Lecture 8

Direction and Weights (chapter 4)

Uploaded

- Papers: The structure of interurban traffic: a weighted network analysis
  Quantifying Global Migration Networks

- Lab script: Lecture8_Weights_and_Directions_MigrationNets.ipynb

DOI:10.1068/b32128

The structure of interurban traffic: a weighted network analysis

Andrea De Montis
Dipartimento di Ingegneria del Territorio, Sezione Construzioni e Infrastrutture, Universita degli Studi di Sassari, Via De Nicola, 07100 Sassari, Italy; e-mail: andreadm@uniss.it

Marc Barthélemy
School of Informatics, Center for Biocomplexity and Department of Physics, Indiana University, Bloomington, USA; e-mail: marc.barthelemy@cea.fr
CEA-DIF, Département de Physique Théorique et Appliquée, 91680 Bruyères-le-Châtel, France; e-mail: barthelemy@cea.fr

Alessandro Chessa
Dipartimento di Fisica, Università degli Studi di Cagliari, Cittadella Universitaria di Monserrato, 09042 Monserrato, Italy; e-mail: alessandro.chessa@dsf.unica.it

Alessandro Vespignani
School of Informatics, Center for Biocomplexity and Department of Physics, Indiana University, Bloomington, IN 47408, USA; e-mail: alexv@indiana.edu
Received 14 July 2005; in revised form 9 May 2006; published online 9 March 2007

In the present work we use a network approach to study the Sardinian inter-municipal commuting network (SMCN), which describes the everyday work and study-led movements among 375 municipalities in the Italian region of Sardinia. We obtain a weighted network representation in which the vertices correspond to the Sardinian municipalities and the valued edges correspond to the amount of commuting traffic among them.

Along with a complex topological structure (number of links), real networks display a large heterogeneity in the capacity and intensity of the connections.

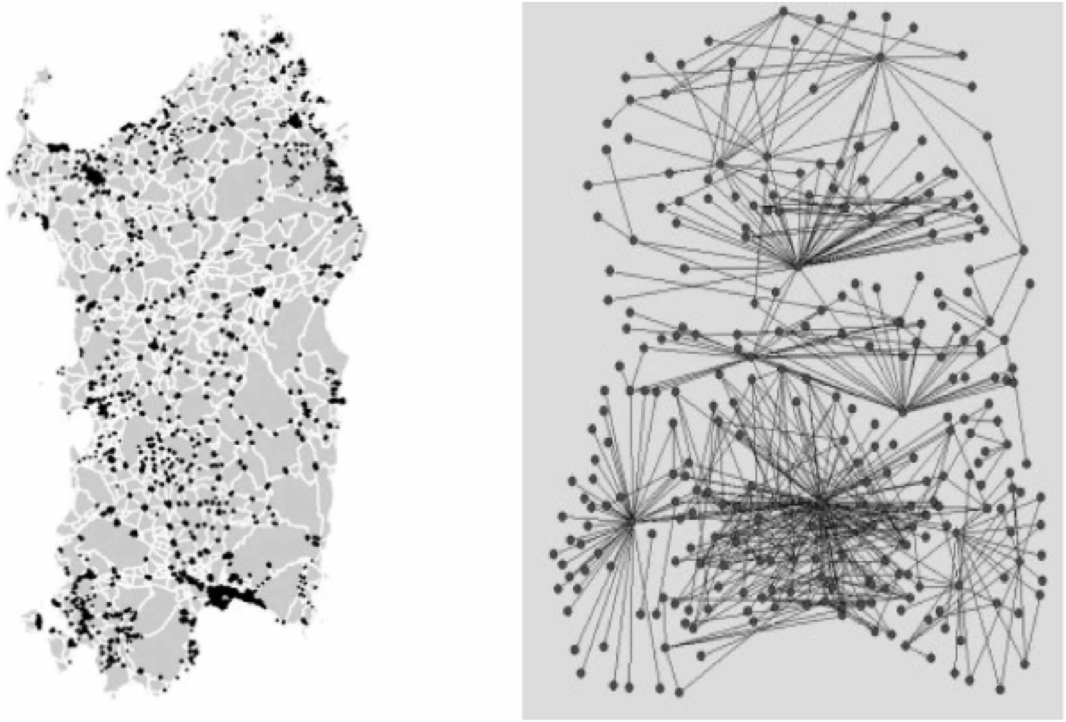
They assign to each edge of the graph a weight proportional to the intensity or capacity of the connections among the various elements of the network.

They define metrics combining weighted and topological observables and characterize the heterogeneity of the actual strength of edges and vertices.

Data

- The commuting network from Sardinia, Italy
- *N=* 375 vertices denoting airports and *E =8,124* edges accounting for the presence of a direct flight connection.
- <k> = 2E/N = 43.33, <l> = 2

Note the maximum
Possible E=140,250



(a) (b)

Figure 1. Geographical versus topological representation of the Sardinian intermunicipal community network: the nodes (points) correspond to the towns, while the links correspond to a flow value larger than fifty commuters between two towns.

Network Constructions

The elements on the principal diagonal (a_{ii}) are set equal to 0, since intramunicipal commuting movements are not considered here. Off-diagonal terms a_{ij} are equal to 1 in the presence of any nonzero flow between i and j ($i \rightarrow j$ or $j \rightarrow i$) and are equal to 0 otherwise. The adjacency matrix is then symmetric, $a_{ij} = a_{ji}$, and describes regular bidirectional displacements among the municipalities.

we thus construct the symmetric weighted adjacency matrix $\mathbf{W}$, in which the elements w_{ij} are computed as the sum of the $i \rightarrow j$ and $j \rightarrow i$ flows between the corresponding municipalities (per day).

Degree Distribution

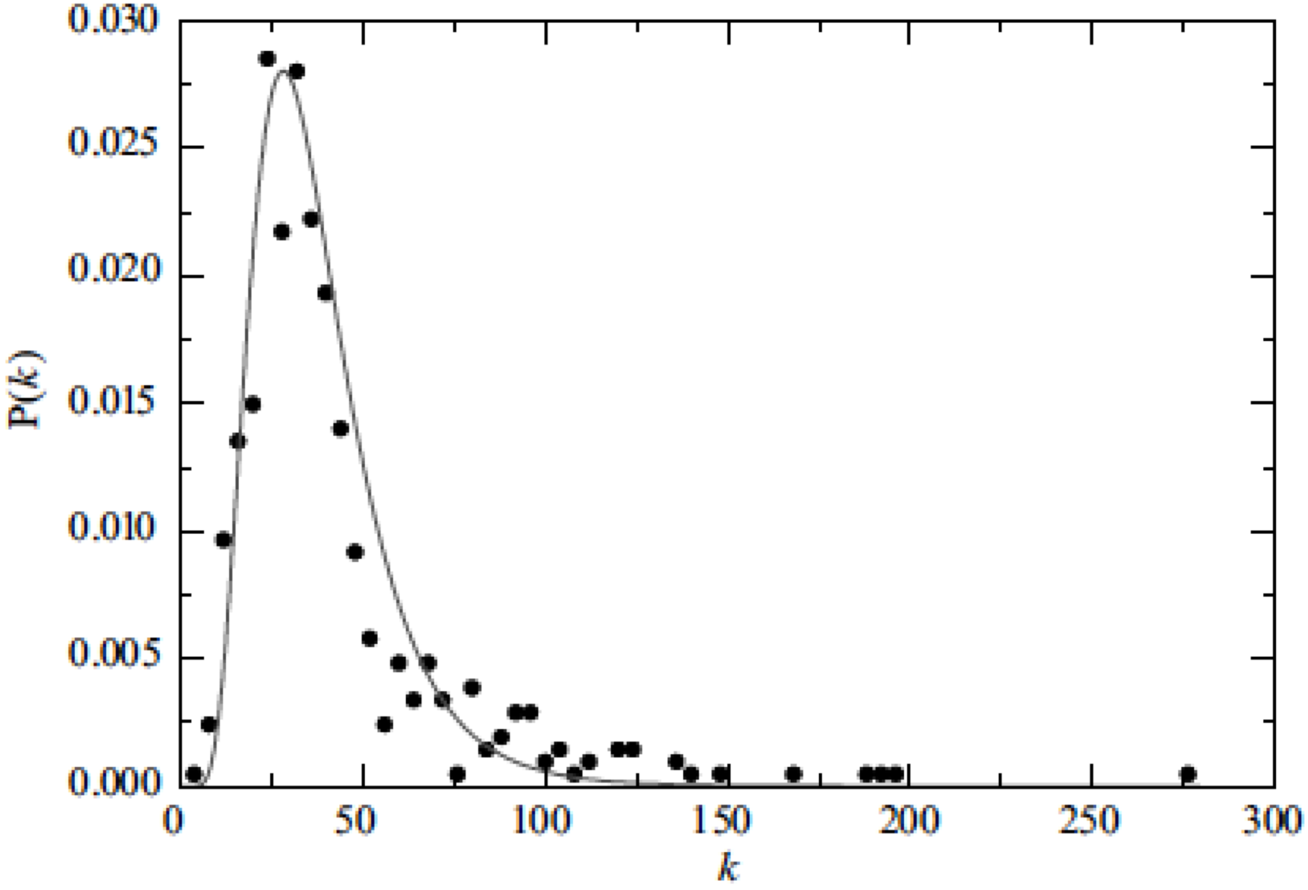


Figure 2. Plot of the probability distribution of the degree. The line is a lognormal fit.

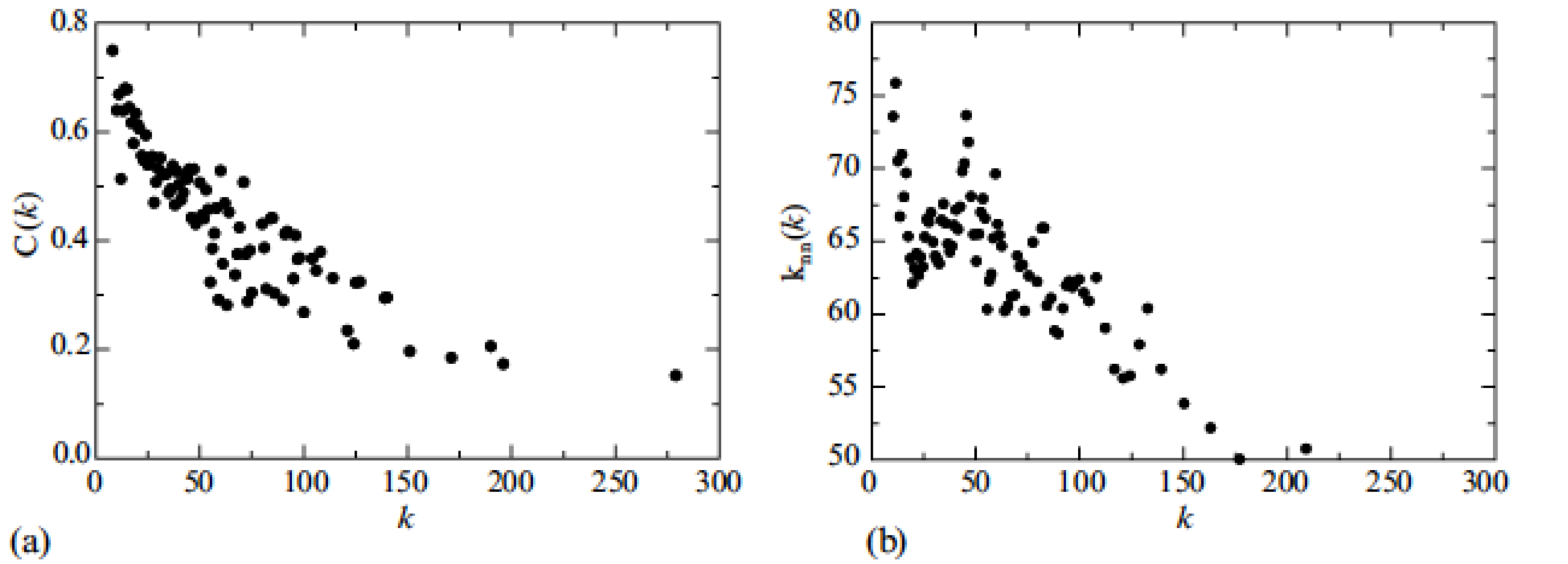


Figure 3. (a) Scatterplot of the clustering coefficient versus degree; (b) assortativity of the Sardinian intermunicipal commuting network, showing a slight disassortative behavior.

$$C(k) = \frac{1}{NP(k)} \sum_{i \mid k_i = k} C(i) \ ,$$

$$C(i) = \frac{2E(i)}{k_i(k_i - 1)}$$

$$k_{nn}(i) = \frac{1}{k_i} \sum_{j \in V(i)} k_j \ ,$$

where $V(i)$ denotes the set of neighbors of i.

networkx.algorithms.assortativity.average_neighbor_degree

`average_neighbor_degree(G, source='out', target='out', nodes=None, weight=None)` [source]

Returns the average degree of the neighborhood of each node.

The average neighborhood degree of a node `i` is

$$k_{nn,i} = \frac{1}{|N(i)|} \sum_{j \in N(i)} k_j$$

where `N(i)` are the neighbors of node `i` and `k_j` is the degree of node `j` which belongs to `N(i)`. For weighted graphs, an analogous measure can be defined [1],

$$k_{nn,i}^{w} = \frac{1}{s_i} \sum_{j \in N(i)} w_{ij} k_j$$

where `s_i` is the weighted degree of node `i`, `w_{ij}` is the weight of the edge that links `i` and `j` and `N(i)` are the neighbors of node `i`.

This network has N = 375 vertices and E = 8124 edges.

1. What are the parameters of a erdos-renyi graph to model it?
2. What are the resulting $<C_r>$ nd $<l_r >$?
3. Is the SCMN a small world network?

Is the SMCN a small world network?

$$\langle C(k) \rangle = 0.26$$

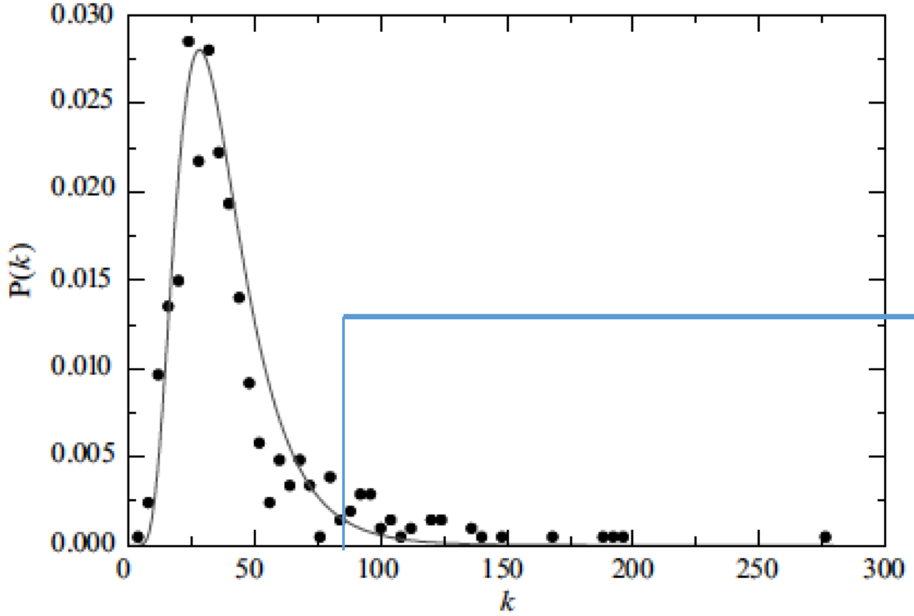
The average shortest path length is <l >= 2.0,

The random CC and l are $<C_r>$ ~ 0.1158 and 1.56

How do we estimate $<C_r>$ and <lr> ?

Answer: This network does not have the property of the Small World networks defined as high Clustering coefficient and small average shortest path length, when compared to the values of a RG with the some size and <k>

Additionally, several of these networks are characterized by the statistical abundance of nodes with a very large degree k; that is, the number of connections to other nodes. For these 'scale-free' networks, the degree probability distribution $P(k)$ spans over a wide range of k values, which signals the appreciable occurrence of large degree nodes, the 'hubs' of the system (O'Kelly, 1998; Taaffe et al, 1996).



Hubs, or nodes with $K>N^{1/2}$

Figure 2. Plot of the probability distribution of the degree. The line is a lognormal fit.

Weights

- For the SMCN the weight w_{ij} of an edge linking municipalities i and j represents the number of commuters in between these two municipalities.

Another relevant quantity that characterizes the traffic is given by the strength of a node, defined as

$$s(i) = \sum_{j \in V(i)} w_{ij} \quad . \tag{7}$$

Reminder: The modeler defines the links and weights when representing the data as a Network.

Same definitions using the adjacency matrix

- Given, the the node degree k_i defined in terms of the Adjacency Matrix which has elements as 1 if there is a link between node i and node j

$$k_i = \sum_j a_{ij}$$

- The weighted degree or strength is defined as:

$$s_i = \sum_{j=1}^{N} a_{ij} w_{ij}.$$

Results: Degree and Strength Distributions

$P(w) \sim w^{-\gamma_w}$ with an exponent $\gamma_W \approx 1.8$ $P(s) \sim s^{-\gamma_s}$ with exponent $\gamma_s \approx 2$

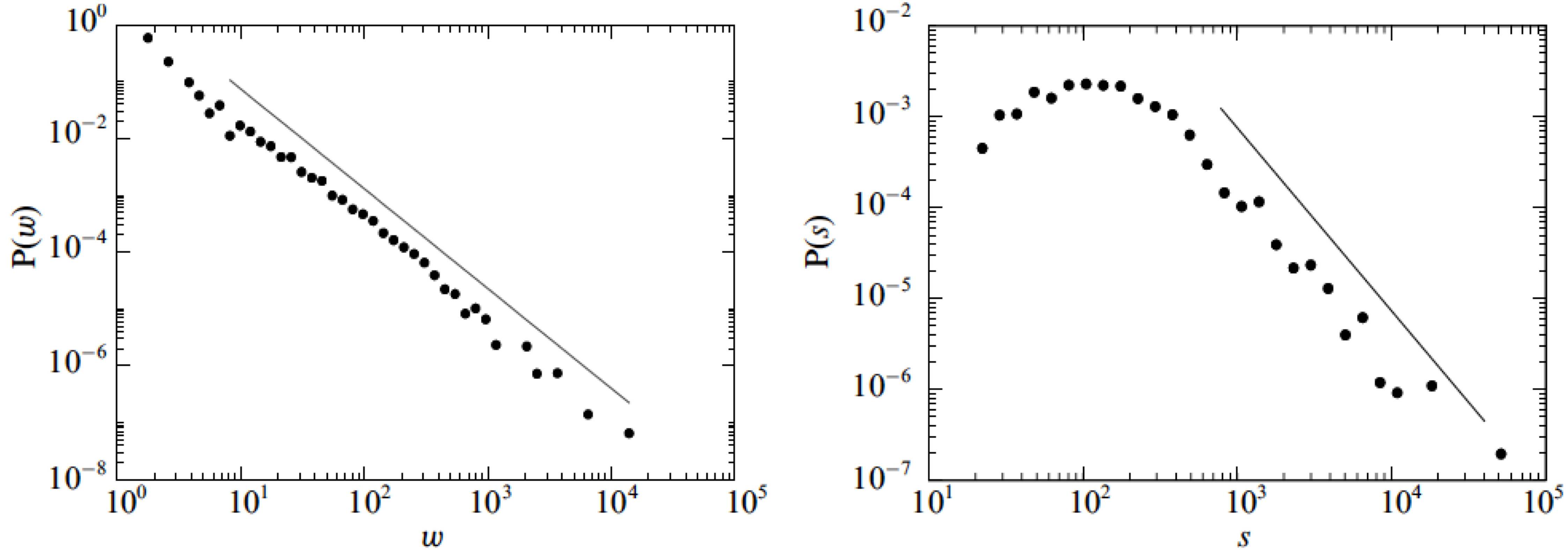
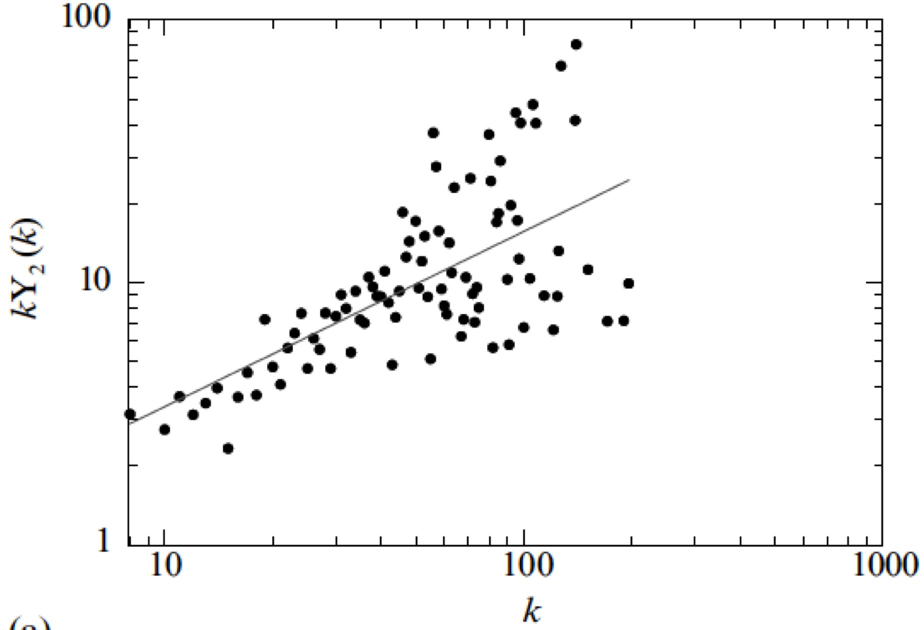


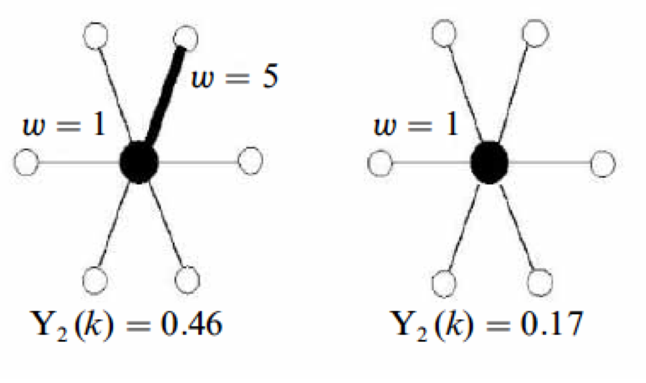
Figure 5. (a) Log – log plot of the probability distribution of the weights (the straight line is a power-law fit with exponent 1.8); (b) log – log plot of the probability distribution of the strength (as a guide to the eye we also plot a power-law fit with exponent 2).

A convenient measure of this is given by the disparity (Barthélemy et al, 2005), defined as

$$Y_2(i) = \sum_j \left(\frac{w_{ij}}{s(i)} \right)^2 . \tag{8}$$



(a)



(b)

Figure 6. (a) Log – log plot of the disparity versus degree for the Sardinian intermunicipal community network; (b) illustration of the disparity for two very different cases. When a few connections dominate, $Y_2(k)$ is of order 1; in contrast, if all connections have the same weight, $Y_2(k)$ is of order $1/k$.

Table 2. Ranking of pairs of Sardinian municipal centers by the weight of their connections.

Rank	Pairs of Sardinian connected municipal centers	w
1	Cagliari – Sassari	13 953
2	Sassari – Olbia	7 246
3	Cagliari – Assemini	4 226
4	Porto Torres – Sassari	3 993
5	Cagliari – Capoterra	3 731

Table 3. Ranking of Sardinian municipalities by their strength.

Rank	Sardinian municipality	s
1	Cagliari	64 834
2	Sassari	21 437
3	Quartu Sant'Elena	18 431
4	Oristano	12 130
5	Selargius	10 084
6	Assemini	7 915
7	Porto Torres	6 886
8	Nuoro	6 834
9	Carbonia	6 616
10	Iglesias	6 479

Flows, topology and demographics

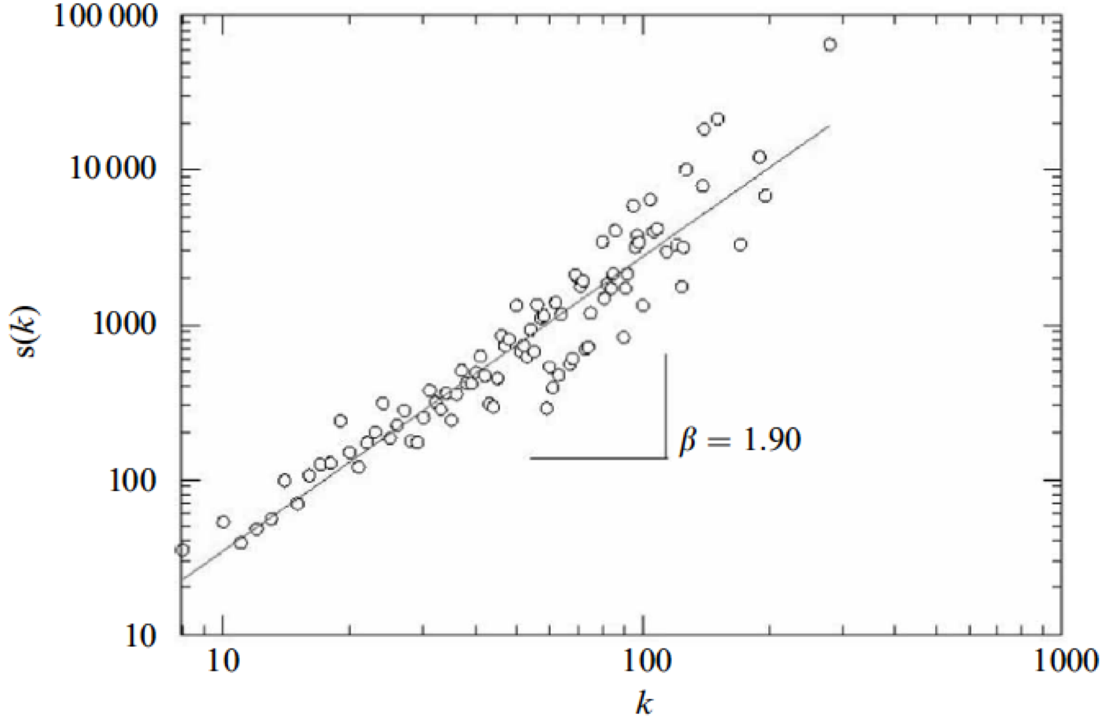
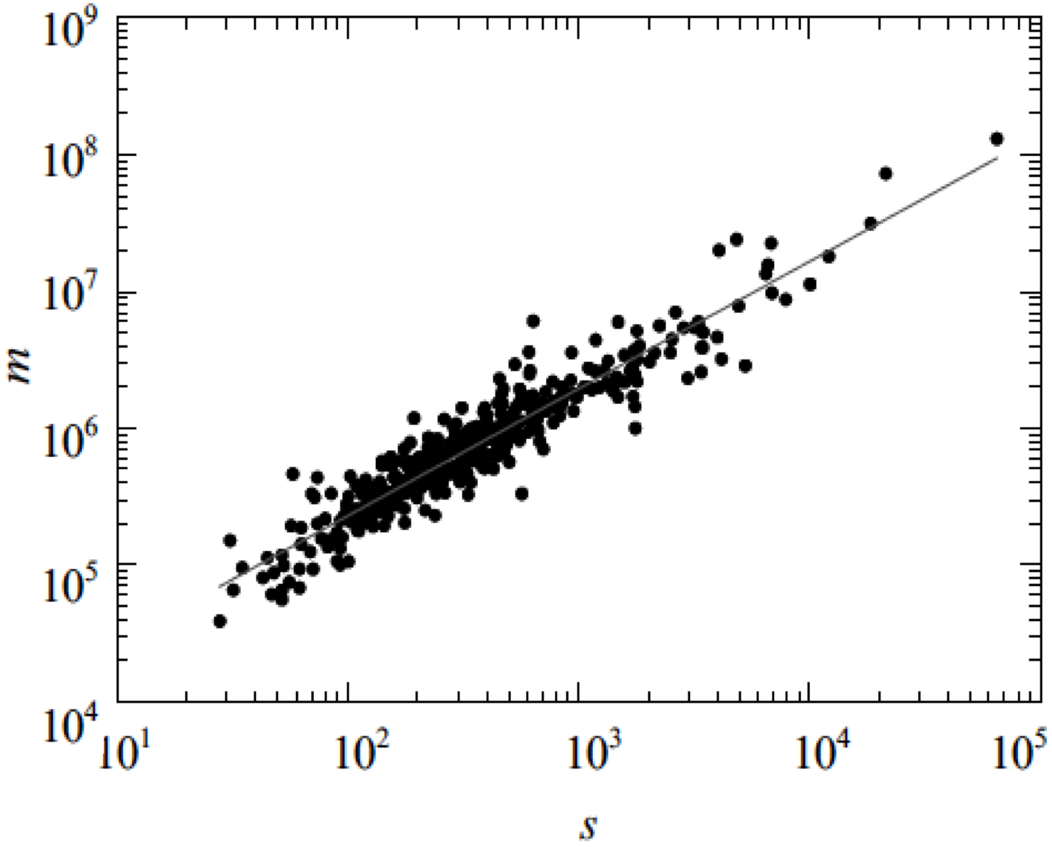


Figure 7. Average strength of the municipalities as a function of the degree.



m is the monthly income of the municipality, defined as per worker times the number of workers

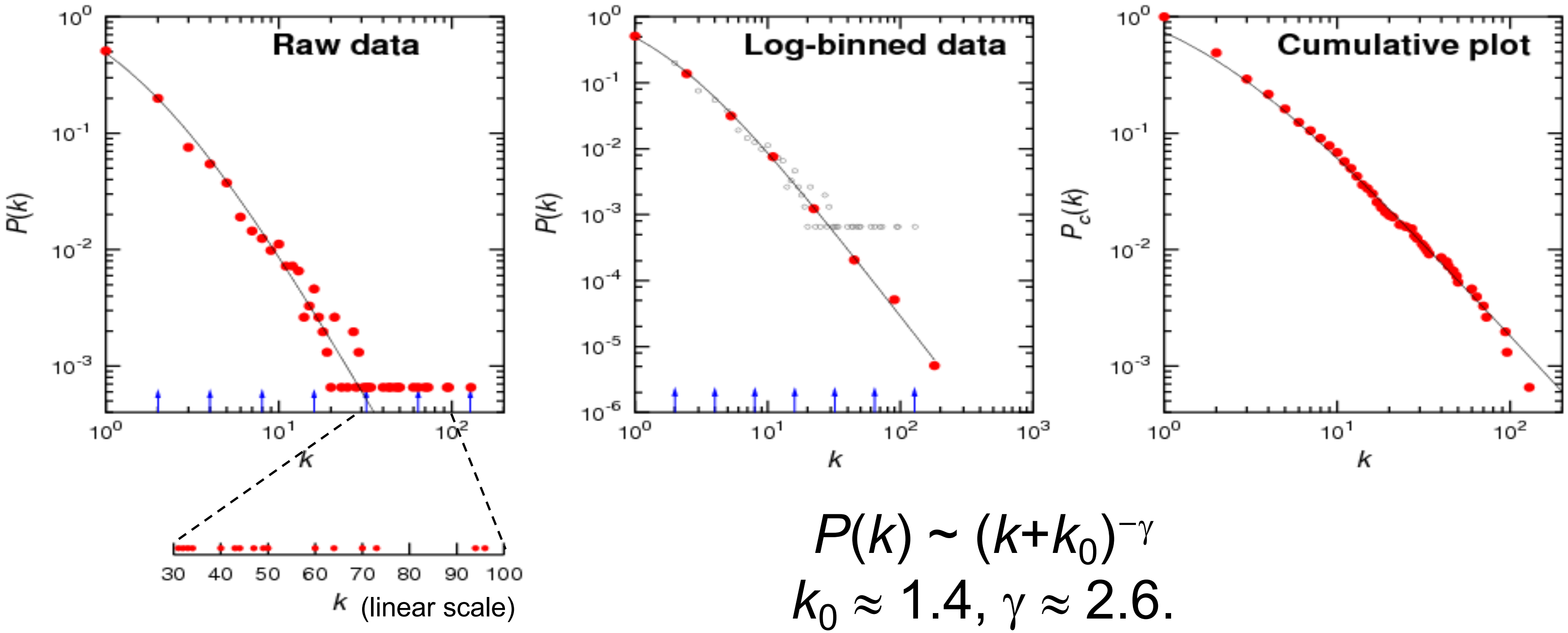
Table 4. Comparison of properties of transportation networks. SNCM denotes the Sardinian intermunicipal commuting network.

Network	$P(k)$	$P(s)$	s	$Y_2(k)$
Global: WAN	heavy tail	broad	$\beta = 1.5$	$\theta = 1.0$
Intercities: SNCM	light tail	broad	$\beta = 1.9$	$\theta = 0.4$
Intraurban (Chowell et al, 2003)	heavy tail	broad	$\beta = 1.0$	–

Conclusions

- They have shown that a more complete view of complex networks is provided by the study of the interactions defining the links of these systems. The weights characterizing the various connections exhibit complex statistical features with highly varying distributions and power-law behavior
- For the case of the SCN and the WAN, the analysis of the weighted quantities and the study of the correlations between weights and topology provide a complementary perspective on the structural organization of the network that might be undetected by quantities based only on topological information.
- In particular, the weighted network approach allows the consideration of clustering algorithms based on the strength and topology of the commuting patterns.
- These approaches could inform models. The role of physical space (or nodes coordinates) play a role and needs to be added.

Reminder: The PDF when it has a heterogeneous distributu



$$P(k) \sim (k+k_0)^{-\gamma}$$
$$k_0 \approx 1.4,\ \gamma \approx 2.6.$$

PageRank

- **Random surfer model:** browse the Web at random
  - A random link is clicked from each page to get to the next
- Random walk model modified with random jumps (**teleportation**)
  - At each step, with probability α, stop browsing and start new session form a random page
- Recursive definition
- PageRank is conserved, $\sum_i R(i) = 1$

Power method to calculate PageRank:

- Initialize each node with

$$R_0 = 1/N$$

- At each iteration *t*, loop over nodes and update PageRank of each node *i* via this recursive equation:

$$R_t(i) = \boxed{\frac{\alpha}{N}} + \boxed{(1-\alpha) \sum_{j \in pred(i)} \frac{R_{t-1}(j)}{k_{out}(j)}}$$

probability to land on *i* by teleportation

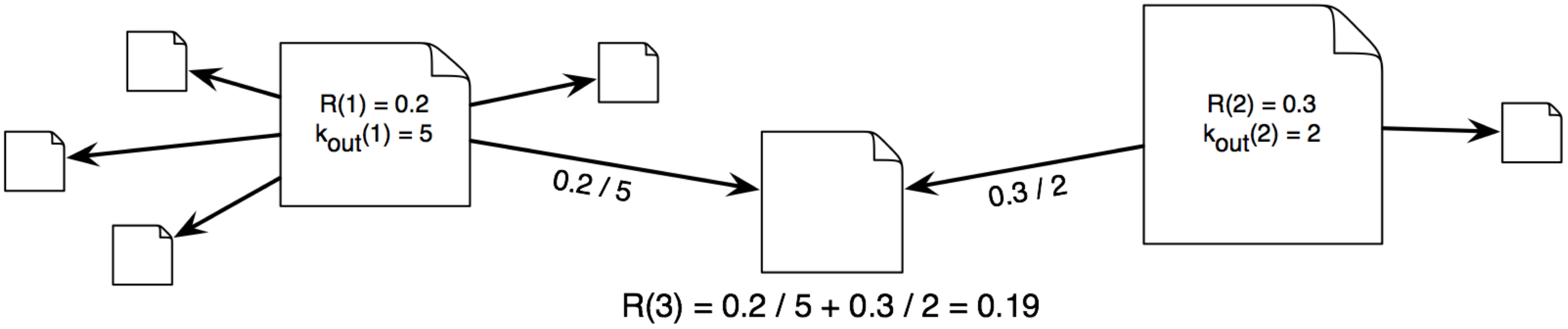
probability to land on *i* by random surfing

PageRank

- α is the **teleportation** or **jump factor**, typically 0.15
  - PageRank converges quickly (in few iterations) if α > 0
- 1– α is the **damping factor**

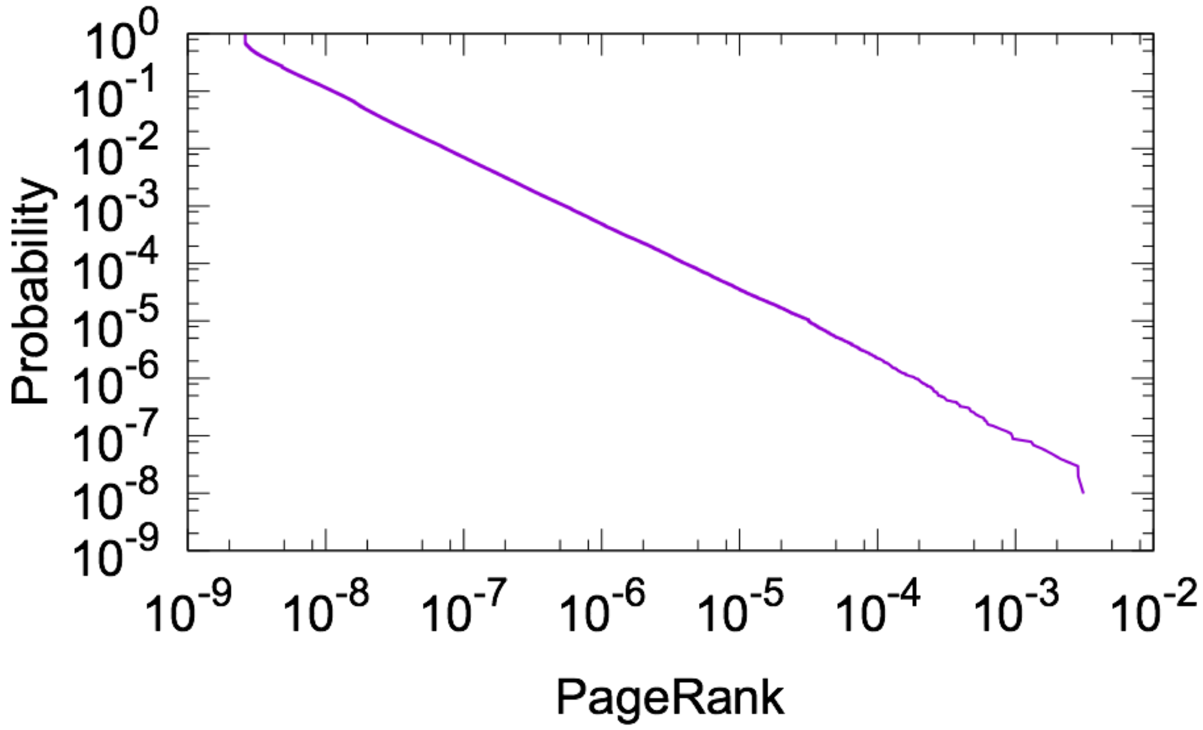
Example with α = 0:

$$R_t(i) = \sum_{j \in pred(i)} \frac{R_{t-1}(j)}{k_{out}(j)}$$



PageRank vs in-degree

- PageRank distribution is very skewed, similar to that of in-degree
- PageRank is similar to in-degree to a first approximation (if all incoming links originate from pages with the same PageRank)
- But links from more important pages bring more importance
- Search engine optimization (SEO) companies try to boost a client website's PageRank
  - Unscrupulous SEOs may use spamdexing but if caught by search engines, client can be de-listed



Lab script: Lecture8_Weights_and_Directions_MigrationNets.ipynb

Learning Objectives:

1. Analyze data as Weighted network
2. Plot Log-binned Probability Density Functions
3. Plot Weighted Histograms
4. Visualize Spatial Networks

Quantifying Global International Migration Flows

Guy J. Abel[*], Nikola Sander[*,†]

+ See all authors and affiliations

Science 28 Mar 2014:
Vol. 343, Issue 6178, pp. 1520-1522
DOI: 10.1126/science.1248676

Monitoring Migration

Migrant "stock" data—the number of people living in a country other than the one in which they were born—are frequently used to understand contemporary trends in international migration, but the data are severely limited. **Abel and Sander** (p. 1520) present a set of global bilateral migration flows estimated from sequential stock data in 5-year intervals. The percentage of the world population moving over 5-year periods has not shown dramatic changes between 1995 and 2010. People from individual African countries tended to move within the continent, whereas people from Europe tended to move to very diverse locations.

Abstract

Widely available data on the number of people living outside of their country of birth do not adequately capture contemporary intensities and patterns of global migration flows. We present data on bilateral flows between 196 countries from 1990 through 2010 that provide a comprehensive view of international migration flows. Our data suggest a stable intensity of global 5-year migration flows at ~0.6% of world population since 1995. In addition, the results aid the interpretation of trends and patterns of migration flows to and from individual countries by placing them in a regional or global context. We estimate the largest movements to occur between South and West Asia, from Latin to North America, and within Africa.

Supplementary Materials

Quantifying Global International Migration Flows

Guy J. Abel, Nikola Sander

Materials/Methods, Supplementary Text, Tables, Figures, and/or References

Download Supplement

- Materials and Methods
- Tables S1 to S5
- Full References List

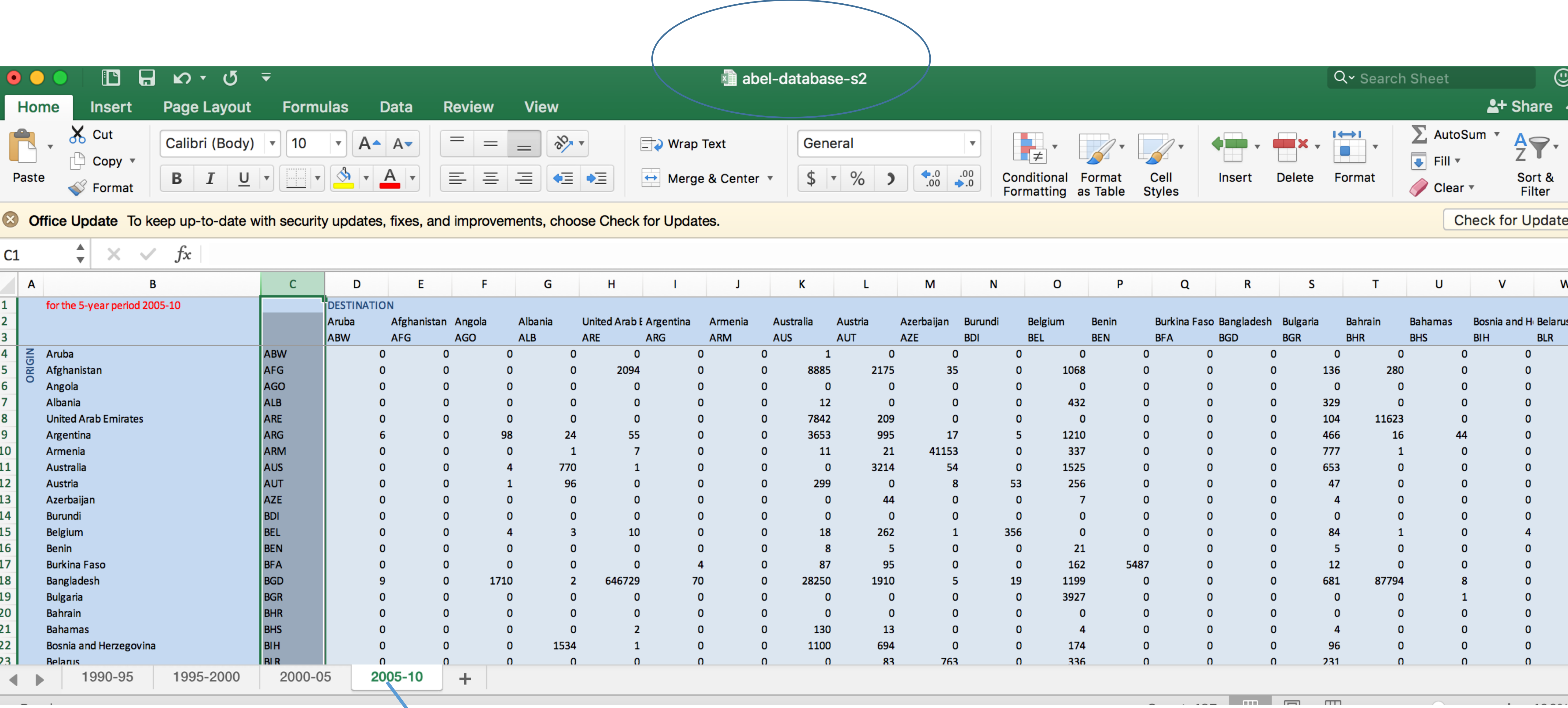
Database S1

- Bilateral flow estimates by region, 2005-10
- This is a 15*15 matrix stored as an excel file. Rows correspond to origins, columns to destinations.

Database S2

- Bilateral flow estimates by country, 1990-95 to 2005-10
- This is a 196*196 matrix stored as an excel file. Rows correspond to origins, columns to destinations. Countries a indicated by their iso-3 code.

https://www.science.org/doi/10.1126/science.1248676



	for the 5-year period 2005-10		DESTINATION																			
			Aruba	Afghanistan	Angola	Albania	United Arab E	Argentina	Armenia	Australia	Austria	Azerbaijan	Burundi	Belgium	Benin	Burkina Faso	Bangladesh	Bulgaria	Bahrain	Bahamas	Bosnia and H	Belarus
			ABW	AFG	AGO	ALB	ARE	ARG	ARM	AUS	AUT	AZE	BDI	BEL	BEN	BFA	BGD	BGR	BHR	BHS	BIH	BLR
ORIGIN	Aruba	ABW	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
	Afghanistan	AFG	0	0	0	0	2094	0	0	8885	2175	35	0	1068	0	0	0	136	280	0	0	
	Angola	AGO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Albania	ALB	0	0	0	0	0	0	0	12	0	0	0	432	0	0	0	329	0	0	0	
	United Arab Emirates	ARE	0	0	0	0	0	0	0	7842	209	0	0	0	0	0	0	104	11623	0	0	
	Argentina	ARG	6	0	98	24	55	0	0	3653	995	17	5	1210	0	0	0	466	16	44	0	
	Armenia	ARM	0	0	0	1	7	0	0	11	21	41153	0	337	0	0	0	777	1	0	0	
	Australia	AUS	0	0	4	770	1	0	0	0	3214	54	0	1525	0	0	0	653	0	0	0	
	Austria	AUT	0	0	1	96	0	0	0	299	0	8	53	256	0	0	0	47	0	0	0	
	Azerbaijan	AZE	0	0	0	0	0	0	0	0	44	0	0	7	0	0	0	4	0	0	0	
	Burundi	BDI	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Belgium	BEL	0	0	4	3	10	0	0	18	262	1	356	0	0	0	0	84	1	0	4	
	Benin	BEN	0	0	0	0	0	0	0	8	5	0	0	21	0	0	0	5	0	0	0	
	Burkina Faso	BFA	0	0	0	0	0	4	0	87	95	0	0	162	5487	0	0	12	0	0	0	
	Bangladesh	BGD	9	0	1710	2	646729	70	0	28250	1910	5	19	1199	0	0	0	681	87794	8	0	
	Bulgaria	BGR	0	0	0	0	0	0	0	0	0	0	0	3927	0	0	0	0	0	1	0	
	Bahrain	BHR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Bahamas	BHS	0	0	0	0	2	0	0	130	13	0	0	4	0	0	0	4	0	0	0	
	Bosnia and Herzegovina	BIH	0	0	0	1534	1	0	0	1100	694	0	0	174	0	0	0	96	0	0	0	
	Belarus	BLR	0	0	0	0	0	0	0	0	83	763	0	336	0	0	0	231	0	0	0	

From the original data we exported: migration.csv

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
	Aruba	Afghanistan	Angola	Albania	United Arab	Argentina	Armenia	Australia	Austria	Azerbaijan	Burundi	Belgium	Benin	Burkina Faso	Bangladesh	Bulgaria	Bahrain	Bahamas	Bosnia and
Aruba	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
Afghanistan	0	0	0	0	2094	0	0	8885	2175	35	0	1068	0	0	0	136	280	0	
Angola	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Albania	0	0	0	0	0	0	0	12	0	0	0	432	0	0	0	329	0	0	
United Arab	0	0	0	0	0	0	0	7842	209	0	0	0	0	0	0	104	11623	0	
Argentina	6	0	98	24	55	0	0	3653	995	17	5	1210	0	0	0	466	16	44	
Armenia	0	0	0	1	7	0	0	11	21	41153	0	337	0	0	0	777	1	0	
Australia	0	0	4	770	1	0	0	0	3214	54	0	1525	0	0	0	653	0	0	
Austria	0	0	1	96	0	0	0	299	0	8	53	256	0	0	0	47	0	0	
Azerbaijan	0	0	0	0	0	0	0	0	44	0	0	7	0	0	0	4	0	0	
Burundi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Belgium	0	0	4	3	10	0	0	18	262	1	356	0	0	0	0	84	1	0	
Benin	0	0	0	0	0	0	0	8	5	0	0	21	0	0	0	5	0	0	
Burkina Faso	0	0	0	0	0	4	0	87	95	0	0	162	5487	0	0	12	0	0	
Bangladesh	9	0	1710	2	646729	70	0	28250	1910	5	19	1199	0	0	0	681	87794	8	
Bulgaria	0	0	0	0	0	0	0	0	0	0	0	3927	0	0	0	0	0	1	
Bahrain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bahamas	0	0	0	0	2	0	0	130	13	0	0	4	0	0	0	4	0	0	
Bosnia and H	0	0	0	1534	1	0	0	1100	694	0	0	174	0	0	0	96	0	0	
Belarus	0	0	0	0	0	0	0	0	83	763	0	336	0	0	0	231	0	0	
Belize	0	0	0	0	0	0	0	14	9	0	0	0	0	0	0	1	0	2	
Bolivia	0	0	0	0	0	56615	0	116	98	0	1	0	0	0	0	13	0	0	
Brazil	22	0	3026	9	61	6715	0	9119	2092	1	2	5398	0	0	0	291	7	23	
Barbados	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	
Brunei	2	0	0	0	1597	2	0	10236	50	0	0	16	0	0	496	1	199	3	
Bhutan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Botswana	0	0	5	0	0	0	0	343	6	0	267	0	0	0	0	14	0	6	
Central Afric	0	0	11	0	0	0	0	10	3	0	850	0	0	0	0	0	0	0	
Canada	0	0	20	35	9	0	0	2403	2162	35	2121	2586	0	0	0	327	62	1	
Switzerland	0	0	1	5	3	0	0	768	2427	0	45	463	0	0	0	96	0	0	
Channel Islar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Chile	0	0	5	1	3	4243	0	4223	361	1	6	173	0	0	0	20	1	9	
China	191	0	0	0	0	2736	0	150947	4402	151	0	4474	0	0	0	465	0	20	
Ivory Coast	0	0	159	0	25	1	0	992	406	0	0	2782	13631	119154	1	28	3	1	
Cameroon	0	0	0	0	0	0	0	123	183	0	0	4606	0	0	0	5	0	0	
Democratic F	0	0	710	0	785	0	0	305	4	0	16780	4	3	0	0	1	92	0	
Republic of C	0	0	159	0	0	0	0	149	7	0	593	13	10	0	0	1	0	0	

Seen as txt files

```
,Aruba,Afghanistan,Angola,Albania,United Arab Emirates,Argentina,Armenia,Australia,Austria,Azerbaijan,Burundi,Belgium,Benin,Burkina Faso,Bangladesh,Bulgaria,Bahrain,Bahamas,Bosnia and Herzegovi
na,Belarus,Belize,Bolivia,Brazil,Barbados,Brunei,Bhutan,Botswana,Central African Republic,Canada,Switzerland,Channel Islands,Chile,China,Ivory Coast,Cameroon,Democratic Republic of the Congo,Re
public of Congo,Colombia,Comoros,Cape Verde,Costa Rica,Cuba,Cyprus,Czech Republic,Germany,Djibouti,Denmark,Dominican Republic,Algeria,Ecuador,Egypt,Eritrea,Western Sahara,Spain,Estonia,Ethiopia
,Finland,Fiji,France,Micronesia,Gabon,United Kingdom,Georgia,Ghana,Guinea,Guadeloupe,Gambia,Guinea-Bissau,Equatorial Guinea,Greece,Grenada,Guatemala,French Guiana,Guam,Guyana,Hong Kong,Honduras
,Croatia,Haiti,Hungary,Indonesia,India,Ireland,Iran,Iraq,Iceland,Israel,Italy,Jamaica,Jordan,Japan,Kazakhstan,Kenya,Kyrgyzstan,Cambodia,South Korea,Kuwait,Laos,Lebanon,Liberia,Libya,Saint Lucia
,Sri Lanka,Lesotho,Lithuania,Luxembourg,Latvia,Macao,Morocco,Moldova,Madagascar,Maldives,Mexico,Macedonia,Mali,Malta,Myanmar,Montenegro,Mongolia,Mozambique,Mauritania,Martinique,Mauritius,Malaw
i,Malaysia,Mayotte,Namibia,New Caledonia,Niger,Nigeria,Nicaragua,Netherlands,Norway,Nepal,New Zealand,Oman,Pakistan,Panama,Peru,Philippines,Papua New Guinea,Poland,Puerto Rico,North Korea,Portu
gal,Paraguay,Palestine,French Polynesia,Qatar,Reunion,Romania,Russia,Rwanda,Saudi Arabia,Sudan,Senegal,Singapore,Solomon Islands,Sierra Leone,El Salvador,Somalia,Serbia,South Sudan,Sao Tome and
 Principe,Suriname,Slovakia,Slovenia,Sweden,Swaziland,Syria,Chad,Togo,Thailand,Tajikistan,Turkmenistan,East Timor,Tonga,Trinidad and Tobago,Tunisia,Turkey,Tanzania,Uganda,Ukraine,Uruguay,United
 States,Uzbekistan,Saint Vincent and the Grenadines,Venezuela,Virgin Islands,Vietnam,Vanuatu,Samoa,Yemen,South Africa,Zambia,Zimbabwe
Aruba,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,14,0,0,1,0,0,0,0,0,0,0,13,0,0,0,42,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,
0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,87,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10
1,0,0,0,0,0,0,0,0
Afghanistan,0,0,0,0,2094,0,0,8885,2175,35,0,1068,0,0,0,136,280,0,0,133,0,1,26,0,0,0,0,0,12270,1247,0,3,0,0,0,0,0,36,0,0,1,0,3,326,32852,0,4478,1,2,33,34,0,0,0,3,0,1379,0,1091,0,0,13444,9,0,0,0,
0,0,0,62,0,0,0,0,0,0,0,0,0,316,0,0,44,266595,18,3,810,1782,0,4,0,567,0,12,0,7,104,0,0,0,37,0,0,0,10,0,17,0,0,63,0,0,2,0,0,1,0,0,0,0,1,0,0,0,145,0,0,0,0,0,3,8894,4701,0,896,0,0,3,0,210,0,62,0,0,
5,0,0,0,550,0,3,5731,0,796,0,0,0,0,0,0,0,0,0,0,0,0,63,4,6388,0,566,0,0,0,0,20,0,0,0,0,4119,0,0,315,0,13781,192,0,5,0,0,0,0,0,13,0,0
Angola,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0
Albania,0,0,0,0,0,0,0,12,0,0,0,432,0,0,0,329,0,0,0,0,0,0,0,0,0,0,0,0,38,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,117,5495,0,41,0,0,10,0,0,0,146,0,0,46,0,196,0,0,4402,0,0,0,0,0,0,0,8912,0,0,0,0,0,0,0,0,0,21
,0,0,0,0,0,4,0,48150,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,6986,0,0,0,0,0,0,0,0,0,0,0,0,0,0,60,75,0,0,0,0,0,0,0,0,0,40,0,0,0,0,0,0,0,0,0,194,0,0,0,0,0,0,0,0,0,0,0,0,0,20,4,240,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
United Arab Emirates,0,0,0,0,0,0,0,7842,209,0,0,0,0,0,0,104,11623,0,0,0,0,3,47,0,0,0,0,0,15701,507,0,7,0,0,0,0,0,57,0,0,0,2,190,8,0,0,463,0,896,9,1381,0,0,0,0,0,126,0,35392,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,281,0,12104,370,0,253,3,0,437,0,238,0,0,0,0,0,0,25166,0,0,0,2817,0,0,0,60,0,17,0,0,0,0,0,10,0,0,19,0,0,0,0,185,0,0,0,0,0,0,0,0,0,0,3,789,389,0,1166,6181,0,2,8,431,0,235,0,0,2,
0,13960,0,24049,0,173,564,0,0,0,0,0,0,0,0,0,0,0,4924,0,0,125,18,1101,0,0,0,0,0,0,0,0,0,0,0,0,0,49,0,0,0,0,11196,0,0,10,0,0,0,0,0,0,209,0,0
Argentina,6,0,98,24,55,0,0,3653,995,17,5,1210,0,0,0,466,16,44,0,51,2,13192,2466,2,1,0,31,17,6996,3546,20,10015,23,0,0,1,17,2226,0,35,143,4,111,2422,5863,0,805,602,103,7313,78,0,0,136370,16,0,49
8,0,3448,0,0,3696,0,2,0,1,0,0,269,763,0,176,0,0,2,7,22,55,254,651,8,0,689,3,2,137,18147,91127,3,8,678,921,0,1,0,32,0,0,2,0,3,1,0,0,108,321,82,0,0,7,8,0,3860,8,0,14,0,17,2,0,56,0,4,0,0,0,0,12,0,
0,14,1097,998,0,611,0,0,665,36,260,0,1820,0,1,1392,22126,2,1,14,0,3107,1325,5,3,0,0,18,0,0,30,0,45,9,0,16,456,308,1658,0,10,0,0,130,0,1,0,0,1,12,820,40,31,343,3682,28117,10,0,3837,0,24,0,0,0,29
07,0,0
Armenia,0,0,0,1,7,0,0,11,21,41153,0,337,0,0,0,777,1,0,0,77,0,2,5,0,0,0,0,0,53,127,0,0,3,0,0,0,3,1,0,1,0,0,60,633,172,0,54,0,4,12,0,0,0,360,19,0,30,0,1158,0,0,10,42,0,0,0,0,0,0,174,0,0,0,0,0,0,0
,0,0,40,0,0,0,2544,3,1,12,546,0,0,0,12905,0,59,0,1,0,0,0,0,0,0,0,19,0,1,0,0,299,0,0,0,4,0,1,0,1,6,0,0,0,0,0,0,0,0,0,0,0,0,0,159,140,0,3,0,0,0,0,0,0,137,0,0,44,0,0,0,2,0,41,10551,0,0,0,0,0,0,0,0
,0,0,0,0,0,13,4,698,0,0,0,0,0,0,111,0,0,0,0,417,0,0,2867,0,4,416,0,0,0,0,0,0,0,4,0,0
Australia,0,0,4,770,1,0,0,0,3214,54,0,1525,0,0,0,653,0,0,0,118,0,63,206,8,0,1,105,9,3577,2548,1,47,1554,0,0,0,7,99,0,3,2,0,1764,5307,13840,0,1415,5,140,553,103,0,3,9483,37,0,883,293,2254,0,0,22
546,0,5,0,0,0,0,0,13132,0,11,0,0,0,407,0,580,0,2611,641,0,911,1,0,322,1939,47436,0,0,1058,1667,0,1,0,1299,0,2,0,0,0,5,0,0,138,247,136,0,0,22,9,0,53,258,0,334,0,405,111,0,1,0,31,0,0,0,0,54,0,0,0
,1559,2486,0,9918,0,0,84,0,450,0,1400,0,97,498,8,0,1,0,0,1939,2480,11,0,0,0,1756,0,0,2,0,6544,1137,0,0,1195,908,2314,21,0,0,0,0,0,7,0,0,0,7,6846,38,18,414,0,308,42,0,29,0,30,155,0,0,1729,0,0
Austria,0,0,1,96,0,0,0,299,0,8,53,256,0,0,0,47,0,0,0,0,0,2,10,0,0,0,0,0,118,677,0,1,0,0,0,11,0,1,0,0,0,0,15,4765,9500,0,85,0,9,11,5,0,0,1221,0,0,41,0,78,0,0,429,0,0,0,0,0,0,0,26,0,0,0,0,0,0,0,4
9,0,1669,0,0,25,1,0,13,117,1600,0,0,0,292,1,3,0,0,0,0,0,39,0,0,0,0,8,65,16,0,0,4,0,0,0,7,0,4,0,42,1,0,0,0,0,0,0,0,0,0,0,0,0,71,142,0,5,0,0,1,0,0,0,414,0,0,27,0,0,0,1,0,160,162,4,0,0,0,0,0,0,0,0
,161,190,0,0,3187,507,154,0,0,0,0,0,0,3,0,0,0,0,746,0,15,1,0,0,18,0,2,0,1,0,0,0,22,0,0
Azerbaijan,0,0,0,0,0,0,0,0,44,0,0,7,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,48,0,0,0,0,0,0,0,0,0,0,0,0,3,13,1,0,3,0,1,1,1,0,0,58,0,0,3,0,21,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,0,0,2,5,0,0,10,1
41,0,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,8,0,1,0,0,0,0,0,0,22,0,0,1,0,0,0,0,0,4,249,0,0,0,0,0,0,0,0,0,0,1,0,0,0,2,2,10,0,0,0,0,0,0,0,0,0,0,0,0,162,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
Burundi,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0
Belgium,0,0,4,3,10,0,0,18,262,1,356,0,0,0,0,84,1,0,4,0,0,6,0,0,0,0,0,0,8,878,0,0,0,0,0,0,0,0,6,0,0,0,0,27,118,3004,0,152,0,22,42,7,0,0,1384,0,0,78,0,1650,0,0,390,0,0,0,0,0,0,0,0,267,0,0,0,0,0,0,0,6
,0,215,0,0,120,1,2,6,594,9910,0,0,0,75,4,1,0,0,0,0,0,0,0,0,0,0,24,1654,27,0,0,0,2,0,0,17,0,5,0,1,0,0,3,0,0,0,0,0,0,0,0,0,0,0,36,136,0,8,0,0,6,0,0,0,1196,0,0,58,0,0,0,3,0,310,12,67,0,0,0,0,0,0,0,0
:
```

```python
# run this from a terminal, to install the packet:
#pip install geopy
import geopy
from geopy.geocoders import Nominatim
import networkx as nx
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import csv
from os.path import join
#this will allow the plot to be inline in the browser
%matplotlib inline
```

```python
lines = np.loadtxt("migration.csv",dtype='str',delimiter=',')
```

```python
destinations=lines[0];
```

```python
destinations
```

```
array(['', 'Aruba', 'Afghanistan', 'Angola', 'Albania',
       'United Arab Emirates', 'Argentina', 'Armenia', 'Australia',
       'Austria', 'Azerbaijan', 'Burundi', 'Belgium', 'Benin',
       'Burkina Faso', 'Bangladesh', 'Bulgaria', 'Bahrain', 'Bahamas',
       'Bosnia and Herzegovina', 'Belarus', 'Belize', 'Bolivia', 'Brazil',
       'Barbados', 'Brunei', 'Bhutan', 'Botswana',
       'Central African Republic', 'Canada', 'Switzerland',
       'Channel Islands', 'Chile', 'China', 'Ivory Coast', 'Cameroon',
       'Democratic Republic of the Congo', 'Republic of Congo',
       'Colombia', 'Comoros', 'Cape Verde', 'Costa Rica', 'Cuba',
       'Cyprus', 'Czech Republic', 'Germany', 'Djibouti', 'Denmark',
       'Dominican Republic', 'Algeria', 'Ecuador', 'Egypt', 'Eritrea',
       'Western Sahara', 'Spain', 'Estonia', 'Ethiopia', 'Finland',
       'Fiji', 'France', 'Micronesia', 'Gabon', 'United Kingdom',
```

```
lines[5]
```

```
array(['United Arab Emirates', '0', '0', '0', '0', '0', '0', '0', '7842',
       '209', '0', '0', '0', '0', '0', '0', '104', '11623', '0', '0', '0',
       '0', '3', '47', '0', '0', '0', '0', '0', '15701', '507', '0', '7',
       '0', '0', '0', '0', '0', '57', '0', '0', '0', '2', '190', '8', '0',
       '0', '463', '0', '896', '9', '1381', '0', '0', '0', '0', '0',
       '126', '0', '35392', '0', '0', '0', '0', '0', '0', '0', '0', '0',
       '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '281', '0',
       '12104', '370', '0', '253', '3', '0', '437', '0', '238', '0', '0',
       '0', '0', '0', '0', '25166', '0', '0', '0', '2817', '0', '0', '0',
       '60', '0', '17', '0', '0', '0', '0', '0', '10', '0', '0', '19',
       '0', '0', '0', '0', '185', '0', '0', '0', '0', '0', '0', '0', '0',
       '0', '3', '789', '389', '0', '1166', '6181', '0', '2', '8', '431',
       '0', '235', '0', '0', '2', '0', '13960', '0', '24049', '0', '173',
       '564', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '4924',
       '0', '0', '125', '18', '1101', '0', '0', '0', '0', '0', '0', '0',
       '0', '0', '0', '0', '49', '0', '0', '0', '0', '11196', '0', '0',
       '10', '0', '0', '0', '0', '0', '209', '0', '0'], dtype='<U32')
```

```
g = nx.DiGraph()

for i in range(1,len(lines)):
    count=0;
    data=lines[i];
    node=data[0];
    #print(data) ##contains each row first element is the name of the origin
    for col in data[1:]:
        count=count+1;
        d=destinations[count];
        wij=int(col)
        if(wij>0):
            g.add_edge(node,d,weight=wij)
            print(node,'  ',d,'  ',col)
```

```
Botswana   Egypt   2
Botswana   Finland   6
Botswana   France   40
Botswana   Greece   1
Botswana   Hungary   2
Botswana   Ireland   23
Botswana   Italy   29
Botswana   Kenya   3
Botswana   Malta   1
Botswana   Namibia   28
Botswana   Nicaragua   1
Botswana   Netherlands   45
Botswana   Norway   36
Botswana   New Zealand   48
Botswana   Rwanda   28
Botswana   South Sudan   198
Botswana   Slovakia   1
Botswana   Slovenia   1
Botswana   Sweden   29
Botswana   Swaziland   2
```

Note: the Dictionary pos is used in draw_network to locate the nodes in space
the library geopy is used to obtain the coordinates of a country given its name

```python
from geopy.exc import GeocoderTimedOut
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="countries")
location = geolocator.geocode("USA")
#print(location.latitude,' ',location.longitude)


### Here we use it to create the dictionary with coordinates
pos = {}
for n in g.nodes():
    location = geolocator.geocode(n,timeout=30)
    pos[n] = (location.longitude,location.latitude) #we are calling a geolocator function to generate coordinates
    print(n," ",location.latitude," ",location.longitude)
```

```
Netherlands   52.15517   5.38721
Norway   60.5000209   9.0999715
Panama   8.559559   -81.1308434
Venezuela   8.0018709   -66.1109318
Afghanistan   33.7680065   66.2385139
United Arab Emirates   24.0002488   53.9994829
Austria   47.59397   14.12456
Azerbaijan   40.3936294   47.7872508
Belgium   50.6402809   4.6667145
Bulgaria   42.6073975   25.4856617
Bahrain   26.1551249   50.5344606
Belarus   53.4250605   27.6971358
Bolivia   -17.0568696   -64.9912286
Brazil   -10.3333333   -53.2
Switzerland   46.7985624   8.2319736
Cyprus   34.9823018   33.1451285
Czech Republic   49.8167003   15.4749544
Germany   51.0834196   10.4234469
Denmark   55.670249   10.3333283
Dominican Republic   19.0974031   -70.3028026
```

Check minimum and maximun weighted in-degree out-degree and of nodes

8]:

```python
min_w_deg=min(dict(g.in_degree(weight='weight')).values())

max_w_deg=max(dict(g.in_degree(weight='weight')).values())
print("min_in= ",min_w_deg,"and max_in= ",max_w_deg)
in_degrees=g.in_degree(weight='weight')
    # sort nodes by degree
from operator import itemgetter
in_nodes=sorted(g.in_degree(weight='weight'),key=itemgetter(1)) #sorted in degree nodes dictionary

```

Here I am sorting the dictionary

```
min_in=  0 and max_in=  6825626
```

9]:

```python
out_degrees=g.out_degree(weight='weight')
from operator import itemgetter
out_nodes=sorted(g.out_degree(weight='weight'),key=itemgetter(1)) #sorted in degree nodes dictionary

print("min_out= ",min(dict(g.out_degree(weight='weight')).values()),"and max_out= ",max(dict(g.out_degree(weight='w

```

```
min_out=  0 and max_out=  3724386
```

```
1  in_nodes
```

```
('Kazakhstan', 478008),
('Malaysia', 740333),
('Singapore', 745572),
('Thailand', 753393),
('India', 765380),
('South Sudan', 784893),
('South Africa', 847585),
('Qatar', 857425),
('Germany', 1063359),
('France', 1064185),
('Canada', 1257093),
('Russia', 1298188),
('Saudi Arabia', 1299438),
('Australia', 1316181),
('United Kingdom', 1699995),
('Italy', 2018126),
('Spain', 2251716),
('United Arab Emirates', 3258880),
('United States', 6825626)]
```

```
1  out_nodes
```

```
('Sudan', 512760),
('Germany', 513701),
('Uzbekistan', 533066),
('Brazil', 550978),
('France', 564404),
('Iran', 570929),
('Malaysia', 654996),
('Morocco', 677053),
('United Kingdom', 679646),
('Peru', 725157),
('Zimbabwe', 900199),
('Philippines', 1248311),
('Indonesia', 1306825),
('United States', 1869951),
('Pakistan', 2024652),
('Mexico', 2040911),
('China', 2090305),
('Bangladesh', 2966047),
('India', 3724386)]
```

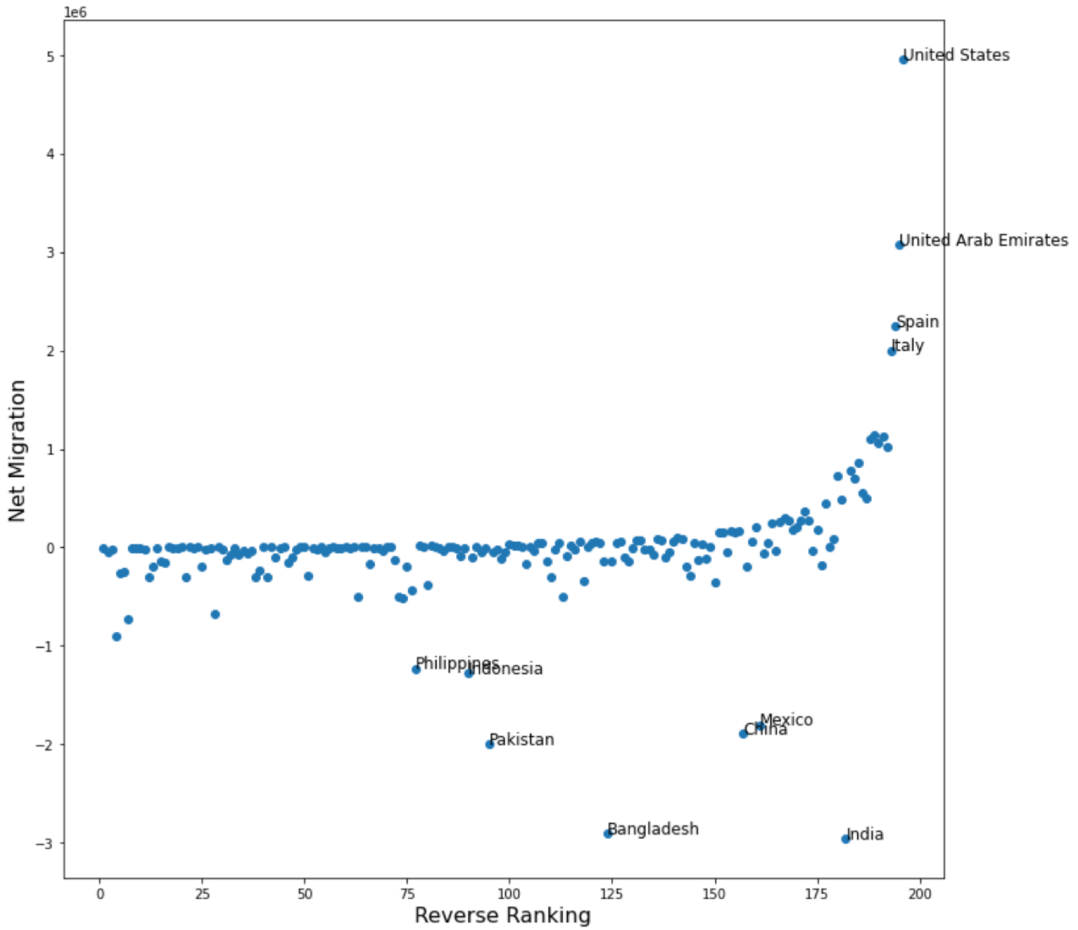
We write inflow outflow and net flow defined as in minus out

```
count=0;
z=[];
y=[];
nn=[];
for n in in_nodes:
    count=count+1; #We are saving the order in the list of incoming population
    z.append(count);
    net=list(dict(g.in_degree(n,'weight')).values())[0]-list(dict(g.out_degree(n,'weight')).values())[0]
    y.append(net)
    nn.append(n[0]) ;
    print(n[0],' ',count,' ',list(dict(g.in_degree(n,'weight')).values())[0],' ',list(dict(g.out_degree(n,'weight'))
```

```
Kazakhstan   178   478008   471099   6909
Malaysia   179   740333   654996   85337
Singapore   180   745572   23757   721815
Thailand   181   753393   260371   493022
India   182   765380   3724386   -2959006
South Sudan   183   784893   0   784893
South Africa   184   847585   148096   699489
Qatar   185   857425   313   857112
Germany   186   1063359   513701   549658
France   187   1064185   564404   499781
Canada   188   1257093   158614   1098479
Russia   189   1298188   163692   1134496
Saudi Arabia   190   1299438   243297   1056141
Australia   191   1316181   191453   1124728
United Kingdom   192   1699995   679646   1020349
Italy   193   2018126   19339   1998787
Spain   194   2251716   1868   2249848
United Arab Emirates   195   3258880   182111   3076769
United States   196   6825626   1869951   4955675
```

Plot in order net flow vs. degree index annotating coutries with largest netflow

```
fig, ax = plt.subplots(figsize=(12,12))
ax.scatter(z, y)
plt.xlabel("Reverse Ranking",fontsize=16)
plt.ylabel("Net Migration", fontsize=16)
for i, txt in enumerate(nn): #writing the names if the growth or decrease is more than 1.2M
    if(abs(y[i])>1200000):
        ax.annotate(txt, (z[i],y[i]), fontsize=12)
```




```
g.number_of_nodes()
```

196

```
np.median(list(dict(g.out_degree(weight='weight')).values()))
```

53820.0

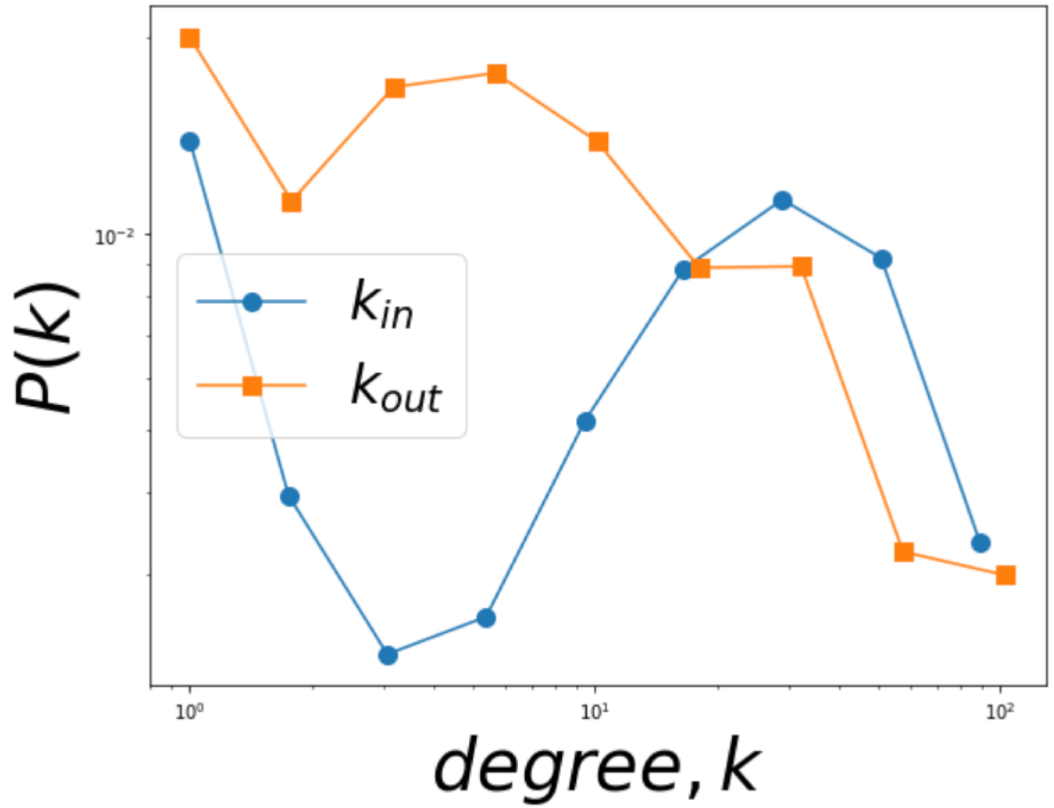
```
np.median(list(dict(g.in_degree(weight='weight')).values()))
```

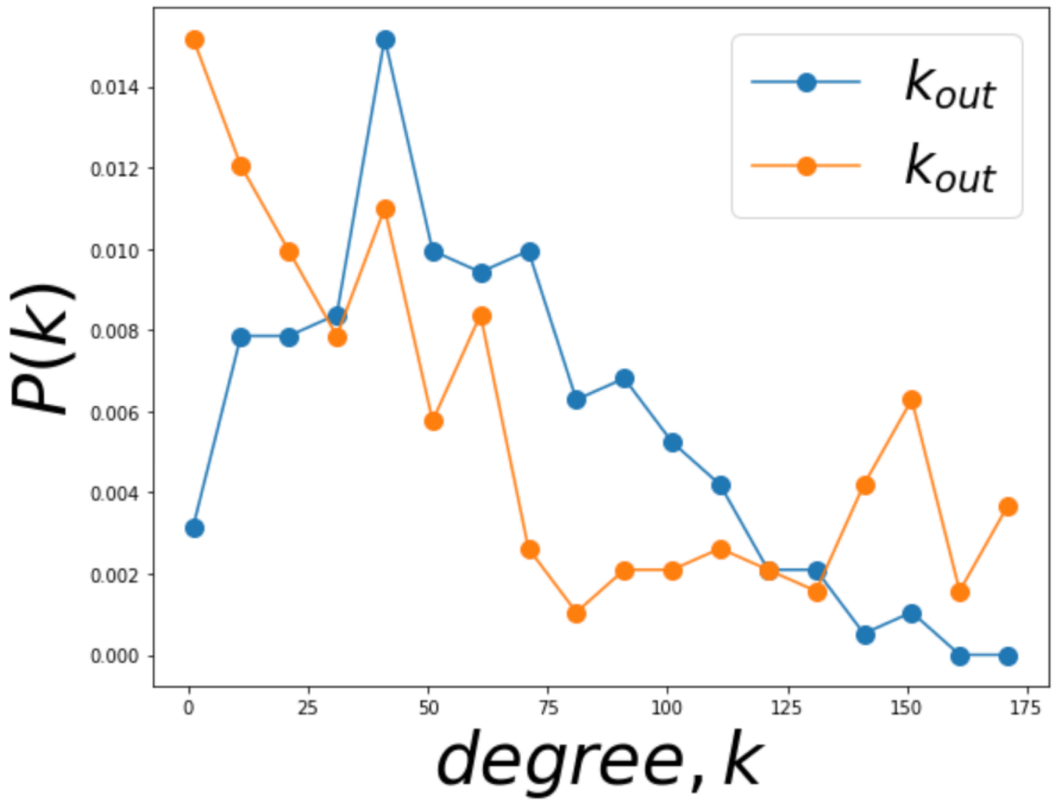
34476.5

Pair of countries with more than 200,000 migrants (from - to)

```python
for a, b, data in sorted(g.edges(data=True), key=lambda a_b_data: a_b_data[2]['weight']):
    if data['weight'] > 200000:
        print(('{a} {b} {w}'.format(a=a, b=b, w=data['weight'])))
```

```
Indonesia United Arab Emirates 200422
China South Korea 216747
India United Kingdom 250403
China Hong Kong 251540
United States Italy 255654
Morocco Spain 265763
Afghanistan Iran 266595
El Salvador United States 268935
Pakistan Saudi Arabia 285441
Tanzania Burundi 294595
India Qatar 297570
Uzbekistan Russia 299656
Kazakhstan Russia 307495
Indonesia Malaysia 346048
Myanmar Thailand 368832
Philippines United States 373024
Malaysia Singapore 395727
Pakistan United Arab Emirates 494846
Zimbabwe South Africa 495779
Bangladesh Saudi Arabia 523342
China United States 615536
Bangladesh India 630451
Bangladesh United Arab Emirates 646729
India United States 701242
India United Arab Emirates 1149965
Mexico United States 1957397
```

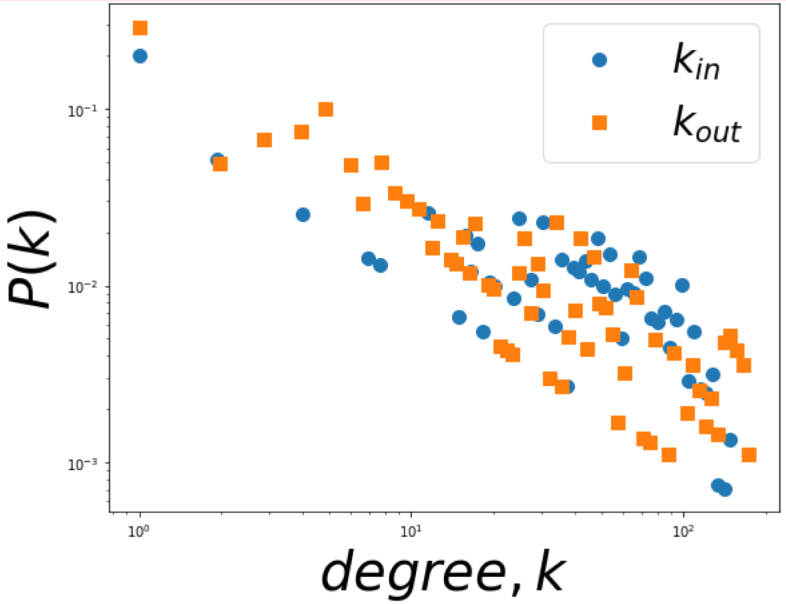




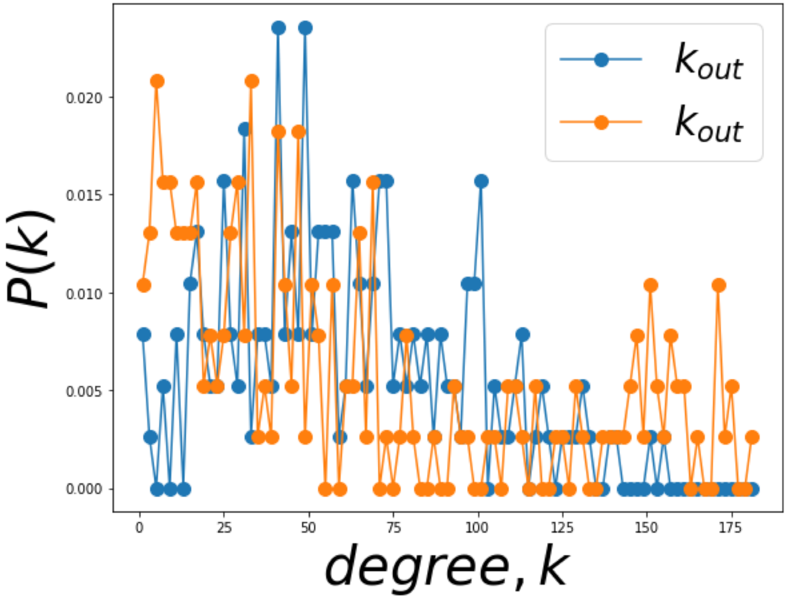
(this is the same plot of Fig 2 from the SCMN paper, also in slide 14)

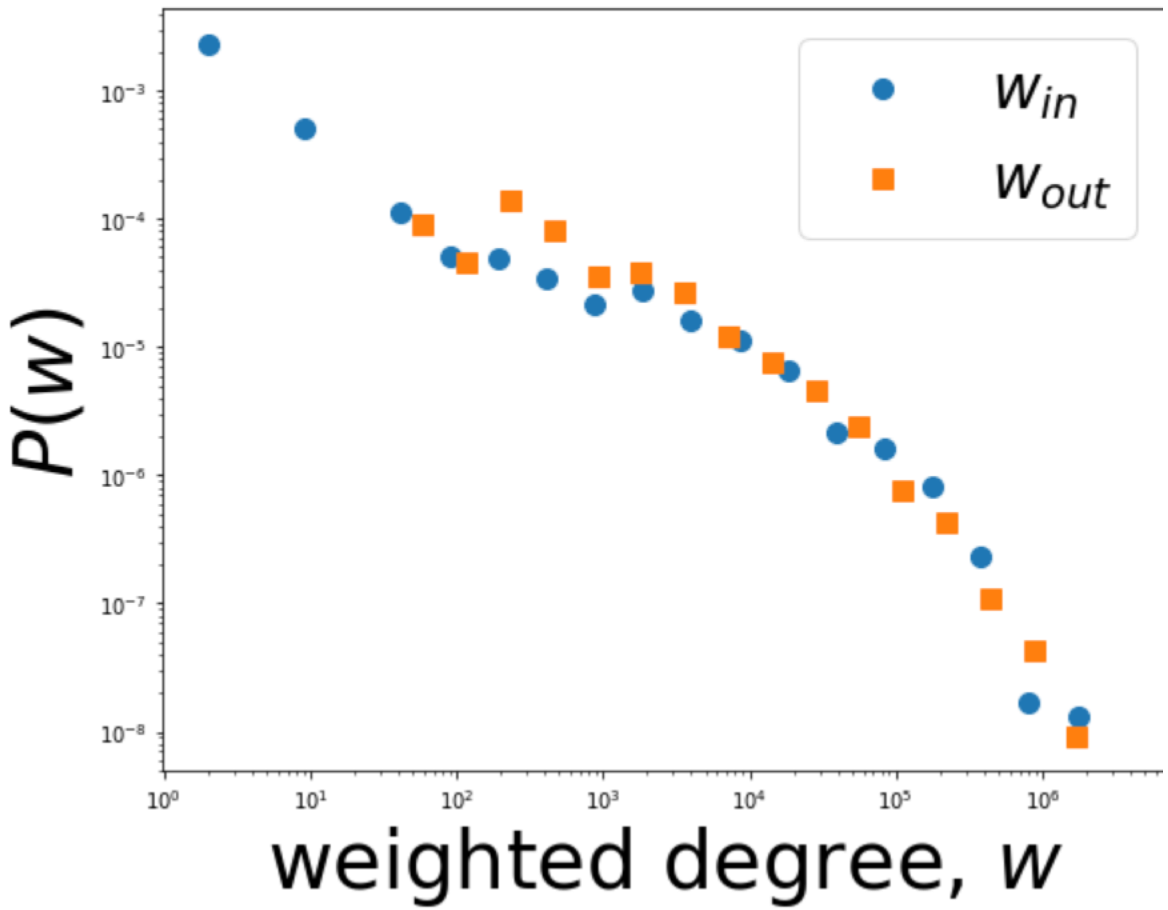
What is the difference with the previous slide?

```python
#plt.hist(out_weighted,log=True)
#plt.yscale('log')
#plt.xscale('log')
fig, ax = plt.subplots()
fig.set_size_inches((9, 7))

n_bins = 100


out_logBins = np.logspace(np.log10(min(out_d)), np.log10(max(out_d)),num=n_bins)
out_logBinDensity, out_binedges = np.histogram(out_d, bins=out_logBins, density=True)


in_logBins = np.logspace(np.log10(min(in_d)), np.log10(max(in_d)),num=n_bins)
in_logBinDensity, in_binedges = np.histogram(in_d, bins=in_logBins, density=True)

ax.loglog(out_logBins[:-1],out_logBinDensity,'o', markersize=10,label=r'$k_{in}$')
ax.loglog(in_logBins[:-1],in_logBinDensity,'s', markersize=10,label=r'$k_{out}$')
ax.legend(fontsize=30)


ax.set_xlabel('$degree, k$',fontsize=40)
ax.set_ylabel('$P(k)$',fontsize=40)
plt.savefig("distributions.eps",dpi=200,bbox_inches='tight')
```




```python
fig, ax = plt.subplots()
fig.set_size_inches((9, 7))
n_out, bins_out = np.histogram(out_d, bins = range(min(in_d), max(in_d)+1,2), density="True")
n_in, bins_in = np.histogram(in_d, bins = range(min(in_d), max(in_d)+1, 2), density="True")

ax.plot(bins_out[:-1],n_out,'-o', markersize=10,label=r'$k_{out}$')
ax.plot(bins_in[:-1],n_in,'-o', markersize=10,label=r'$k_{out}$')
ax.set_xlabel('$degree, k$',fontsize=40)
ax.set_ylabel('$P(k)$',fontsize=40)
ax.legend(fontsize=30)

plt.show()
```





(this is the same plot of Fig 5b from the paper, also slide 17)

Page Rank

```python
PR=nx.pagerank(g, alpha=0.85, personalization=None, max_iter=100,
```

```python
PR['Aruba']
```

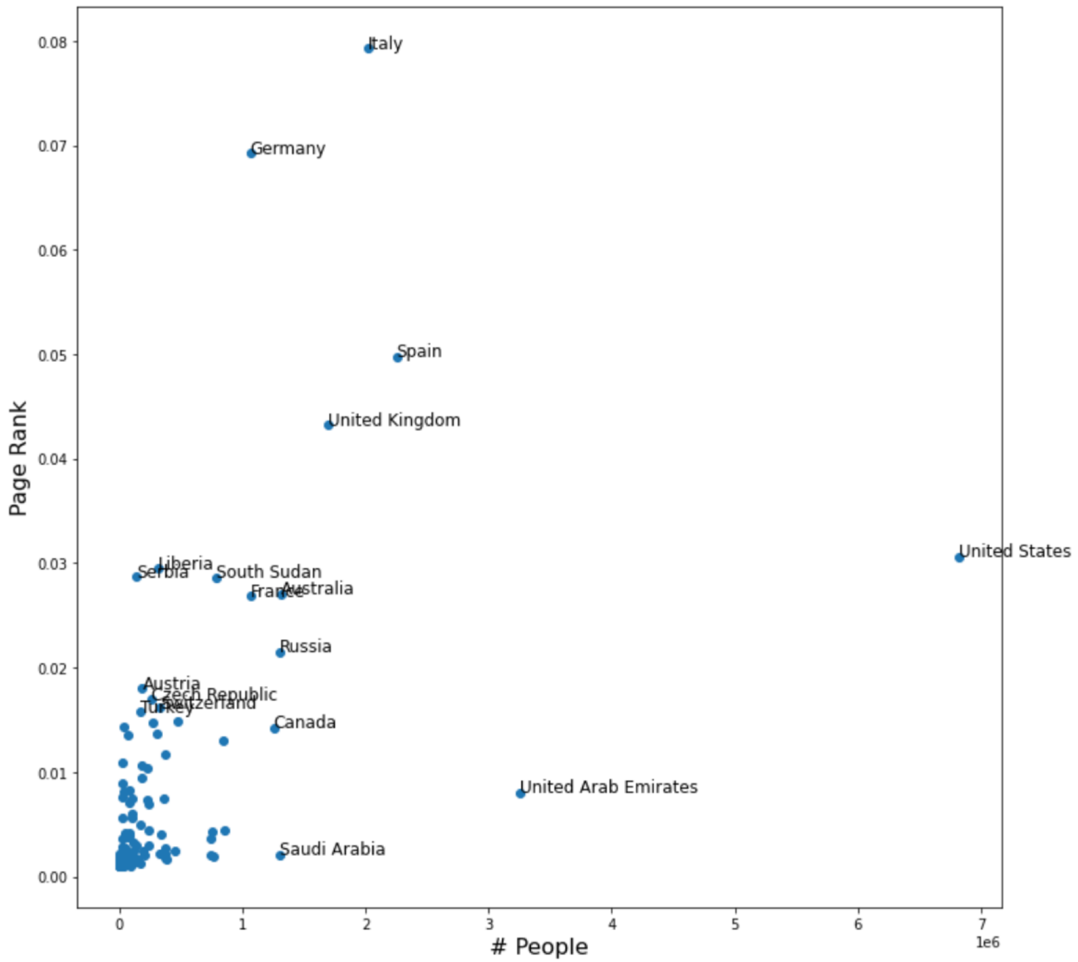
```
0.0010492771032448898
```

```python
g.in_degree('Aruba','weight')
```

```
4271
```

```python
count=0;
z=[];
y=[];
nn=[];
for n in in_nodes:
    y.append(PR[n[0]]);
    z.append(g.in_degree(n[0],'weight'))
    nn.append(n[0]) ;
    print(n[0],' ',g.in_degree(n[0],'weight'),PR[n[0]])
```

```python
fig, ax = plt.subplots(figsize=(12,12))
ax.scatter(z, y)
plt.xlabel("# People",fontsize=16)
plt.ylabel("Page Rank", fontsize=16)
for i, txt in enumerate(nn): #writing the names if the growth or decrease is more than 1.5M
    if(abs(z[i])>1200000 or y[i]>0.015):
        ax.annotate(txt, (z[i],y[i]), fontsize=12)
```



From the Migration Paper:

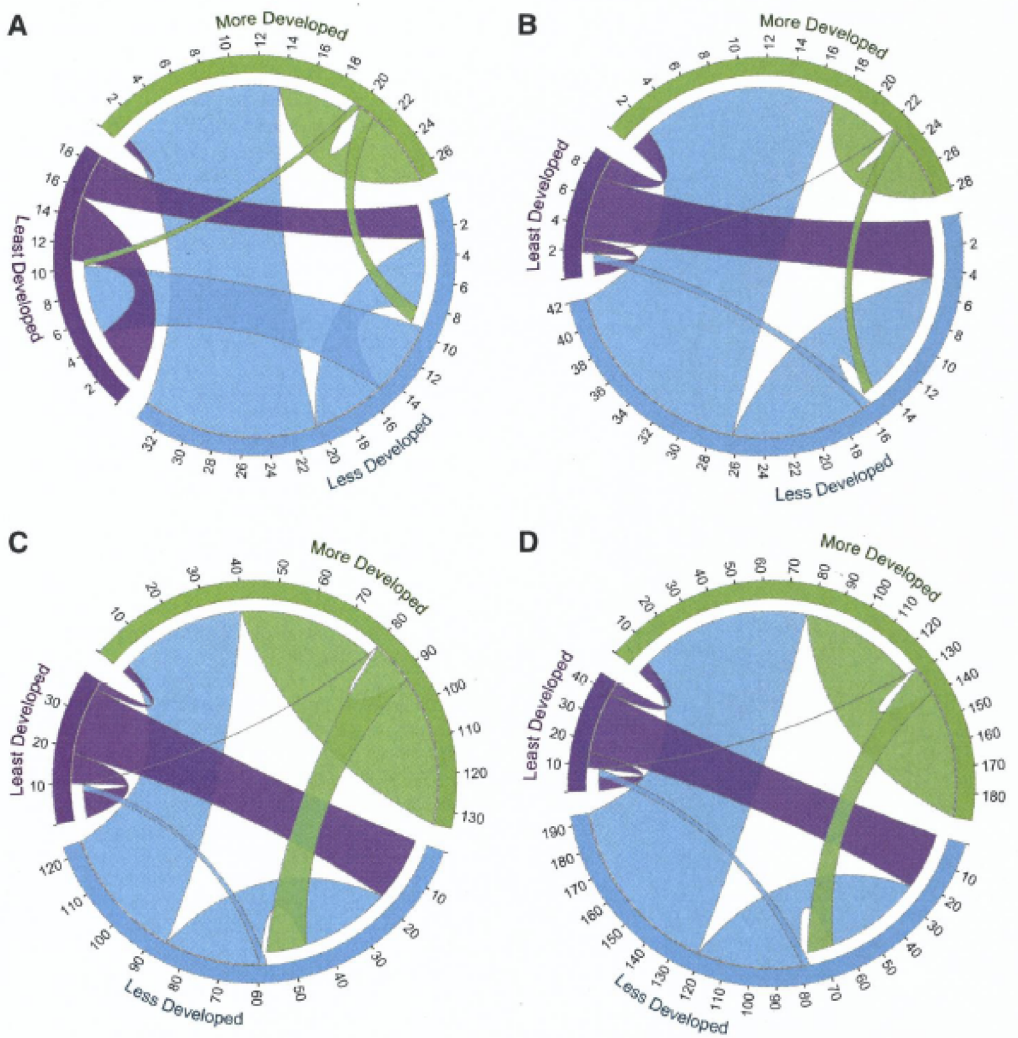


Fig. 2. Comparing estimated migrant flows to stocks in early 1990s and late 2000s. Migration flows between more developed (green), less developed (blue), and least developed (purple) countries. **(A)** Flows during 1990 to 1995. **(B)** Flows during 2005 to 2010. **(C)** Stock data from 1990. **(D)** Stock data from 2010. Tick marks on the circle segments show the number of migrants (inflows and outflows) in millions.

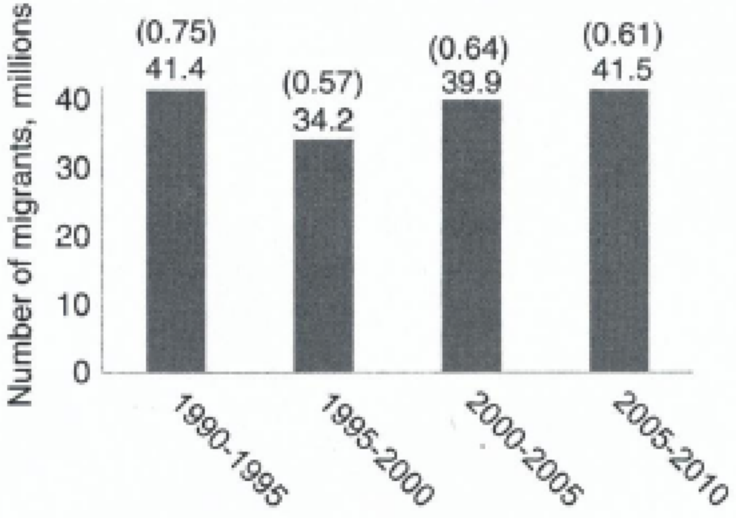


Fig. 3. The global number of international movements between 196 countries in four quinquennial periods, 1990 to 2010. Percentages (shown in parentheses) are calculated by using the world population at the beginning of the period.

```python
kk = []
wij = []
degrees = g.degree()
for n in g.nodes(data=True):
    for e in g.edges(n[0],data=True):
        kk.append(degrees[e[0]]*degrees[e[1]])
        wij.append(g[e[0]][e[1]]['weight'])
```

Here you can change it

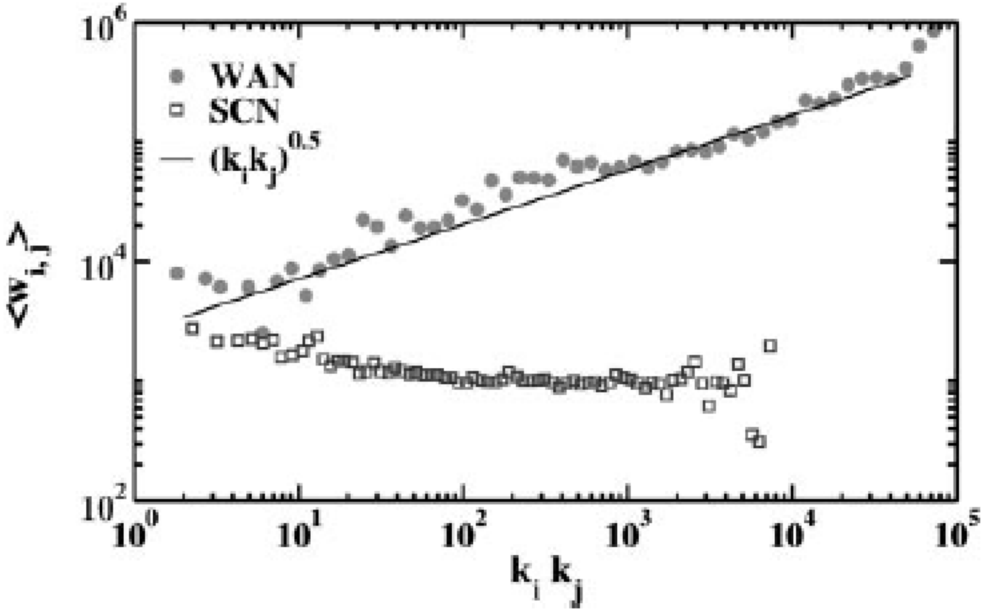


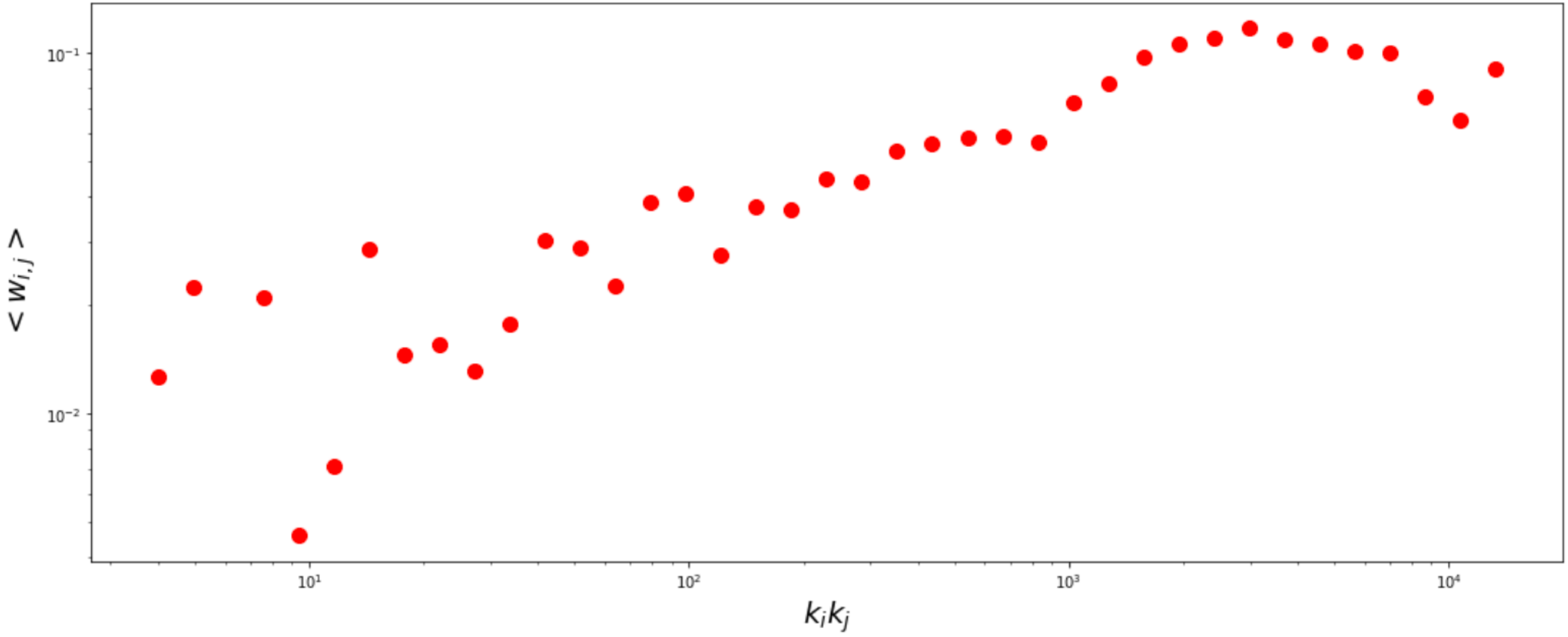
Fig. 4. Average weight as a function of the end-point degree. The solid line corresponds to a power-law behavior $\langle w_{ij} \rangle \sim (k_i k_j)^\theta$, with exponent $\theta = 0.5 \pm 0.1$. In the case of the SCN it is possible to observe an almost flat behavior for roughly two orders of magnitude.

WAN=World Air Network
SCN=Science Collaboration Network

https://www.pnas.org/content/101/11/3747

```python
n_bins = 40
kk_logBins = np.logspace(np.log10(min(kk)), np.log10(max(kk)),num=n_bins)
counts, bins = np.histogram(kk, bins=kk_logBins);
sums, bins = np.histogram(kk, bins=kk_logBins,weights=wij);
avg_w = sums/counts;
```

```
fig, ax = plt.subplots()
fig.set_size_inches((18, 7))
ax.loglog(bins[:-1],avg_w,linewidth=0,color='r',marker='o',markersize=10)
ax.set_xlabel('$k_ik_j$',fontsize=20)
ax.set_ylabel('$<w_{i,j}>$',fontsize=20)
plt.savefig("commutes_directed_.eps",dpi=200,bbox_inches='tight')
```



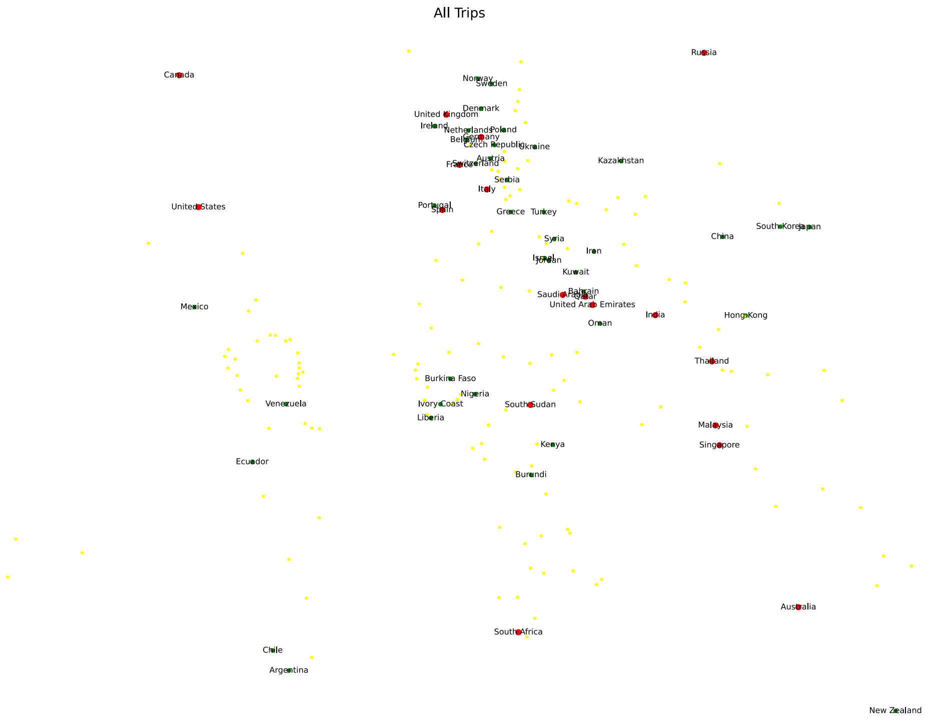
Create list of nodes with 3 different sizes and colors depending on their incoming number of migrants

```
import matplotlib.colors as colors
plt.figure(3)

degree_in=dict(g.in_degree(weight='weight')).values()

ns=[] #saves node size
nc=[] #saves node color
n_label = {}
#for n in g.nodes_iter():
 #   n_label[n] = n

for node in g.nodes():
    dn=g.in_degree(node,'weight')
    #print node
    if(dn>=500000):
        n_label[node] = node
        ns.append(4000)  #select node size
        nc.append("red") #select node color
    if(dn>=100000 and dn<500000):
        ns.append(2000)
        nc.append("green")
        n_label[node] = node
    if(dn<100000):
        ns.append(1000)
        nc.append("yellow")


```

Note: In this case Node labels are not saved for countries with less than 10,000 inmigrants

Visualizing nodes in space

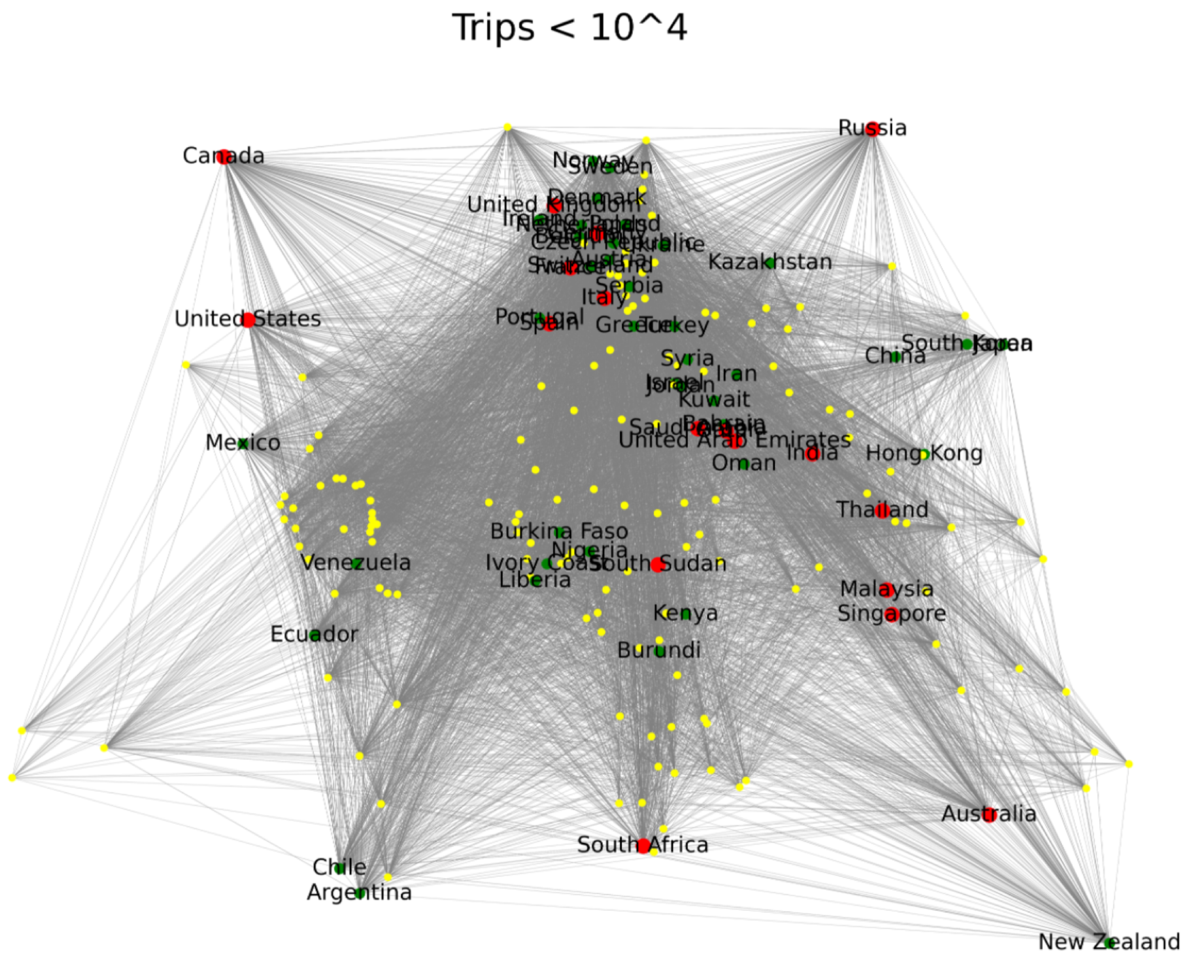
```
#pos=nx.spring_layout(g)
plt.figure(figsize=(200,150))
nx.draw_networkx_nodes(g, pos=pos, node_size=ns, #draw network passing coordinates, node sozes and colors
              node_color = nc)
nx.draw_networkx_labels(g,pos,n_label,font_size=90,font_color='k') #write labels
plt.title('All Trips',fontsize=150)
plt.axis('off')
plt.savefig('map0', format='pdf')
```



Showing links only in several ranges of weight values

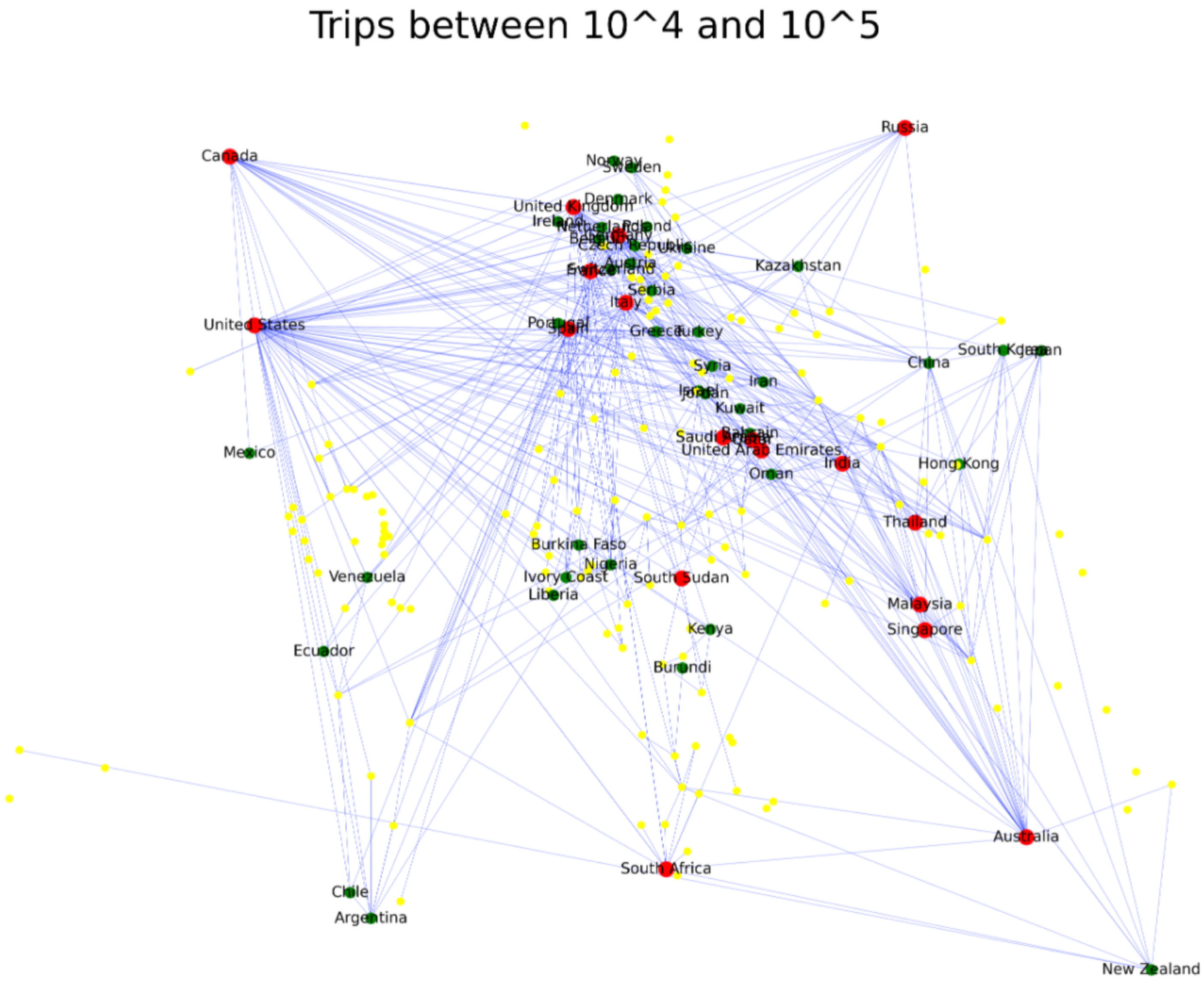
```
# Create list of edeges by color

eI = []
eII = []
eIII = []
eIV= []

for u,v in g.edges():
    wij=g.get_edge_data(u,v)['weight']
    if (wij>=500000):
       eI.append((u,v))
    if (wij>=100000 and wij<500000):
       eII.append((u,v))
    if (wij>=10000 and wij<100000):
       eIII.append((u,v))
    if (wij<10000):
       eIV.append((u,v))

```

```
#here we show the edgelist I
plt.figure(5)
plt.figure(figsize=(100,75))
nx.draw_networkx_edges(g,pos,edgelist=eIV,width=1,alpha=0.9,edge_color='gray')
nx.draw_networkx_nodes(g,pos,label=n_label,node_size=ns,node_color=nc)
nx.draw_networkx_labels(g,pos,n_label,font_size=90,font_color='k')
plt.title('Trips < 10^4',fontsize=150)
plt.axis('off')

```



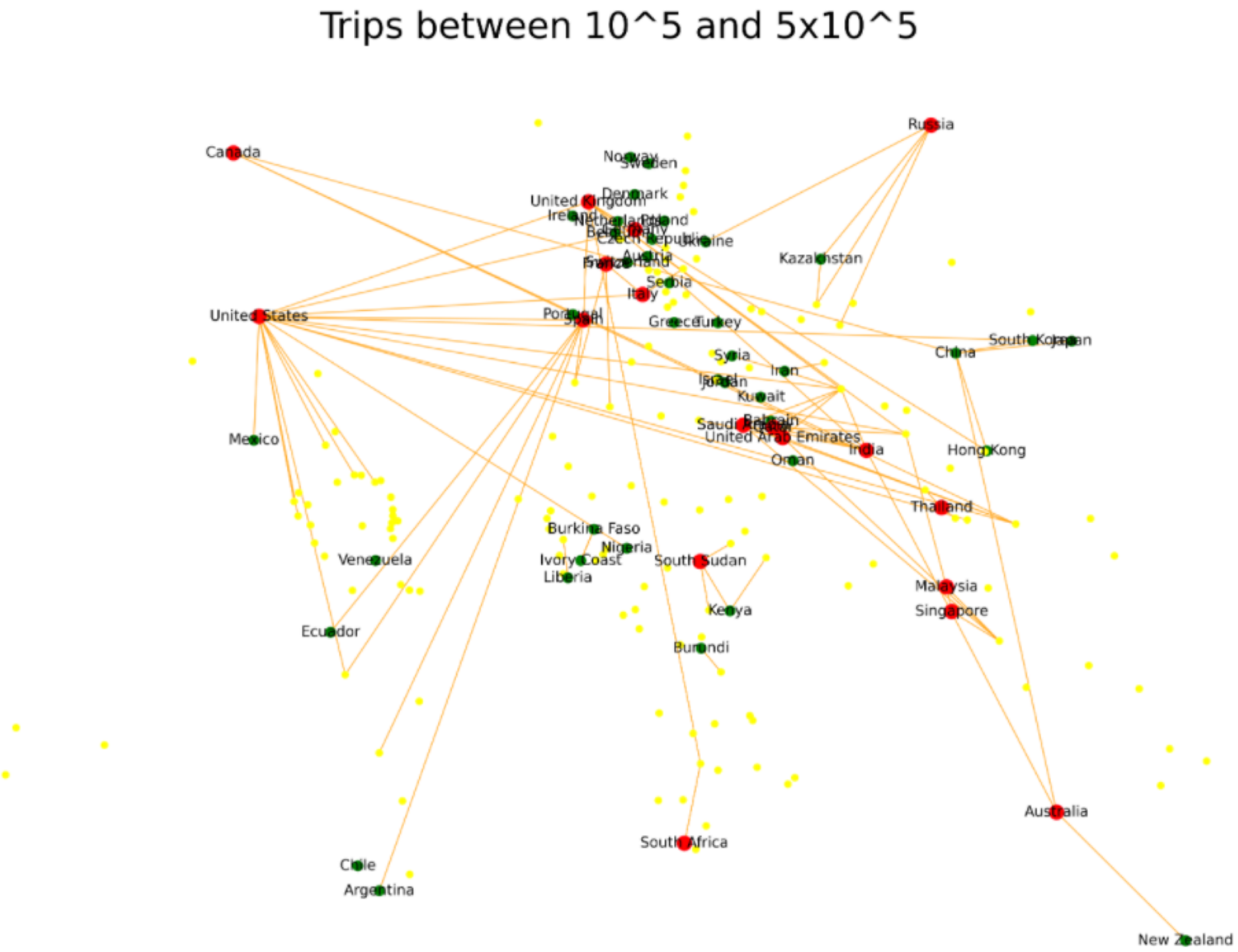
```python
plt.figure(6)
plt.figure(figsize=(100,75))
nx.draw_networkx_edges(g, pos, edgelist = eIII, width = 1 , alpha = 0.9,edge_color='blue')
nx.draw_networkx_nodes(g,pos=pos,node_size=ns,node_color=nc)
nx.draw_networkx_labels(g,pos,n_label,font_size=60,font_color='k')
plt.title('Trips between 10^4 and 10^5',fontsize=150)
plt.axis('off')
plt.savefig('map3.pdf', format='pdf',dpi=1000)
```

```
<Figure size 432x288 with 0 Axes>
```



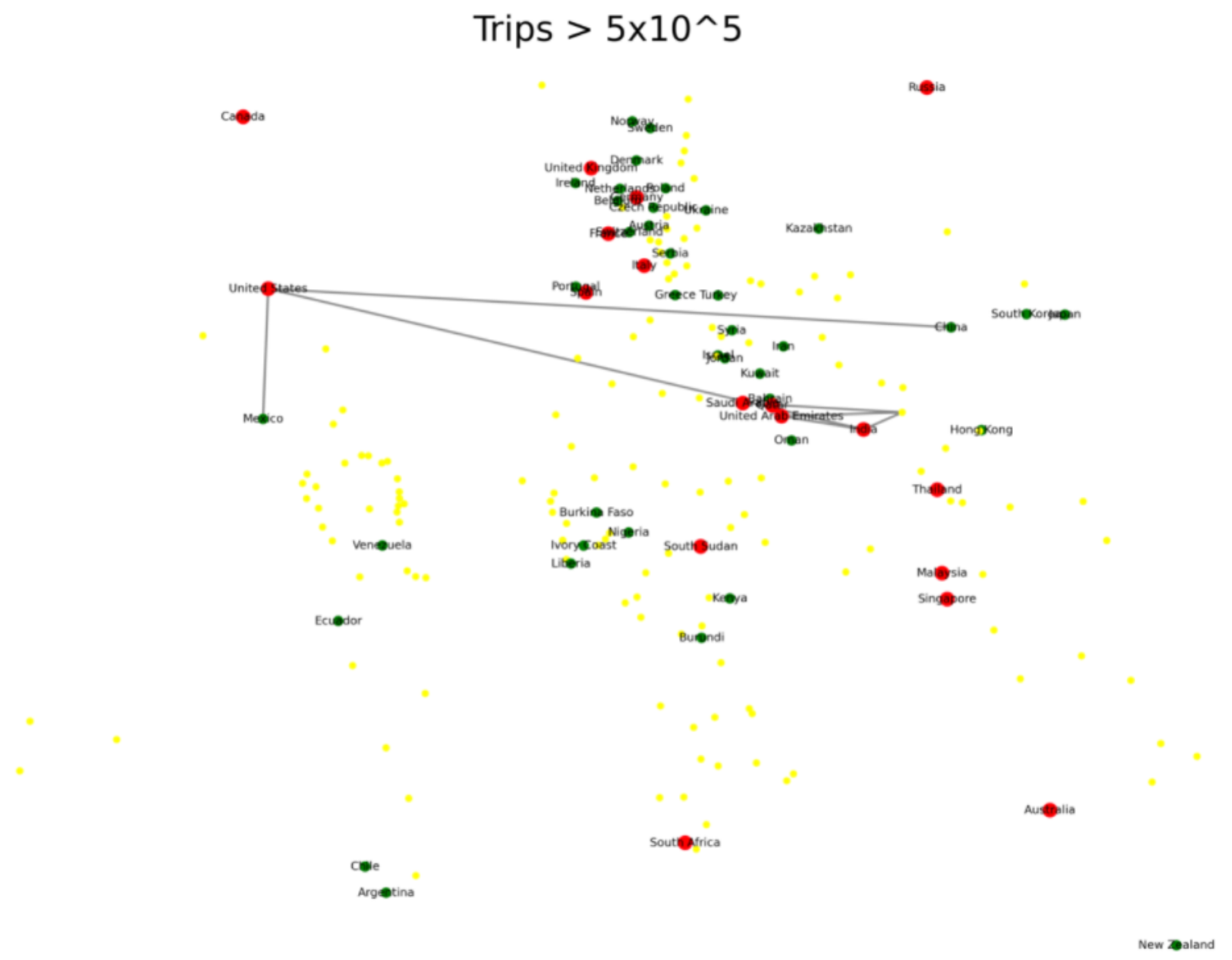
```python
#here we show the edgelist II
plt.figure(7)
plt.figure(figsize=(100,75))
nx.draw_networkx_edges(g, pos, edgelist = eII, width = 5 , alpha = 0.9, edge_color='orange')
nx.draw_networkx_nodes(g,pos=pos,node_size=ns,node_color=nc)
nx.draw_networkx_labels(g,pos,n_label,font_size=60,font_color='k')
plt.title('Trips between 10^5 and 5x10^5',fontsize=150)
plt.axis('off')
plt.savefig('map2.pdf', format='pdf',dpi=1000)
```

```
<Figure size 432x288 with 0 Axes>
```



```python
#here we show the edgelist I
plt.figure(8)
plt.figure(figsize=(100,75))
nx.draw_networkx_edges(g, pos, edgelist = eI, width = 10 , alpha = 0.9, edge_color='gray')
nx.draw_networkx_nodes(g,pos=pos,node_size=ns,node_color=nc)
nx.draw_networkx_labels(g,pos,n_label,font_size=50,font_color='k')
plt.title('Trips > 5x10^5',fontsize=150)
plt.axis('off')
plt.savefig('map1.pdf', format='pdf',dpi=1000)
```

```
<Figure size 432x288 with 0 Axes>
```



"This course has taught me to look at everything as a connected system, and I see that more and more each day as I continue to investigate issues. This training has been very useful for me to address issues of equity in many settings."

Mission accomplished, I can retire now... ☺️

Lec 8: March 8th	Migration Networks Weights and Directions (chapter 4) Due March 29th Assignment 3 (Project Preparation)
Lec 9: March 15th	Spatial Networks (Part 1)
March 22nd	Spring Recess
Lec 10: March 29th	Spatial Networks (Part 2) OSMNx + Project Ideas Discussions Due April 5th Assignment 4 ()
Lec 11: April 5th	Centrality Metrics and Resilience (chapter 3)
Lec 12: April 12th	Clustering Networks (chapter 6) Airport Networks + Project Ideas Discussions
Lec 13: April 19th	Dynamics on Networks (chapter 7) Twitter Paper + Project Ideas Discussions Due April 15th (project check-up)
Lec 14: April 26th	Gephi + Project Ideas Discussions
Final Presentations	Project Presentations (May 3rd, 4th and 5th)

Assignment 3

- Think about the implications of data analysis in relation to networks or spatial analysis. Propose a domain of interest
  to apply the tools learned in class.

- Search for related publications and data sets available for your study.

- Read your data