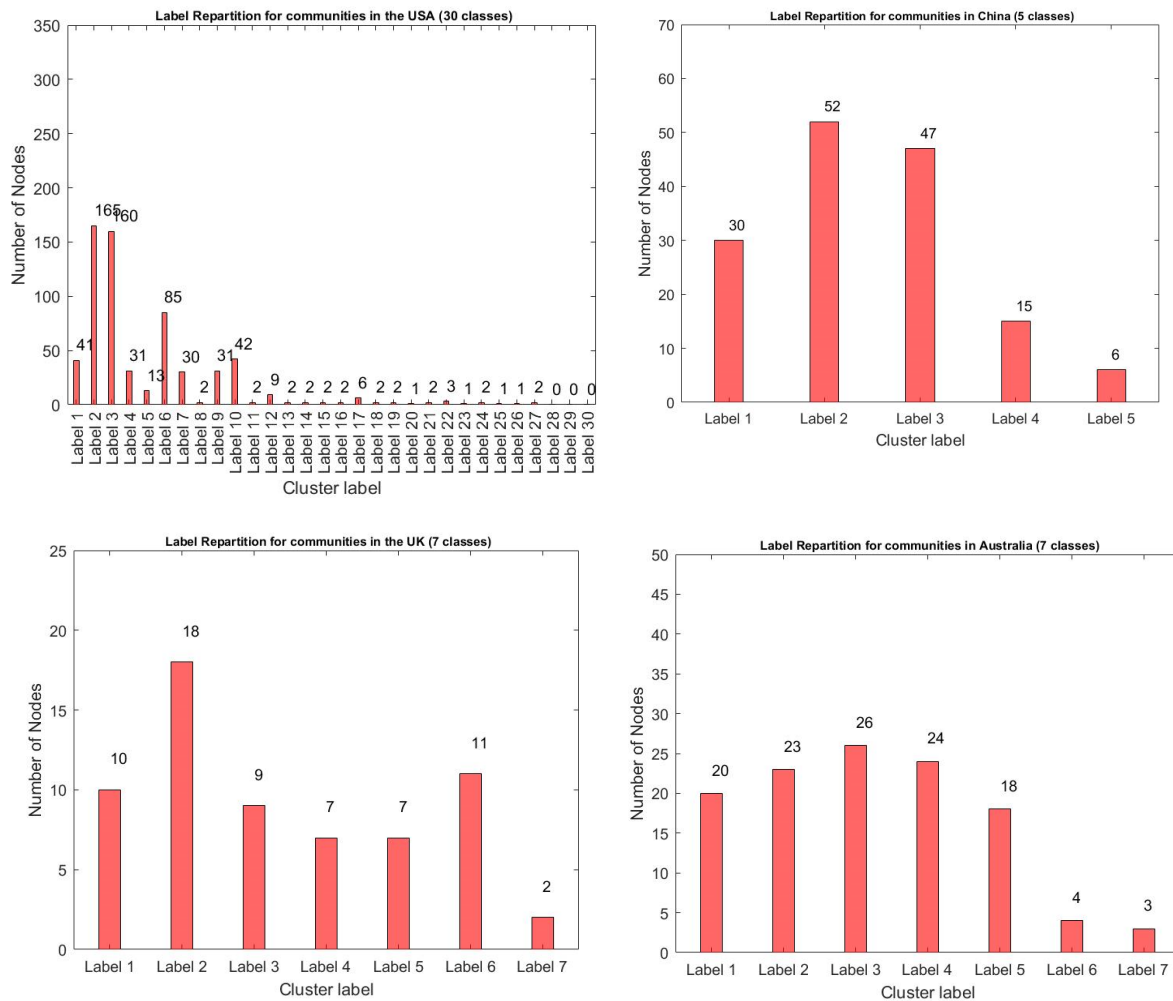


Figure 16: Communities' histograms for each country

For each, the main clusters are selected for map visualizations. This means the higher five clusters for the USA, the three higher for China, the six higher for the UK, and the five higher for Australia. These choices are arbitrary and done for a best visualization of the communities.



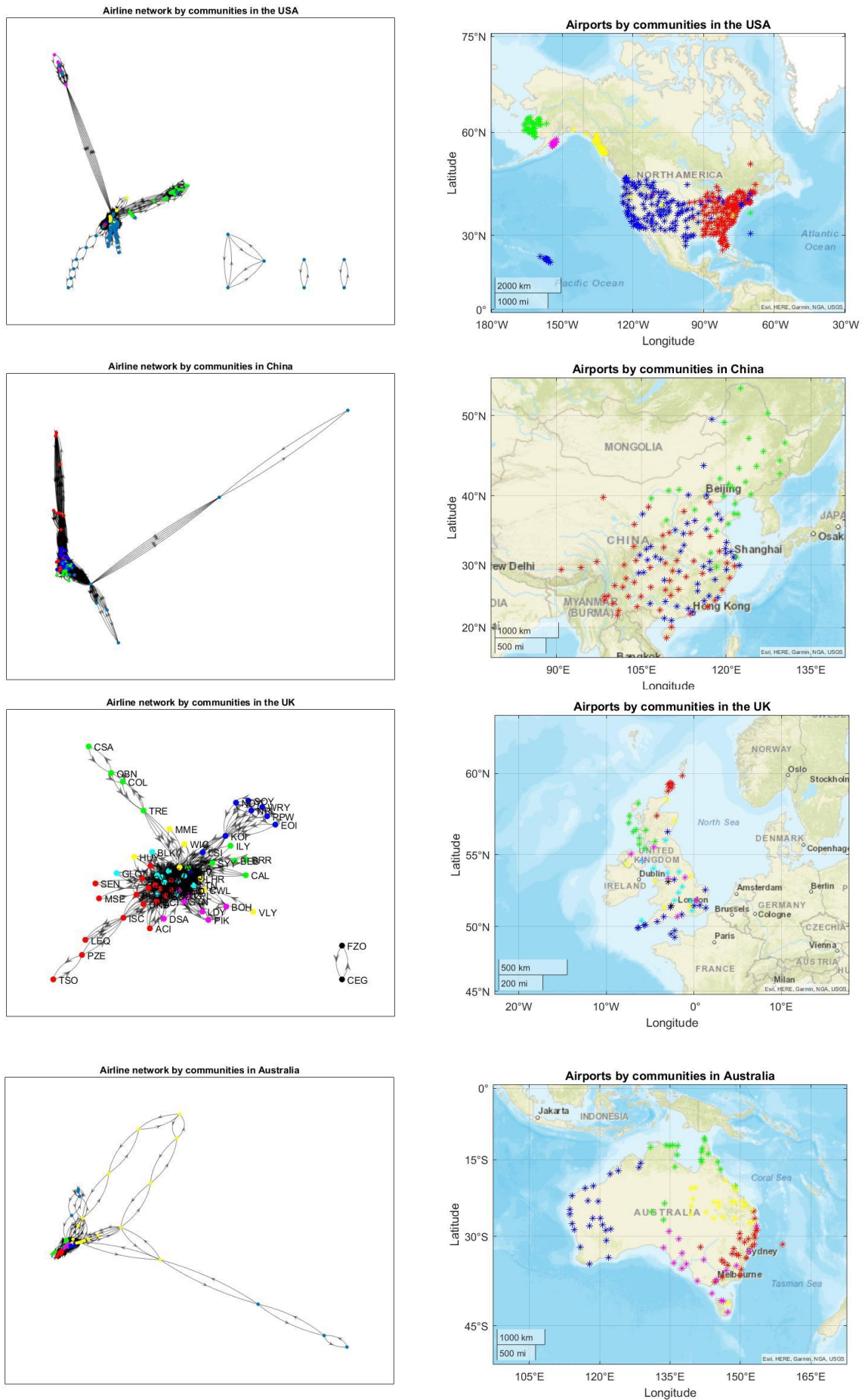


Figure 17: Communities' visualizations for each country

On *Figure 17*, different communities can be visualized on the maps for each country. For the USA and Australia, they are clearly parted. In the USA, three main clusters are present: one at the West, one at the East, and one in Alaska. In Australia, clusters are parted all around the coasts since the centre of the country is a desert. At the opposite, for the UK and China, clusters are less clearly visible. Indeed, their air transport networks are less clustered. This means that the most of airports serve many different other airports, and not only the ones of their zone.

Finally, the results of this part are to take lightly since the modularity method doesn't give the same results at each run due to the random pattern.

# 7 Conclusion

Nowadays, it is essential to study the air transport networks to control the transportation of people and goods all around the world. It is essential not to focus on a single method but implement several ones in order to draw the right conclusion at the end. Indeed, methods such as assortativity, betweenness and degree analysis are complementary. In some cases, analysing network taking into consideration current socioeconomic factors help to understand why a network is organized in a specific manner. More, besides to help predicting the future aircrafts in the network, the socioeconomic factors can also be deduced observing the organization of a network. For instance, by looking at different evolutions of the communities over time, the changing states of the market in specific regions could be understood.

## 8 Contents of figures

Figure 1: GeoData Dataset.....	4
Figure 2: GeoData Dataset extraction for the USA .....	4
Figure 3: Airports' visualization for each country .....	4
Figure 4: FlightData Dataset .....	5
Figure 5: Airline network visualization for each country .....	7
Figure 6: Airline network visualization for each country by GeoPlot .....	8
Figure 7: Weighted Degree Distribution for each country .....	9
Figure 8: Degree VS Betweenness graph for each country .....	11
Figure 9: Degree VS Betweenness graph for each country (with labels) .....	12
Figure 10: Degree distribution histogram for each country .....	13
Figure 11: Adjacency matrix for each country .....	14
Figure 12: Core periphery structure for each country .....	15
Figure 13: Examples of random graphs for the USA and the UK .....	17
Figure 14: Weighted Degree Cumulative Distribution for each country .....	18
Figure 15: Normalized Betweenness VS Normalized Degree for each country .....	19
Figure 16: Communities' histograms .....	20
Figure 17: Communities' visualizations for each country .....	21

# 9 Bibliography

## Contents

- [1] *Wikipedia*, accessed 5 December 2021, [https://en.wikipedia.org/wiki/Main\\_Page](https://en.wikipedia.org/wiki/Main_Page)
- [2] *MathWorks*, accessed 8 December 2021, <https://uk.mathworks.com/>
- [3] *Github*, accessed 11 December 2021, <https://github.com/>
- [4] *A complex network analysis of the United States Air Transportation*, Dorothy P.Cheung, Mehmet Hadi Gunes.
- [5] *The structure and periodicity of the Chinese Air Passenger Network*, Hongqi Li, Haotian Wang, Ming Bai, Bin Duan, 2018.
- [6] *Network Science, Degree Correlation*, Albert Laszlo Barabasi, 2014.

# 10 Code

## Graph Analysis of Air Transport Network

### Packages importation

```
import numpy.*
import pandas.*
import networkx.*
import matplotlib.pyplot.*
import Basemap.*
clear all
clc
```

### Data Importation

```
GeoData = readtable("Airports.csv");
GeoData = sortrows(GeoData);
% FlightData = readtable("FlightData.xlsx", 'Sheet', '2010-2016');
FlightData = readtable("FlightData.xlsx", 'Sheet', '2003-2009');
```

### Visualization of airports for each country (ExtractLocation function)

This function creates a specific dataset extraction for each country, and calculate the airports locations for each of them.

```
[USADData, USALat, USALong] = ExtractLocation('United States', GeoData);
[ChinaData, ChinaLat, ChinaLong] = ExtractLocation('China', GeoData);
[UKData, UKLat, UKLong] = ExtractLocation('United Kingdom', GeoData);
[AustraliaData, AustraliaLat, AustraliaLong] = ExtractLocation('Australia', GeoData);

figure,
geoplot(USALat,USALong, '*')
geolimits([0 75],[-160 -50]) % USA
title 'Airports in the USA';
geobasemap streets

figure,
geoplot(ChinaLat,ChinaLong, '*')
geolimits([0 60],[50 150]) % China
title 'Airports in China';
geobasemap streets

figure,
geoplot(UKLat,UKLong, '*')
hold on
for i = 1:length(UKLat)
    text(UKLat(i), UKLong(i), UKData.id(i), 'FontSize', 5);
end
hold on
geolimits([45 63],[-15 10]) % UK
title 'Airports in the UK';
geobasemap streets
```



```
figure,
geoplot(AustraliaLat,AustraliaLong,'*')
geolimits([-50 0],[120 150]) % Australia
title 'Airports in Australia';
geobasemap streets
```

### Cleaning of the data before getting the sources and targets positions :

- Delete the sources and targets which don't have their positions in GeoData

```
[USASource, USATarget, USAWeights, USALines] = cleaning('USA', FlightData, USAData);
[ChinaSource, ChinaTarget, ChinaWeights, ChinaLines] = cleaning('China', FlightData, ChinaData);
[UKSource, UKTarget, UKWeights, UKLines] = cleaning('United Kingdom', FlightData, UKData);
[AustraliaSource, AustraliaTarget, AustraliaWeights, AustraliaLines] = cleaning('Australia',
FlightData, AustraliaData);
```

### Plot Directed Graph

```
G_USA = digraph(USASource, USATarget, USAWeights);
G_China = digraph(ChinaSource, ChinaTarget, ChinaWeights);
G_UK = digraph(UKSource, UKTarget, UKWeights);
G_Australia = digraph(AustraliaSource, AustraliaTarget, AustraliaWeights);

figure,
nodes = unique(USATarget)
plot(G_USA,'NodeLabel', nodes);
title('Airline network in the USA')

figure,
nodes = unique(ChinaTarget)
plot(G_China,'NodeLabel', nodes);
title('Airline network in China')

figure,
nodes = unique(UKTarget)

plot(G_UK,'NodeLabel', nodes);
title('Airline network in the UK')

figure,
nodes = unique(AustraliaTarget)

plot(G_Australia,'NodeLabel', nodes);
title('Airline network in Australia')

% figure,
% nodes = unique(USATarget)
% plot(G_USA,'EdgeLabel',G_USA.Edges.Weight);
% title('Airports connections Network in the USA (with weights labels)')
```

Largest connected component subgraph (for the USA)

```
[bin_max,binsize_max] = conncomp(G_USA)
idx_max = binsize_max(bin_max) == max(binsize_max);
SG_max = subgraph(G_USA, idx_max);

figure,
un = unique(USATarget)
nodes_max = un(idx_max)

h_max = plot(SG_max,'NodeLabel', nodes_max)
h_max.NodeColor = 'r';
h_max.EdgeColor = 'black';

title('Well connected airports network in the USA')
```

Smallest connected component subgraph (for the USA)

```
[bin_min,binsize_min] = conncomp(G_USA)
idx_min = binsize_min(bin_min) == min(binsize_min);
SG_min = subgraph(G_USA, idx_min);

figure,
nodes_min = un(idx_min)

h_min = plot(SG_min,'NodeLabel', nodes_min)
h_min.NodeColor = 'r';
h_min.EdgeColor = 'black';

title('Worse connected airports network in the USA')
```

Medium connected component subgraph

```
[bin_av,binsize_av] = conncomp(G_USA)

idx_av = binsize_av(bin_av) == 3;
SG_av = subgraph(G_USA, idx_av);

figure,
nodes_av = un(idx_av)

h_avg = plot(SG_av,'NodeLabel', nodes_av)
h_avg.NodeColor = 'r';
h_avg.EdgeColor = 'black';

title('Not very well connected airports network in the USA', 'FontSize',9)
```

Table with the degree corresponding to each node

```

TableDegreeUSA = TableDegree(G_USA);
TableDegreeChina = TableDegree(G_China);
TableDegreeUK = TableDegree(G_UK);
TableDegreeAustralia = TableDegree(G_Australia);

```

## Airports Network Visualization

```

% Positions of airports for sources and targets of the lines
[USALatSource, USALongSource, USALatTarget, USALongTarget] = PositionsLines(USASource, USATarget,
USAData);

[ChinaLatSource, ChinaLongSource, ChinaLatTarget, ChinaLongTarget] = PositionsLines(ChinaSource,
ChinaTarget, ChinaData);
[UKLatSource, UKLongSource, UKLatTarget, UKLongTarget] = PositionsLines(UKSource, UKTarget, UKData);

[AustraliaLatSource, AustraliaLongSource, AustraliaLatTarget, AustraliaLongTarget] =
PositionsLines(AustraliaSource, AustraliaTarget, AustraliaData);

```

### AUSTRALIA

```

figure,
for i = 1:size(AustraliaLatSource,1)
    geoplan([AustraliaLatSource(i) AustraliaLatTarget(i)], [AustraliaLongSource(i)
AustraliaLongTarget(i)], '-', 'Color', [0.3 0.7 0]);

    text(AustraliaLatSource(i), AustraliaLongSource(i), AustraliaSource(i), 'FontSize', 5);
    text(AustraliaLatTarget(i), AustraliaLongTarget(i), AustraliaTarget(i), 'FontSize', 5);

    Size1 = table2array(TableDegreeAustralia(strcmp(TableDegreeAustralia.Name,
AustraliaSource(i)),2)); % Size corresponding to the node degree
    Size2 = table2array(TableDegreeAustralia(strcmp(TableDegreeAustralia.Name,
AustraliaTarget(i)),2)); % Size corresponding to the node degree

    geoplan(AustraliaLatSource(i), AustraliaLongSource(i), '.', 'Color', 'r', 'MarkerSize', Size1/10)
    geoplan(AustraliaLatTarget(i), AustraliaLongTarget(i), '.', 'Color', 'r', 'MarkerSize', Size2/10)

    hold on
end
hold on
geolimits([-50 0], [120 150]) % Australia
title ('Connection density visualization for airports in Australia', 'FontSize', 9);
geobasemap streets

```

### UK

```

figure,
for i = 1:size(UKLatSource,1)
    geoplan([UKLatSource(i) UKLatTarget(i)], [UKLongSource(i) UKLongTarget(i)], '-', 'Color', [0.3 0.7
0]);

```

```

text(UKLatSource1(i), UKLongSource1(i), UKSource1(i), 'FontSize', 5)
text(UKLatTarget1(i), UKLongTarget1(i), UKTarget1(i), 'FontSize', 5)

Size1 = table2array(TableDegreeUK(strcmp(TableDegreeUK.Name, UKSource(i)),2)); % Size
corresponding to the node degree
Size2 = table2array(TableDegreeUK(strcmp(TableDegreeUK.Name, UKTarget(i)),2)); % Size
corresponding to the node degree

geoplot(UKLatSource(i),UKLongSource(i),'.','Color','r','MarkerSize',Size1/10)
geoplot(UKLatTarget(i),UKLongTarget(i),'.','Color','r','MarkerSize',Size2/10)

hold on
end
hold on
geolimits([45 63],[-15 10]) % UK
title ('Connection density visualization for airports in the UK', 'FontSize', 9);
geobasemap streets

```

## USA

```

figure,
for i = 1:size(USALatSource,1)
    % geoplot([USALatSource(i) USALatTarget(i)],[USALongSource(i) USALongTarget(i)],'-','Color',[0.3
0.7 0],'LineWidth', USAWeights1(i)/10000)

    % text(USALatSource(i), USALongSource(i), USASource(i), 'FontSize', 5)
    % text(USALatTarget(i), USALongTarget(i), USATarget(i), 'FontSize', 5)

    Size1 = table2array(TableDegreeUSA(strcmp(TableDegreeUSA.Name, USASource(i)),2)); % Size
corresponding to the node degree
    Size2 = table2array(TableDegreeUSA(strcmp(TableDegreeUSA.Name, USATarget(i)),2)); % Size
corresponding to the node degree

    geoplot(USALatSource(i),USALongSource(i),'.','Color','r','MarkerSize',Size1/10)
    geoplot(USALatTarget(i),USALongTarget(i),'.','Color','r','MarkerSize',Size2/10)

    hold on
end
hold on
geolimits([0 75],[-160 -50]) % USA
title ('Connection density visualization for airports in the USA', 'FontSize', 9);
geobasemap streets

```

## China

```

figure,
for i = 1:size(ChinaLatSource,1)
    geoplot([ChinaLatSource(i) ChinaLatTarget(i)],[ChinaLongSource(i) ChinaLongTarget(i)],'-
','Color',[0.3 0.7 0]);

```

```

text(ChinaLatSource(i), ChinaLongSource(i), ChinaSource(i), 'FontSize', 5)
text(ChinaLatTarget(i), ChinaLongTarget(i), ChinaTarget(i), 'FontSize', 5)

Size1 = table2array(TableDegreeChina(strcmp(TableDegreeChina.Name, ChinaSource(i)),2)); % Size
corresponding to the node degree
Size2 = table2array(TableDegreeChina(strcmp(TableDegreeChina.Name, ChinaTarget(i)),2)); % Size
corresponding to the node degree

geoplot(ChinaLatSource(i),ChinaLongSource(i),'.','Color','r','MarkerSize',Size1/10)
geoplot(ChinaLatTarget(i),ChinaLongTarget(i),'.','Color','r','MarkerSize',Size2/10)

hold on
end
hold on
geolimits([0 60],[50 150]) % China
title ('Connection density visualization for airports in China', 'FontSize', 9);
geobasemap streets

```

### Weighted Degree Distribution

```

[y1, rank1, weights1, rsq1, p1] = WDD(G_USA, 200, 500);
[y2, rank2, weights2, rsq2, p2] = WDD(G_China, 50, 100);
[y3, rank3, weights3, rsq3, p3] = WDD(G_UK, 20, 40);
[y4, rank4, weights4, rsq4, p4] = WDD(G_Australia, 40, 80);

```

USA

```

figure,
plot(weights1);
hold on
plot(rank1,y1);
xlim([0 700]);
xlabel('Rank');
ylabel('Weighted Degree (log scale)');
txt = {'b', round(p1(1),3)},
text(100,2,txt,'Color','blue','FontSize',13);
txt2 = {'R^2', round(rsq1,3)},
text(400,2,txt2,'Color','red','FontSize',13);
title('Weighted Degree Distribution for the USA');

```

CHINA

```

figure,
plot(weights2);
hold on
plot(rank2,y2);
xlabel('Rank');
ylabel('Weighted Degree (log scale)');
xlim([0 160]);
txt = {'b', round(p2(1),3)},

```

```
text(20,3,txt,'Color','blue','FontSize',13);
txt2 = {'R^2', round(rsq2,3)},
text(70,3,txt2,'Color','red','FontSize',13);
title('Weighted Degree Distribution for China');
```

UK

```
figure,
plot(weights3);
hold on
plot(rank3,y3);
xlabel('Rank');
ylabel('Weighted Degree (log scale)');
xlim([0 70]);
txt = {'b', round(p3(1),3)},
text(10,3,txt,'Color','blue','FontSize',13);
txt2 = {'R^2', round(rsq3,3)},
text(30,3,txt2,'Color','red','FontSize',13);
title('Weighted Degree Distribution for the UK');
```

AUSTRALIA

```
figure,
plot(weights4);
hold on
plot(rank4,y4);
xlabel('Rank');
ylabel('Weighted Degree (log scale)');
xlim([0 130]);
txt = {'b', round(p4(1),3)},
text(10,3,txt,'Color','blue','FontSize',13);
txt2 = {'R^2', round(rsq4,3)},
text(60,3,txt2,'Color','red','FontSize',13);
title('Weighted Degree Distribution for Australia');
```

## Weighted Degree Cumulative Distribution

```
[Dwz_USA, C_USA] = WDCD(TableDegreeUSA, G_USA);
[Dwz_China, C_China] = WDCD(TableDegreeChina, G_China);
[Dwz_UK, C_UK] = WDCD(TableDegreeUK, G_UK);
[Dwz_Australia, C_Australia] = WDCD(TableDegreeAustralia, G_Australia);
```

USA

```
figure,
scatter(Dwz_USA,C_USA, 's', 'MarkerSize', 3);
xlim([1 1000]) ;
xlabel('Dw/z')
ylabel('P(>Dw/z)')
title('Weighted Degree Cumulative Distribution for the USA','FontSize',8)
```

China



```
figure,
scatter(Dwz_China,C_China, 's', 'MarkerSize', 3);
xlim([1 1000]);
xlabel('Dw/z')
ylabel('P(>Dw/z)')
title('Weighted Degree Cumulative Distribution for the USA','FontSize',8)
```

UK

```
figure,
scatter(Dwz_UK,C_UK, 's', 'MarkerSize', 3);
xlim([1 1000]);
xlabel('Dw/z')
ylabel('P(>Dw/z)')
title('Weighted Degree Cumulative Distribution for the UK','FontSize',8)
```

AUSTRALIA

```
figure,
scatter(Dwz_Australia,C_Australia, 's', 'MarkerSize', 3);
xlim([1 1000]);
xlabel('Dw/z')
ylabel('P(>Dw/z)')
title('Weighted Degree Cumulative Distribution for Australia','FontSize',8)
```

### Assortativity

```
% Adjacency matrix and Assortativity

Adj_USA = adjacency(G_USA);
A_USA = assortativity(Adj_USA); % -0.2669

Adj_China = adjacency(G_China);
A_China = assortativity(Adj_China); % -0.4409

Adj_UK = adjacency(G_UK);
A_UK = assortativity(Adj_UK); % -0.1390

Adj_Australia = adjacency(G_Australia);
A_Australia = assortativity(Adj_Australia); % -0.3042

figure,
spy(Adj_USA) % plot the non-zero elements (with 1)
% when there is a point there is a route between both concerned nodes
% Square core visualization (top left hand corner) : core periphery
title('Adjacency Matrix for the USA')

figure,
spy(Adj_China)
title('Adjacency Matrix for China')

figure,
```

```

spy(Adj_UK)
title('Adjacency Matrix for the UK')

figure,
spy(Adj_Australia)
title('Adjacency Matrix for Australia')

```

## Degree Distribution

```

figure,
histogram(table2array(TableDegreeUSA(:,2)));
% restore default
xlabel('Degrees')
ylabel('Number of Nodes')
title('Degree Distribution for the USA')
txt = {'Assortativity ' round(A_USA,3)},
text(500,400,txt,'Color','blue','FontSize',13);

figure,
histogram(table2array(TableDegreeChina(:,2)));
% restore default
xlabel('Degrees')
ylabel('Number of Nodes')
title('Degree Distribution for China')
txt = {'Assortativity ' round(A_China,3)},
text(400,60,txt,'Color','blue','FontSize',13);

figure,
histogram(table2array(TableDegreeUK(:,2)));
% restore default
xlabel('Degrees')
ylabel('Number of Nodes')
title('Degree Distribution for the UK')
txt = {'Assortativity ' round(A_UK,3)},
text(60,20,txt,'Color','blue','FontSize',13);

figure,
histogram(table2array(TableDegreeAustralia(:,2)));
% restore default
xlabel('Degrees')
ylabel('Number of Nodes')
title('Degree Distribution for Australia')
txt = {'Assortativity ' round(A_Australia,3)},
text(100,60,txt,'Color','blue','FontSize',13);

```

## Normalized Betweenness VS Normalized Degree

```

[BC_USA, BC_n_USA, D_USA, D_n_USA] = nbnd(G_USA);
[BC_China, BC_n_China, D_China, D_n_China] = nbnd(G_China);
[BC_UK, BC_n_UK, D_UK, D_n_UK] = nbnd(G_UK);
[BC_Australia, BC_n_Australia, D_Australia, D_n_Australia] = nbnd(G_Australia);

```

```

figure,
scatter(BC_n_USA,D_n_USA,30,'. ');
for i=1:size(BC_USA,1)
    text(BC_n_USA(i), D_n_USA(i), table2array(G_USA.Nodes(i,1)), 'FontSize',5);
end
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Normalized Betweenness VS Normalized Degree for the USA','FontSize',8)

figure,
scatter(BC_n_China,D_n_China,30,'. ');
for i=1:size(BC_China,1)
    text(BC_n_China(i), D_n_China(i), table2array(G_China.Nodes(i,1)), 'FontSize',5);
end
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Normalized Betweenness VS Normalized Degree for China','FontSize',8)

figure,
scatter(BC_n_UK,D_n_UK,30,'. ');
for i=1:size(BC_UK,1)
    text(BC_n_UK(i), D_n_UK(i), table2array(G_UK.Nodes(i,1)), 'FontSize',5);
end
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Normalized Betweenness VS Normalized Degree for the UK','FontSize',8)

figure,
scatter(BC_n_Australia,D_n_Australia,30,'. ');
for i=1:size(BC_Australia,1)
    text(BC_n_Australia(i), D_n_Australia(i), table2array(G_Australia.Nodes(i,1)), 'FontSize',5);
end
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Normalized Betweenness VS Normalized Degree for Australia','FontSize',8)

```

### Degree VS Betweenness

```

figure,
scatter(log10(D_USA),log10(BC_USA),'. ');
% for i=1:size(BC_USA,1)
%     text(log10(D_USA(i)),log10(BC_USA(i)), table2array(G_USA.Nodes(i,1)), 'FontSize',5);
% end
xlabel('Degree (log scale)')
ylabel('Betweenness (log scale)')
title('Degree VS Betweenness for the USA','FontSize',8)

```

```

figure,
scatter(log10(D_China),log10(BC_China),'.');
% for i=1:size(BC_China,1)
%     text(log10(D_China(i)),log10(BC_China(i)), table2array(G_China.Nodes(i,1)), 'FontSize',5);
% end
xlabel('Degree (log scale)')
ylabel('Betweenness (log scale)')
title('Degree VS Betweenness for China','FontSize',8)

figure,
scatter(log10(D_UK),log10(BC_UK),'.');
% for i=1:size(BC_UK,1)
%     text(log10(D_UK(i)),log10(BC_UK(i)), table2array(G_UK.Nodes(i,1)), 'FontSize',5);
% end
xlabel('Degree (log scale)')
ylabel('Betweenness (log scale)')
title('Degree VS Betweenness for the UK','FontSize',8)

figure,
scatter(log10(D_Australia),log10(BC_Australia),'.');
% for i=1:size(BC_Australia,1)
%     text(log10(D_Australia(i)),log10(BC_Australia(i)),
table2array(G_Australia.Nodes(i,1)), 'FontSize',5);
% end
xlabel('Degree (log scale)')
ylabel('Betweenness (log scale)')
title('Degree VS Betweenness for the Australia','FontSize',8)

```

## Core Periphery Structure

USA

```

[ConnectionUSA] = CPS(TableDegreeUSA, G_USA);

figure,
RankNodeUSA = 1:size(ConnectionUSA,1); % Rank of nodes
RankNodeUSA = reshape(RankNodeUSA,[size(ConnectionUSA,1),1])
CoNumbUSA = table2array(ConnectionUSA(:,{'GroupCount'}));
[x1USA, y1USA] = max(CoNumbUSA); % y1 = index of the max value
plot(RankNodeUSA,CoNumbUSA);
hold on
xline(y1USA,'-r');

xlim([1 710]);
xlabel('Node Number (Ranked by Descending Degree)');
ylabel('No of Connections to Higher Degree Nodes');
title('Core Periphery Structure in the USA');
for i=1:y1USA
    text(RankNodeUSA(i), CoNumbUSA(i), table2array(ConnectionUSA(i,2), 'FontSize', 6, 'FontWeight',
'bold'));

```

```

% ZOOM
figure,
plot(RankNodeUSA,CoNumbUSA);
hold on
xline(y1USA,'-r');
xlim([1 80]);
%ylim([0 40]);
xlabel('Node Number (Ranked by Descending Degree)','FontSize',8);
ylabel('No of Connections to Higher Degree Nodes','FontSize',8);
title('Core Periphery Structure in the USA: ZOOM');
text(60,80,'Periphery Nodes','FontSize',8);
text(3,80,'Core Nodes','FontSize',8);
for i=1:y1USA
    text(RankNodeUSA(i), CoNumbUSA(i), table2array(ConnectionUSA(i,2), 'FontSize', 6, 'FontWeight',
'bold'));

```

China

```

[ConnectionChina] = CPS(TableDegreeChina, G_China);

figure,
RankNodeChina = 1:size(ConnectionChina,1); % Rank of nodes
RankNodeChina = reshape(RankNodeChina,[size(ConnectionChina,1),1])
CoNumbChina = table2array(ConnectionChina(:,{'GroupCount'}));
[x1China, y1China] = max(CoNumbChina); % y1 = index of the max value
plot(RankNodeChina,CoNumbChina);
hold on
xline(y1China,'-r');

xlabel('Node Number (Ranked by Descending Degree)');
ylabel('No of Connections to Higher Degree Nodes');
title('Core Periphery Structure in China');
for i=1:y1China
    text(RankNodeChina(i), CoNumbChina(i), table2array(ConnectionChina(i,2), 'FontSize', 6,
'FontWeight', 'bold'));

% ZOOM
figure,
plot(RankNodeChina,CoNumbChina);
hold on
xline(y1China,'-r');
xlim([1 80]);
%ylim([0 40]);
xlabel('Node Number (Ranked by Descending Degree)','FontSize',8);
ylabel('No of Connections to Higher Degree Nodes','FontSize',8);
title('Core Periphery Structure in China: ZOOM');
text(40,50,'Periphery Nodes','FontSize',8);
text(3,50,'Core Nodes','FontSize',8);
for i=1:y1China

```

```
text(RankNodeChina(i), CoNumbChina(i), table2array(ConnectionChina(i,2), 'FontSize', 6,
'FontWeight', 'bold'));
```

UK

```
[ConnectionUK] = CPS(TableDegreeUK, G_UK);

figure,
RankNodeUK = 1:size(ConnectionUK,1); % Rank of nodes
RankNodeUK = reshape(RankNodeUK,[size(ConnectionUK,1),1])
CoNumbUK = table2array(ConnectionUK(:,{'GroupCount'}));
[x1UK, y1UK] = max(CoNumbUK); % y1 = index of the max value
plot(RankNodeUK,CoNumbUK);
hold on
xline(y1UK,'-r');

xlim([1 70]);
xlabel('Node Number (Ranked by Descending Degree)');
ylabel('No of Connections to Higher Degree Nodes');
title('Core Periphery Structure in the UK');
for i=1:y1UK
    text(RankNodeUK(i), CoNumbUK(i), table2array(ConnectionUK(i,2), 'FontSize', 6, 'FontWeight',
'bold'));
```

```
% ZOOM
figure,
plot(RankNodeUK,CoNumbUK);
hold on
xline(y1UK,'-r');
xlim([1 50]);
ylim([0 40]);
xlabel('Node Number (Ranked by Descending Degree)','FontSize',8);
ylabel('No of Connections to Higher Degree Nodes','FontSize',8);
title('Core Periphery Structure in the UK: ZOOM');
text(30,20,'Periphery Nodes','FontSize',8);
text(2,20,'Core Nodes','FontSize',8);
for i=1:y1UK
    text(RankNodeUK(i), CoNumbUK(i), table2array(ConnectionUK(i,2), 'FontSize', 6, 'FontWeight',
'bold'));
```

Australia

```
[ConnectionAustralia] = CPS(TableDegreeAustralia, G_Australia);

figure,
RankNodeAustralia = 1:size(ConnectionAustralia,1); % Rank of nodes
RankNodeAustralia = reshape(RankNodeAustralia,[size(ConnectionAustralia,1),1])
CoNumbAustralia = table2array(ConnectionAustralia(:,{'GroupCount'}));
[x1Australia, y1Australia] = max(CoNumbAustralia); % y1 = index of the max value
plot(RankNodeAustralia,CoNumbAustralia);
hold on
```

```

xline(y1Australia, '-r');

xlabel('Node Number (Ranked by Descending Degree)');
ylabel('No of Connections to Higher Degree Nodes');
title('Core Periphery Structure in Australia');
for i=1:y1Australia
    text(RankNodeAustralia(i), CoNumbAustralia(i), table2array(ConnectionAustralia(i,2),
'FontSize', 6, 'FontWeight', 'bold');

% ZOOM
figure,
plot(RankNodeAustralia,CoNumbAustralia);
hold on
xline(y1Australia, '-r');
xlim([1 40]);
%ylim([0 40]);
xlabel('Node Number (Ranked by Descending Degree)', 'FontSize', 8);
ylabel('No of Connections to Higher Degree Nodes', 'FontSize', 8);
title('Core Periphery Structure in Australia: ZOOM');
text(20,20, 'Periphery Nodes', 'FontSize', 8);
text(2,20, 'Core Nodes', 'FontSize', 8);
for i=1:y1Australia
    text(RankNodeAustralia(i), CoNumbAustralia(i), table2array(ConnectionAustralia(i,2),
'FontSize', 6, 'FontWeight', 'bold');

```

## Communities Detection

USA

```

% DETECTION

Comm2USA=GCMoDu1Max(Adj_USA) % Modularity method
USAcusters = Comm2USA;

% PLOT 1
figure,
gscatter(BC_n_USA,D_n_USA,USAcusters);
% for i=1:size(BC_USA,1)
%     text(BC_n_USA(i), D_n_USA(i), table2array(G_simp_USA.Nodes(i,1)), 'FontSize', 5);
% end
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Degree vs Betweenness Distribution for the USA', 'FontSize', 8)

```

```

% PLOT 2 : HISTOGRAM
nb = categorical(USAcusters,[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30],{'Label 1','Label 2','Label 3','Label 4','Label 5','Label 6','Label 7','Label 8','Label
9','Label 10','Label 11','Label 12','Label 13','Label 14','Label 15','Label 16','Label 17','Label
18','Label 19','Label 20','Label 21','Label 22','Label 23','Label 24','Label 25','Label 26','Label
27','Label 28','Label 29','Label 30'}))

```

```

figure,
b=histogram(nb,'BarWidth',0.3,'FaceColor','r');
xlabel('Cluster label')
ylabel('Number of Nodes')
ylim([0 350]);
title(' Label Repartition for communities in the USA (30 classes)',FontSize=8)
y = b.Values;
text(1:b.NumDisplayBins, y+15, string(y));

```

```

% PLOT 3 : GRAPH
[indexlabel1USA, indexlabel2USA, indexlabel3USA, LatClass1USA, LongClass1USA, LatClass2USA,
LongClass2USA, LatClass3USA, LongClass3USA] = PlotComm(G_USA, USAclusters, USAData);
TableLabel = [G_USA.Nodes, array2table(string(USAclusters))]; % cluster
Table4 = TableLabel(strcmp(TableLabel.Var1, '4'),:); % cluster
Table5 = TableLabel(strcmp(TableLabel.Var1, '5'),:); % cluster
listelabel4 = Table4(:,1); % cluster
listelabel5 = Table5(:,1); % cluster
indexlabel4USA = ind2(table2array(listelabel4), TableLabel); % cluster
indexlabel5USA = ind2(table2array(listelabel5), TableLabel); % cluster
listelabel4.Properties.VariableNames(1) = {'id'} % cluster
listelabel5.Properties.VariableNames(1) = {'id'} % cluster
I4 = innerjoin(listelabel4,USAData); % cluster
I5 = innerjoin(listelabel5,USAData); % cluster
LatClass4USA = table2array(I4(:,{'Lat'})); % cluster
LongClass4USA = table2array(I4(:,{'Lon'}));
LatClass5USA = table2array(I5(:,{'Lat'})); % cluster
LongClass5USA = table2array(I5(:,{'Lon'}));

figure,
h = plot(G_USA)

```

```

highlight(h,indexlabel1USA,'NodeColor','g')
highlight(h,indexlabel2USA,'NodeColor','r')
highlight(h,indexlabel3USA,'NodeColor','b')
highlight(h,indexlabel4USA,'NodeColor','y')
highlight(h,indexlabel5USA,'NodeColor','m')
h.EdgeColor = 'black';
title('Airline network by communities in the USA','FontSize',8)

```

```

% PLOT 4 : GEOPLOT
figure,
geoplot(LatClass1USA,LongClass1USA,'g*')
hold on
geoplot(LatClass2USA,LongClass2USA,'r*')
geoplot(LatClass3USA,LongClass3USA,'b*')
geoplot(LatClass4USA,LongClass4USA,'y*')
geoplot(LatClass5USA,LongClass5USA,'m*')
geolimits([0 75],[-160 -50]) % USA
geobasemap streets

```



```
hold off
title('Airports by communities in the USA')
```

China

```
% DETECTION

Comm2China=GCMoDuLMax(Adj_China) % Modularity method
Chinaclusters = Comm2China;

% PLOT 1
figure,
gscatter(BC_n_China,D_n_China,Chinaclusters);
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Degree vs Betweenness Distribution for China','FontSize',8)

% PLOT 2 : HISTOGRAM
nb = categorical(Chinaclusters,[1 2 3 4 5],{'Label 1','Label 2','Label 3','Label 4','Label 5'})
figure,
b=histogram(nb,'BarWidth',0.3,'FaceColor','r');
xlabel('Cluster label')
ylabel('Number of Nodes')
ylim([0 70]);
title(' Label Repartition for communities in China (5 classes)','FontSize=8)
y = b.Values;
text(1:b.NumDisplayBins, y+3, string(y));

% PLOT 3 : GRAPH
[indexlabel1China, indexlabel2China, indexlabel3China, LatClass1China, LongClass1China,
LatClass2China, LongClass2China, LatClass3China, LongClass3China] = PlotComm(G_China, Chinaclusters,
ChinaData);

figure,
h = plot(G_China)
highlight(h,indexlabel1China,'NodeColor','g')
highlight(h,indexlabel2China,'NodeColor','r')
highlight(h,indexlabel3China,'NodeColor','blue')
h.EdgeColor = 'black';
title('Airline network by communities in China','FontSize',8)

% PLOT 4 : GEOPLOT
figure,
geoplot(LatClass1China,LongClass1China,'g*')
hold on
geoplot(LatClass2China,LongClass2China,'r*')
geoplot(LatClass3China,LongClass3China,'blue*')
geolimits([0 60],[50 150]) % China
geobasemap streets
hold off
```

```
title('Airports by communities in China')
```

UK

```
% DETECTION

Comm2UK=GCModulMax(Adj_UK) % Modularity method
UKclusters = Comm2UK;

% PLOT 1
figure,
gscatter(BC_n_UK,D_n_UK,UKclusters);
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Degree vs Betweenness Distribution for the UK','FontSize',8)

% PLOT 2 : HISTOGRAM
nb = categorical(UKclusters,[1 2 3 4 5 6 7],{'Label 1','Label 2','Label 3','Label 4','Label 5','Label 6','Label 7'})
figure,
b=histogram(nb,'BarWidth',0.3,'FaceColor','r');
xlabel('Cluster label')
ylabel('Number of Nodes')
ylim([0 25]);
title(' Label Repartition for communities in the UK (7 classes)','FontSize=8)
y = b.Values;
text(1:b.NumDisplayBins, y+2, string(y));

% PLOT 3 : GRAPH
[indexlabel1UK, indexlabel2UK, indexlabel3UK, LatClass1UK, LongClass1UK, LatClass2UK, LongClass2UK, LatClass3UK, LongClass3UK] = PlotComm(G_UK, UKclusters, UKData);

TableLabel = [G_UK.Nodes, array2table(string(UKclusters))]; % cluster
Table4 = TableLabel(strcmp(TableLabel.Var1, '4'),:); % cluster
Table5 = TableLabel(strcmp(TableLabel.Var1, '5'),:); % cluster
Table6 = TableLabel(strcmp(TableLabel.Var1, '6'),:); % cluster
Table7 = TableLabel(strcmp(TableLabel.Var1, '7'),:); % cluster
listelabel4 = Table4(:,1); % cluster
listelabel5 = Table5(:,1); % cluster
listelabel6 = Table6(:,1); % cluster
listelabel7 = Table7(:,1); % cluster
indexlabel4UK = ind2(table2array(listelabel4), TableLabel); % cluster
indexlabel5UK = ind2(table2array(listelabel5), TableLabel); % cluster
indexlabel6UK = ind2(table2array(listelabel6), TableLabel); % cluster
indexlabel7UK = ind2(table2array(listelabel7), TableLabel); % cluster
listelabel4.Properties.VariableNames(1) = {'id'} % cluster
listelabel5.Properties.VariableNames(1) = {'id'} % cluster
listelabel6.Properties.VariableNames(1) = {'id'} % cluster
listelabel7.Properties.VariableNames(1) = {'id'} % cluster
I4 = innerjoin(listelabel4,UKData); % cluster
I5 = innerjoin(listelabel5,UKData); % cluster
I6 = innerjoin(listelabel6,UKData); % cluster
```

```

I7 = innerjoin(listelabel17,UKData); % cluster
LatClass4UK = table2array(I4(:,{'Lat'})); % cluster
LongClass4UK = table2array(I4(:,{'Lon'}));
LatClass5UK = table2array(I5(:,{'Lat'})); % cluster
LongClass5UK = table2array(I5(:,{'Lon'}));
LatClass6UK = table2array(I6(:,{'Lat'})); % cluster
LongClass6UK = table2array(I6(:,{'Lon'}));
LatClass7UK = table2array(I7(:,{'Lat'})); % cluster
LongClass7UK = table2array(I7(:,{'Lon'}));

figure,
h = plot(G_UK)
highlight(h,indexlabel1UK,'NodeColor','g')
highlight(h,indexlabel2UK,'NodeColor','r')
highlight(h,indexlabel3UK,'NodeColor','b')
highlight(h,indexlabel4UK,'NodeColor','y')
highlight(h,indexlabel5UK,'NodeColor','m')
highlight(h,indexlabel6UK,'NodeColor','cyan')
highlight(h,indexlabel7UK,'NodeColor','black')
h.EdgeColor = 'black';
title('Airline network by communities in the UK','FontSize',8)

% PLOT 4 : GEOPLOT
figure,
geoplot(LatClass1UK,LongClass1UK,'g*')
hold on
geoplot(LatClass2UK,LongClass2UK,'b*')
geoplot(LatClass3UK,LongClass3UK,'r*')
geoplot(LatClass4UK,LongClass4UK,'y*')
geoplot(LatClass5UK,LongClass5UK,'m*')
geoplot(LatClass6UK,LongClass6UK,'cyan*')
geoplot(LatClass7UK,LongClass7UK,'black*')
geolimits([45 63],[-15 10]) % UK
geobasemap streets
hold off
title('Airports by communities in the UK')

```

Australia

```

% DETECTION

Comm2Australia=GCMoModulMax(Adj_Australia) % Modularity method
Australiaclusters = Comm2Australia;

% PLOT 1
figure,
gscatter(BC_n_Australia,D_n_Australia,Australiaclusters);
xlabel('Normalized Betweenness')
ylabel('Normalized Degree')
title('Degree vs Betweenness Distribution for Australia','FontSize',8)

% PLOT 2 : HISTOGRAM

```

```

nb = categorical(Australiaclusters,[1 2 3 4 5 6 7],{'Label 1','Label 2','Label 3','Label 4','Label 5','Label 6','Label 7'})
figure,
b=histogram(nb,'BarWidth',0.3,'FaceColor','r');
xlabel('Cluster label')
ylabel('Number of Nodes')
ylim([0 50]);
title('Label Repartition for communities in Australia (7 classes)',FontSize=8)
y = b.Values;
text(1:b.NumDisplayBins, y+3, string(y));

```

```

% PLOT 3 : GRAPH
[indexlabel1Australia, indexlabel2Australia, indexlabel3Australia, LatClass1Australia,
LongClass1Australia, LatClass2Australia, LongClass2Australia, LatClass3Australia, LongClass3Australia]
= PlotComm(G_Australia, Australiaclusters, AustraliaData);

TableLabel = [G_Australia.Nodes, array2table(string(Australiaclusters))]; % cluster
Table4 = TableLabel(strcmp(TableLabel.Var1, '4'),:); % cluster
Table5 = TableLabel(strcmp(TableLabel.Var1, '5'),:); % cluster
listelabel4 = Table4(:,1); % cluster
listelabel5 = Table5(:,1); % cluster
indexlabel4Australia = ind2(table2array(listelabel4), TableLabel); % cluster
indexlabel5Australia = ind2(table2array(listelabel5), TableLabel); % cluster
listelabel4.Properties.VariableNames(1) = {'id'} % cluster
listelabel5.Properties.VariableNames(1) = {'id'} % cluster
I4 = innerjoin(listelabel4,AustraliaData); % cluster
I5 = innerjoin(listelabel5,AustraliaData); % cluster
LatClass4Australia = table2array(I4(:,{'Lat'})); % cluster
LongClass4Australia = table2array(I4(:,{'Lon'}));
LatClass5Australia = table2array(I5(:,{'Lat'})); % cluster
LongClass5Australia = table2array(I5(:,{'Lon'}));

figure,
h = plot(G_Australia)

highlight(h,indexlabel1Australia,'NodeColor','g')
highlight(h,indexlabel2Australia,'NodeColor','r')
highlight(h,indexlabel3Australia,'NodeColor','b')
highlight(h,indexlabel4Australia,'NodeColor','y')
highlight(h,indexlabel5Australia,'NodeColor','m')
h.EdgeColor = 'black';
title('Airline network by communities in Australia','FontSize',8)

% PLOT 4 : GEOPLOT
figure,
geoplot(LatClass1Australia,LongClass1Australia,'g*')
hold on
geoplot(LatClass2Australia,LongClass2Australia,'b*')
geoplot(LatClass3Australia,LongClass3Australia,'r*')
geoplot(LatClass4Australia,LongClass4Australia,'y*')

```

```
geoplot(LatClass5Australia,LongClass5Australia,'m*')
geolimits([-50 0],[120 150]) % Australia
geobasemap streets
hold off
title('Airports by communities in Australia')
```

# FUNCTION PART

## Function **ExtractLocation**

```
function [CountryData, CountryLat, CountryLong] = ExtractLocation (Country, Dataset)

% OUTPUT
% CountryData : Airports of the country with its position (table)
% CountryLat : Latitude of all airports of the country (array)
% CountryLong : Latitude of all airports of the country (array)

% INPUT
% Country : Country to study (string)
% Dataset : Here GeoData

CountryData = Dataset(strcmp(Dataset.country, Country),:);
CountryLat = table2array(CountryData(:,{'Lat'}));
CountryLong = table2array(CountryData(:,{'Lon'}));
end
```

## Function **Cleaning**

```
function [CountrySource, CountryTarget, CountryWeights, lines] = cleaning(Country, Dataset, CountryData)

% OUTPUT
% CountrySource : Sources IDs (array)
% CountryTarget : Targets IDs (array)
% CountryWeights : Weights of the lines (array)
% NewLines : Lines for the country, with the return (table)

% INPUT
% Country : Country to study (string)
% Dataset : Here FlightData
% CountryData : Extract dataset for the country studied

%% STORE SOURCES AND TARGETS FROM FIGHTDATA
CountryLines = Dataset(strcmp(Dataset.SourceCountry, Country),:); % Lines with Source = Country
CountryLines = CountryLines(strcmp(CountryLines.TargetCountry, Country),:); % Lines with Source = Country and Target = Country

CountryLines.TimeSeries = string(CountryLines.TimeSeries);
CountryLines = CountryLines(strcmp(CountryLines.TimeSeries, '01-Jul-2009'),:); % Lines with Source = Country and Target = Country

CountrySource = table2array(CountryLines(:,{'Source'})); % ID Lines Source
CountryTarget = table2array(CountryLines(:,{'Target'})); % ID Lines Target
CountryWeights = table2array(CountryLines(:,{'Weight'})); % ID Lines Weights
```

```

%% SEARCHING FOR SOURCES AND TARGET WHOSE POSITIONS ARE MISSING IN GEODATA
SourceMissing = ismember(CountrySource, CountryData.id);
IndexSourceMissing = find(SourceMissing == 0); % index of the ID sources missing
TargetMissing = ismember(CountryTarget, CountryData.id);
IndexTargetMissing = find(TargetMissing==0); % index of the ID targets missing
Remove = sort([IndexSourceMissing ; IndexTargetMissing]); % Airports index to remove

for i = size(Remove,1):-1:1
    CountrySource(Remove(i))=[]; % New CountrySource
    CountryTarget(Remove(i))=[]; % New CountryTarget
    CountryWeights(Remove(i))=[]; % New CountryWeights
end

lines = table(CountrySource, CountryTarget, CountryWeights);
lines = sortrows(lines, [1 2 3]); % Sort by weight

end

```

#### Function **TableDegree**

```

function [TableDegreeCountry] = TableDegree(G_Country)

% OUTPUT
% TableDegreeCountry : Table with nodes and degrees associated for a country (table)

% INPUT
% G_country : Country's graph (graph)

NodesNamesCountry = G_Country.Nodes;
NodesDegreeCountry = array2table(indegree(G_Country)+outdegree(G_Country));

TableDegreeCountry = [NodesNamesCountry,NodesDegreeCountry];

end

```

#### Function **PositionsLines**

```

function [CountryLatSource, CountryLongSource, CountryLatTarget, CountryLongTarget] =
PositionsLines(CountrySource, CountryTarget, CountryData)

% OUTPUT
% CountryLatSource : Sources IDs Latitudes (array)
% CountryLongSource : Targets IDs Longitudes (array)
% CountryLatTarget : Sources IDs Latitudes (array)
% CountryLongTarget : Targets IDs Longitudes (array)

% INPUT
% CountrySource : Sources IDs (array)
% CountryTarget : Targets IDs (array)
% CountryData : Extract dataset for the country studied

```

```

    IndexCountrySource = ind(CountrySource, CountryData); % Index of ID Sources in GeoData for the
positions
    IndexCountryTarget = ind(CountryTarget, CountryData); % Index of ID Targets in GeoData for the
positions

    CountryLatSource = table2array(CountryData(IndexCountrySource,{'Lat'})); % Latitude linked with
the Source Index
    CountryLongSource = table2array(CountryData(IndexCountrySource,{'Lon'})); % Longitude linked with
the Source Index
    CountryLatTarget = table2array(CountryData(IndexCountryTarget,{'Lat'})); % Latitude linked with
the Target Index
    CountryLongTarget = table2array(CountryData(IndexCountryTarget,{'Lon'})); % Longitude linked with
the Target Index

end

```

Function **WDD (Weighted Degree Distribution)**

```

function [y, rank, weights, rsq, p] = WDD(G, inf, sup)

% OUTPUT
% y : Sources IDs (array)
% rank : Targets IDs (array)
% weights : Log Weights of the lines sorted (array)
% rsq : R2 (double)
% p : (slope, intercept) of the linear regression

% INPUT
% G : Graph of the country (graph)
% inf : lower bound regression (int)
% sup : upper bound regression (int)
% TableDegree : Degrees associated with each node (table)

WD = [];
for n=1:size(G.Nodes,1)
    edges_n = G.Edges([inedges(G,n); outedges(G,n)],:); % edges
    WD = [WD; sum(table2array(edges_n(:,2)))]; % weights
end

WD = log10(WD); % log scale
WD(isinf(WD)) = 1;
WD = table(WD);
WD = sortrows(WD,1,'descend');
weights = table2array(WD(:,1));

rank = 1:size(weights,1);

p = polyfit(rank,weights,1); % (slope, intercept)
p = polyfit(rank(inf:sup),weights(inf:sup),1); % (slope, intercept)
y = p(1)*rank+p(2); % linear regression

```



```

% R^2 Calculation
y = reshape(y,[length(y),1]);
yresid = weights - y;
SSresid = sum(yresid.^2);
SStotal = (length(weights)-1) * var(weights);
rsq = 1 - SSresid/SStotal;
end

```

Function **WDCD (Weighted Degree Cumulative Distribution)**

```

function [Dwz, C] = WDCD(TableDegree, G)

% OUTPUT
% Dwz : Weighted Degree (array)
% C : Probability computed (array)

% INPUT
% G : Graph of the country (graph)
% TableDegree : Degrees associated with each node (table)

Dw = [];
for n=1:size(G.Nodes,1)
    edges_n = G.Edges([inedges(G,n); outedges(G,n)],:); % edges
    Dw = [Dw; sum(table2array(edges_n(:,2)))]; % weights
end
Dwz = Dw/length(Dw);

Node = table2array(TableDegree(:,2)); % degree
C = zeros(length(Dwz),1);
for j=1:length(Dwz)
    for i=1:length(Node)
        if Node(i)>Dwz(j)
            C(j)=C(j)+1;
        end
    end
end
C = C/length(Node); % probability y axis

end

```

Function **Assortativity**

```

function r = assortativity(CIJ)

% Assortativity coefficient
%
% r = assortativity(CIJ);

```

```

%
% The assortativity coefficient is a correlation coefficient between the
% degrees of all nodes on two opposite ends of a link. A positive
% assortativity coefficient indicates that nodes tend to link to other
% nodes with the same or similar degree.
%
% Inputs:    CIJ,        binary directed/undirected connection matrix
% Outputs:   r,          assortativity

[id,od,deg] = degrees_dir(CIJ);
[i,j] = find(CIJ>0);
K = length(i);
    for k=1:K
        degi(k) = deg(i(k));
        degj(k) = deg(j(k));
    end

    r = (sum(degi.*degj)/K - (sum(0.5*(degi+degj))/K)^2)/(sum(0.5*(degi.^2+degj.^2))/K -
(sum(0.5*(degi+degj))/K)^2);
end

function [id,od,deg] = degrees_dir(CIJ)

% Node degree is the number of links connected to the node. The indegree
% is the number of inward links and the outdegree is the number of
% outward links.
%
% Input:    CIJ,        directed (binary/weighted) connection matrix
%
% Output:   id,         node indegree
%           od,         node outdegree
%           deg,        node degree (indegree + outdegree)
%
% Notes: Inputs are assumed to be on the columns of the CIJ matrix.
%        Weight information is discarded.

% ensure CIJ is binary...
CIJ = double(CIJ~=0);

% compute degrees
id = sum(CIJ,1);    % indegree = column sum of CIJ
od = sum(CIJ,2)';  % outdegree = row sum of CIJ
deg = id+od;       % degree = indegree+outdegree
end

```

Function **NBND (Normalized Betweenness VS Normalized Degree)**

```
% Weights integreted in G_simp OK
```

```

function [BC, BC_n, D, D_n] = nbnd(G)

% OUTPUT
% BC : Betweenness centrality (array)
% BC_n : Normalized Betweenness centrality (array)
% D : Degrees (array)
% D_n : % Normalized Degrees (array)

% INPUT
% G : Graph of the country (graph)

BC = centrality(G,'betweenness'); % Betweenness centrality
D = indegree(G)+outdegree(G); % Degrees
n = numnodes(G); % Number of nodes
BC_n = 2*BC./((n-2)*(n-1)); % Normalized Betweenness
D_n = 2*(D./((n-2)*(n-1))); % Normalized Degree %NodesDegree1

end

```

#### Function **GCModuMax (Modularity)**

```

function VV= GCModuMax(A)
% Modularity Maximization community detection
% INPUT
% A: Adjacency matrix of graph
%
% OUTPUT
% VV: N-ny-1 matrix, VV(n) is teh cluster to which node n belongs

N=length(A);
W=PermMat(N); % permute the graph node labels
A=W*A*W';
[VV,Q] = fast_newman(A);
VV=W'*VV; % unpermute the graph node labels

end

function [com,Q] = fast_newman(adj)

% Modularity optimisation based on a greedy agglomerative method
%
% Input
% - adj: (symmetrical) adjacency matrix
%
% Output
% - com: communities (listed for each node)
% - Q : modularity value of the given partition

% Set initial communities with one node per community
cur_com = [1:length(adj)]';

```

```

% Initialise best community to current value
com = cur_com;
% Compute initial community matrix
e = get_community_matrix(adj,com);
% Lines and columns sum (speed optimisation)
ls = sum(e,2);
cs = sum(e,1);
% Initialise best known and current Q values
cur_Q = trace(e) - sum(sum(e^2));
Q = cur_Q;
% Loop until no more aggregation is possible
while length(e) > 1
    % Print progress
    %fprintf('Loop %d/%d...',length(adj)-length(e)+1,length(adj));
    %tic
    % Best Q variation
    loop_best_dQ = -inf;
    % For all the pairs of nodes that could be merged
    can_merge = false;
    for i=1:length(e)
        for j=i+1:length(e)
            % If they share edges
            if e(i,j) > 0
                % Compute the variation in Q
                dQ = 2 * (e(i,j) - ls(i)*cs(j));
                % If best variation, then keep track of the pair
                if dQ > loop_best_dQ
                    loop_best_dQ = dQ;
                    best_pair = [i,j];
                    can_merge = true;
                end
            end
        end
    end
    if ~can_merge
        disp('!!! Graph with isolated communities, no more merging possible !!!');
        break;
    end
    % Merge the pair of clusters maximising Q
    best_pair = sort(best_pair);
    for i=1:length(cur_com)
        if cur_com(i) == best_pair(2)
            cur_com(i) = best_pair(1);
        elseif cur_com(i) > best_pair(2)
            cur_com(i) = cur_com(i) - 1;
        end
    end
    % Update community matrix
    % Slow way (for precision comparison)
    %e = get_community_matrix(adj,com);
    % Faster way
    e(best_pair(1),:) = e(best_pair(1),:) + e(best_pair(2),:);

```

```

    e(:,best_pair(1)) = e(:,best_pair(1)) + e(:,best_pair(2));
    e(best_pair(2),:) = [];
    e(:,best_pair(2)) = [];
    % Update lines/columns sum
    ls(best_pair(1)) = ls(best_pair(1)) + ls(best_pair(2));
    cs(best_pair(1)) = cs(best_pair(1)) + cs(best_pair(2));
    ls(best_pair(2)) = [];
    cs(best_pair(2)) = [];
    % Update Q value
    cur_Q = cur_Q + loop_best_dQ;
    % Check consistency
    %eqQ = trace(e) - sum(sum(e*e));
    %if Q ~= eqQ
    %    fprintf('Warning: found Q=%d, should be Q=%d. Diff = %d\n',Q,eqQ,abs(Q-eqQ));
    %end
    % If new Q is better, save current partition
    if cur_Q > Q
        Q = cur_Q;
        com = cur_com;
    end
    %fprintf(' completed in %f(s)\n',toc);
end
end

```

```

function [e] = get_community_matrix(adj,com)
% Create the community matrix from a list giving for each node its community
%
% Input
% - adj: symmetrical (binary or weighted) adjacency matrix
% - com: community list
%
% Output
% - e: community adjacency matrix where e(i,j) is half the sum of the
%       weights of edges connecting communities i and j except for the
%       e(i,i) elements which contain the full sum of the weights of edges
%       connecting community i to itself.

% Number of communities
nc = length(unique(com));
% Initialise adjacency matrix list
e = zeros(nc,nc);
% Create edges and normalise values by dividing by the sum of all edges
m = 0;
for i=1:length(adj)
    for j=i:length(adj)
        if adj(i,j) ~= 0
            ic = com(i);
            jc = com(j);
            if ic == jc

```

```

        e(ic,ic) = e(ic,ic) + adj(i,j);
    else
        e(ic,jc) = e(ic,jc) + (0.5 * adj(i,j));
        e(jc,ic) = e(ic,jc);
    end
    m = m + adj(i,j);
end
end
end
e = e / m;
end

```

#### Function **PerMat**

```

function W=PerMat(N)
% function W=PerMat(N)
%
% Creates an N-by-N permutation matrix W
%
% INPUT
% N      size of permutation matrix
%
% OUTPUT
% W      the permutation matrix: N-by-N matrix with exactly one 1 in every
%        row and column, all other elements equal to 0
W=zeros(N,N);
q=randperm(N);
for n=1:N;
    W(q(n),n)=1;
end
end

```

#### Function **PlotComm**

```

function [indexlabel1, indexlabel2, indexlabel3, LatClass1, LongClass1, LatClass2, LongClass2,
LatClass3, LongClass3] = PlotComm(G, clusters, CountryData)

% OUTPUT
% indexlabel1 (array)
% indexlabel2 (array)
% indexlabel3 (array)
% LatClass1 : Latitude of airports in Class 1 (array)
% LatClass2 : Latitude of airports in Class 2 (array)
% LatClass3 : Latitude of airports in Class 3 (array)
% LongClass1 : Longitude of airports in Class 1 (array)
% LongClass2 : Longitude of airports in Class 2 (array)
% LongClass3 : Longitude of airports in Class 3 (array)

% INPUT

```

```

% G : Graph of the country (graph)
% clusters : List of clusters for each node (array)
% CountryData : Extract dataset for the country studied

TableLabel = [G.Nodes, array2table(string(clusters))];
Table1 = TableLabel(strcmp(TableLabel.Var1, '1'),:);
Table2 = TableLabel(strcmp(TableLabel.Var1, '2'),:);
Table3 = TableLabel(strcmp(TableLabel.Var1, '3'),:);

listelabel1 = Table1(:,1);
listelabel2 = Table2(:,1);
listelabel3 = Table3(:,1);

indexlabel1 = ind2(table2array(listelabel1), TableLabel);
indexlabel2 = ind2(table2array(listelabel2), TableLabel);
indexlabel3 = ind2(table2array(listelabel3), TableLabel);

listelabel1.Properties.VariableNames(1) = {'id'};
listelabel2.Properties.VariableNames(1) = {'id'};
listelabel3.Properties.VariableNames(1) = {'id'};

I1 = innerjoin(listelabel1,CountryData);
I2 = innerjoin(listelabel2,CountryData);
I3 = innerjoin(listelabel3,CountryData);

LatClass1 = table2array(I1(:,{'Lat'}));
LongClass1 = table2array(I1(:,{'Lon'}));
LatClass2 = table2array(I2(:,{'Lat'}));
LongClass2 = table2array(I2(:,{'Lon'}));
LatClass3 = table2array(I3(:,{'Lat'}));
LongClass3 = table2array(I3(:,{'Lon'}));

end

```

#### Function **CPS (Core Periphery Structure)**

```

function [Other] = CPS(TableDegree, G)

% OUTPUT
% G : Graph of the country (graph)
% TableDegree : Table with the degrees for each node (table)

% INPUT
% Other : number of connections with a higher degree node

TableDegree2 = sortrows(TableDegree,2,'descend'); % Sorted by descendent degree
TableDegree2.Properties.VariableNames(2) = {'Degree'};

E0bis = G.Edges; % Edges for the graph unsimplified
E12bis = table2array(E0bis(:,1)); % store just the edges

```

```

P1 = E12bis(:,1); % source
P2 = E12bis(:,2); % target
edges = table(P1,P2); % Edges Table
edges.Properties.VariableNames(1) = {'Name'};

% Join the degree to each node of the lines
Y = innerjoin(edges,TableDegree2);
Y.Properties.VariableNames(3) = {'DegreeName'};
TableDegree2.Properties.VariableNames(1) = {'P2'};
Y = innerjoin(Y,TableDegree2);
Y.Properties.VariableNames(4) = {'DegreeP2'};

% Reference : right columns
L=zeros(size(Y,1),1);
DegName = table2array(Y(:,3));
DegP2 = table2array(Y(:,4));
for i = 1:length(DegName)
    if DegName(i)>DegP2(i) % connection with a node who has a higher degree
        L(i)=1;
    end
end

% Remove the lines where the reference node is not connectd with a higher
% degree node
for i=length(L):-1:1
    if L(i)==0
        Y([i],:) = [];
    end
end

% Count the number of connections for those concerned
S = groupcounts(Y,'P2'); % Right column : Other concerned nodes
S = removevars(S,{'Percent'});

% Select nodes without connections with high degree node
Other = outerjoin(S,TableDegree2);
Other = removevars(Other,{'P2_S'});
idx = isnan(Other.GroupCount);

% Update the count of connections for those who haven't to 0
NewCount = Other.GroupCount;
for j=1:length(idx)
    if idx(j)==1
        NewCount(j)=0;
    end
end

Other(:,{'GroupCount'})=array2table(NewCount); % update the column
Other = sortrows(Other,{'Degree'},'descend');

end

```



## Function **Index**

```
function [Index] = ind(IDlist,data)
    Index=[];
    for i = 1:size(IDlist,1)
        Index = [Index;find(strcmp(IDlist(i),data.id))];
    end
end

function [Index] = ind2(IDlist,data)
    Index=[];
    for i = 1:size(IDlist,1)
        Index = [Index;find(strcmp(IDlist(i),data.Name))];
    end
end
```