Utrecht University

School of Economics

Flowbca

A flow-based cluster algorithm in Stata

J. Meekes
W. H. J. Hassink

U.S.E. Discussion Paper Series nr: 17-09

How to reach the authors

Please direct all correspondence to the first author.

Jordy Meekes
Wolter H.J. Hassink
Utrecht University
Utrecht University School of Economics
Kriekenpitplein 21-22
3584 TC Utrecht
The Netherlands.
E-mail: J.Meekes@uu.nl
W.H.J.Hassink@uu.nl

Utrecht University School of Economics
Tjalling C. Koopmans Research Institute
Discussion Paper Series 17-09

Flowbca: A flow-based cluster algorithm in Stata

J. Meekes
W. H. J. Hassink

Utrecht School of Economics
Utrecht University

July 2017

currently under review by the Stata Journal

Abstract
In this article, we introduce the Stata implementation of a flow-based cluster algorithm written in Mata. The main purpose of the flowbca command is to identify clusters based on relational data of flows. We illustrate the command by providing multiple applications, from the research fields of economic geography, industrial input-output analysis, and social network analysis.

Keywords: clusters; aggregation; flows; regions; industries; economic geography; input-output analysis; social network analysis

JEL classification: C38; C43; C87; D57; D85; R23

Acknowledgements
We are grateful for the comments of seminar participants at the Utrecht University School of Economics. We also thank Rense Corten, Elena Fumagalli and Bastian Westbrock for insightful comments.

1 Introduction

In this article, we introduce the Stata implementation of a flow-based cluster algorithm, `flowbca`, written in Mata. Currently, flow-based cluster algorithms available in Stata focus on visualizing social networks (e.g. Corten 2011; Miura 2012). However, these algorithms lack the ability to flexibly aggregate units into clusters based on relational data of flows. The main motivation to write `flowbca` is that there is a need in many statistical applications, also in research fields other than social network analysis (SNA), for an algorithm to flexibly aggregate non-overlapping units into clusters. Specifically, as it provides a choice of how to operate clusters in empirical analyses and allows a researcher to compare alternative sets of clusters.

Given the increasing availability and use of relational data of various types of flows, `flowbca` can be of use to a variety of research fields. For example, the field of economic geography makes use of flows to cluster regional units into regional clusters of economic activity (Coombes, Green, and Openshaw 1986; Brezzi, Piacentini, Rosina, and Sanchez-Serra 2012).[1] Alternatively, industrial input-output analysis is based on trade linkages that register the flows of goods that are produced in one production chain and used as input in another production chain (Leontief 1986; Timmer, Dietzenbacher, Los, Stehrer, and de Vries 2015). Finally, SNA detects communities (Fortunato 2010) and identifies flow networks (Ford Jr and Fulkerson 1962; Beguerisse-Díaz, Garduño-Hernández, Vangelov, Yaliraki, and Barahona 2014) in graphs as connected groups based on the strength of flows between nodes.

`flowbca` is the implementation in Stata of a so-called agglomerative hierarchical clustering algorithm (Fortunato 2010) to identify clusters based on relational data of flows, which has been used in the aforementioned research fields. The key difference between `flowbca` and other agglomerative hierarchical clustering algorithms currently available in Stata is the focus on flow-based clustering instead of distance-based clustering.[2] In general terms, the flow-based cluster algorithm behind `flowbca` can be described as follows. It starts from a set of

1. Global regional clusters of economic activity are identified using trade flows (Smith and White 1992), foreign direct investment flows (Bathelt and Li 2014) or multinational firms relocation flows (Chen and Moore 2010) Within-country regional clusters such as local labor markets or local housing markets can be identified using commuting flows from place of residence to place of work, job-to-job turnover flows, household migration flows or job search flows (e.g. Duranton 2015).
2. Distance-based clustering uses the distance between a pair of units as a measure of similarity. Similar units are grouped into the same cluster and dissimilar units into separate clusters.

K disjoint units. The algorithm aggregates two units into one, considering the bilateral flows between the units. Clusters are identified by iteratively repeating this procedure. In each iteration of the algorithm, the decision criterion for aggregating two units into one is based on an optimization function selecting the maximum flow out of all bilateral flows. The source unit from which the largest flow starts is aggregated to the destination unit.

The algorithm is flexible in various aspects. First, the optimization function can be based on two definitions of flows, directed and undirected. The former refers to the maximum of the single directed flows that are flowing from one unit to another; the latter denotes the maximum of the sum of two bilateral flows. Second, the optimization function can be based on absolute flows and relative flows, which are computed by taking each absolute flow relative to the unit-specific total of outgoing flows. Third, the algorithm allows for flexibility in the stopping criterium by allowing for five optional ex ante user choices different from the default one. Using different options the researcher can thus create different sets of clusters.

After the algorithm has been terminated, the researcher could evaluate the choice of optimization function and stopping criterion by analyzing the level of self-containment of the set of clusters. The level of self-containment is approximated by the average of the internal relative flows. A higher average of the internal relative flows implies a stronger connectivity within each cluster and a weaker connectivity to outside clusters.

2 The flow-based cluster algorithm

2.1 The algorithm

The main inputs of the algorithm is a K-dimensional square matrix that contains the absolute flows between K different units. Effectively, each row represents a different source unit and each column represents a different destination unit.

The algorithm consists of the following five steps. The steps of the algorithm are provided given the default options of the algorithm that will be described in Section 3.

Step 1: the absolute flows between K different units are rewritten as a K-dimensional square matrix (i.e. an adjacency matrix in graph theory) of absolute flows $F^{(K)}$:

$$F^{(K)} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1K} \\ f_{21} & f_{22} & \cdots & f_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ f_{K1} & f_{K2} & \cdots & f_{KK} \end{bmatrix} \tag{1}$$

where f_{ij} ($i{\neq}j$) represents the flow in absolute term from source unit i to destination unit j. Flows f_{ii} are defined as the internal absolute flows.

Step 2: the matrix $F^{(K)}$ is rewritten in terms of relative flows as a K-dimensional square matrix $G^{(K)}$. Relative flows are computed by taking each absolute flow relative to the unit-specific total of outgoing flows. $G^{(K)}$ can be expressed as:

$$G^{(K)} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1K} \\ g_{21} & g_{22} & \cdots & g_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ g_{K1} & g_{K2} & \cdots & g_{KK} \end{bmatrix} \tag{2}$$

where

$$g_{ij} = f_{ij} / \sum_{t=1}^{K} f_{it} \qquad i, j = 1, 2, \ldots, K$$

Note that the matrix is row-normalized.

Step 3: the optimization function selects the arguments of the maximum directed relative flow from one unit to another, for $i{\neq}j$, of all $K \times (K-1)$ pairs of i and j:

$$(r, s) = \arg\max_{\substack{i,j \\ i \neq j}} g_{ij} \tag{3}$$

where units r and s are defined as the source unit and destination unit, respectively. If $g_{rs} = 0$ or $K = 1$, the default stopping criterion of the procedure is met and the algorithm is terminated.

Step 4: source unit r will be aggregated to destination unit s. The core of the cluster is defined as the receiving unit, i.e. destination unit s. To be able to adjust matrix $F^{(K)}$, a $K \times (K-1)$-dimensional matrix $C^{(K)}$ is specified. $C^{(K)}$ can be expressed as:

$$C^{(K)} = (e_1, e_2, \ldots, e_{r-1}, e_r + e_s, e_{r+1}, \ldots, e_{s-1}, e_{s+1}, \ldots, e_K) \tag{4}$$

where e_i represents the i-th unit column vector. For sake of convenience in the exposition of this algorithm, matrix $C^{(K)}$ is based on the assumption that the identifier value of unit r is larger than the identifier value of unit s.

Step 5: the absolute flows to and from units r and s will be added. The new matrix $F^{(K-1)}$ can be expressed as:

$$F^{(K-1)} = (C^{(K)})^T F^{(K)} C^{(K)} \tag{5}$$

where T refers to the transpose operator. $F^{(K-1)}$ is now a square matrix of dimension $(K-1)$. The algorithm continues with step 1 starting with $F^{(K-1)}$ as an input.

After the stopping criterion of step 3 has been met, the algorithm is terminated and K^* clusters are returned. The matrix of absolute flows between the K^* clusters, $F^{(K^*)}$, equals

$$F^{(K^*)} = C^T F^{(K)} C \tag{6}$$

where matrix C is a matrix product of the matrices $C^{(K)} \ldots C^{(K^*)}$, which can be expressed as $C = C^{(K)} C^{(K-1)} C^{(K-2)} \ldots C^{(K^*)}$.

2.2 Stopping criteria

In step 3, the stopping criterion of the algorithm is defined as $g_{rs} = 0$ or $K = 1$. The algorithm allows for five alternative stopping criteria, and each of them is a modification of the stopping criterion mentioned in step 3.

First, the researcher could specify a flow threshold q. The threshold q represents the minimum level of interaction at which a source unit is aggregated to a destination unit. The algorithm is terminated in step 3 if $g_{rs} < q$.

Second, the researcher could specify a minimum number of clusters k. The algorithm is terminated in step 3 if the number of units have reduced to this minimum, i.e. if $k = K^*$.

Third, the researcher could specify a minimum average of the internal relative flows l_a. The average of the internal relative flows L_a equals the sum of the internal relative flows g_{ii} relative to the number of clusters K^*:

$$L_a = \frac{1}{K^*} \sum_{i=1}^{K^*} g_{ii} \tag{7}$$

The algorithm is terminated in step 3 if $l_a \leq L_a$.

Fourth, the researcher could specify a minimum weighted average of the internal relative flows l_w. The weighted average of the internal relative flows L_w

equals the sum of the internal absolute flows relative to the sum of all absolute flows:

$$L_w = \frac{1}{N} \sum_{i=1}^{K^*} f_{ii} \tag{8}$$

for which the sum of all absolute flows equals $N = \sum_{i=1}^{K^*} \sum_{j=1}^{K^*} f_{ij}$. The algorithm is terminated in step 3 if $l_w \leq L_w$.

Finally, the researcher could impose a minimum internal relative flow l_m that all of the clusters must satisfy. The minimum of the internal relative flows L_m can be expressed as:

$$L_m = \min_i g_{ii} \tag{9}$$

The algorithm is terminated in step 3 if $l_m \leq L_m$.

2.3 An alternative optimization function

The algorithm provides two different optimization functions, which are based either on the directed or on the undirected flows approach. The optimization function based on the directed flows selects arg max g_{ij}, considering the maximum of the directed flows from one unit to another. In contrast, the optimization function based on the undirected flows approach selects arg max $g_{ij} + g_{ji}$, considering the maximum of the sum of two bilateral flows, which can be expressed as:

$$(r, s) = \arg\max_{\substack{i,j \\ i \neq j}} g_{ij} + g_{ji} \tag{10}$$

Using the undirected flows approach, the algorithm is terminated if $g_{rs} + g_{sr} = 0$ or $K = 1$. Otherwise, the procedure will continue with step 4 of the algorithm. Note that the new cluster gets the identification number of the unit with the largest incoming flow, which represents the core of the new cluster.

2.4 Some caveats

In step 3 it might be the case that $\arg\max g_{ij}$ holds for multiple pairs of units i, j. Consequently, the pair r, s will not be unique. The following rules are imposed to close the algorithm:

1. If there are two or more source units r, e.g. r_1 and r_2, that both have the maximum flow to the same destination unit s, r_1 is aggregated to s if r_1 has the highest incoming flow from the other source unit(s) r.

2. If source unit r has identical flows to two or more units s, e.g. s_1 and s_2, r is aggregated to s_1 if s_1 has the highest incoming flow from the other destination unit(s) s.

3. If both r and s are not unique, e.g. there are two pairs r_1,s_1 and r_2,s_2, the algorithm aggregates r_1 to s_1 if s_1 has the highest incoming flow from the other destination unit(s) s.

4. For the iterations where a unique pair is still not identified, the algorithm picks one pair r, s of all pairs that correspond to the maximum flow.

3 The flowbca command

3.1 Syntax

`flowbca` *varname varlist* [`, q(`#`) k(`#`) la(`*fraction*`) lw(`*fraction*`) lm(`*fraction*`)` `opt_f(`#`) save_k`]

3.2 Description

`flowbca` implements the algorithm that is discussed in Section 2 and performs it in Mata. The main inputs for `flowbca` are the variables listed in *varname* and *varlist*. *varname* contains one variable representing the source unit identifier. This variable should be numerical, as string variables are ignored by the `flowbca` command. *varlist* contains a set of variables, one variable for each distinct destination unit, representing the absolute flows from the source units to the destination units.

Effectively, the destination unit variables represent the columns of a K-dimensional square matrix of flows between the K units. For example, the value of the first observation of a destination unit variable represents the absolute flow from the first source unit to the corresponding destination unit. The source and destination units should be numbered such that if they are sorted and ordered in a sequential order, the diagonal elements of the K-dimensional square matrix represent the internal absolute flows. If the flow data of the researcher is only available in an $K \times 3$-dimensional matrix in which there are three columns that represent the source unit identifier, destination unit identifier and absolute flows between the units, respectively, the data should be reshaped by the user into a K-dimensional square matrix.

3.3 Option

`q(#)` sets the flow threshold. To set a relative flow threshold, place a fraction in parentheses after q. To set an absolute threshold, place an integer number in parentheses after `q`. If the threshold is higher than the maximum of all flows, the stopping criterion of the procedure has been met and the algorithm is terminated. The default is to have a flow threshold equal to zero.

`k(#)` specifies the number of distinct clusters the algorithm should identify. The default is to identify one cluster.

`la(`*fraction*`)` specifies the minimum average of the internal relative flows. If the fraction is lower than or equal to the average of the internal relative flows, the stopping criterion of the procedure is met. The default is no minimum average of the internal relative flows.

`lw(`*fraction*`)` specifies the minimum weighted average of the internal relative flows. If the fraction is lower than or equal to the weighted average of the internal relative flows, the stopping criterion of the procedure is met. The default is no minimum weighted average of the internal relative flows.

`lm(`*fraction*`)` specifies the minimum internal relative flow. If the fraction is smaller than or equal to the minimum value of the internal relative flows, the stopping criterion of the procedure is met. The default is no minimum internal relative flow.

`opt_f(#)` specifies the optimization function. Four optimization functions can be chosen. The default is `opt_f(1)`, implementing the directed relative flows approach. The other are: `opt_f(2)` implementing the undirected relative flows approach; `opt_f(3)` implementing the directed absolute flows approach; `opt_f(4)` implementing the undirected absolute flows approach.

`save_k` is an option to save the cluster_setk data sets (see below). For each k, the data set contains the absolute flows between the remaining k units (i.e. the matrix $F^{(K)}$ for each k). To save the data sets, place `save_k`.

3.4 Output

`flowbca` saves three data sets.[3]

3. We suggest that the researcher creates a data set that consists of the variable sourceunit and a variable that represents the source unit labels. This data set could be merged to the data sets cluster_set and unit_set.

1. cluster_set containing variables that characterize the identified clusters, including variables that represent the cluster identifier (clusterid), the cluster-specific internal relative flow (internal), the average of the internal relative flows (La), the weighted average of the internal relative flows (Lw), the minimum of the internal relative flows (Lm), the cluster-specific total value of outgoing flows (rowflows), the total value of all flows (N) and a set of variables that represents the flows between all clusters (destinationunit).

2. unit_set containing variables that provide information about each starting unit, including variables that represent the source unit identifier (sourceunit), the cluster the unit is aggregated to (clusterid), the relative flow at the iteration the source unit is aggregated to a destination unit (g), the number of distinct clusters that were remaining after aggregating the source unit (round) and an indicator variable that equals zero if the unit is the core of a cluster and zero otherwise (core).

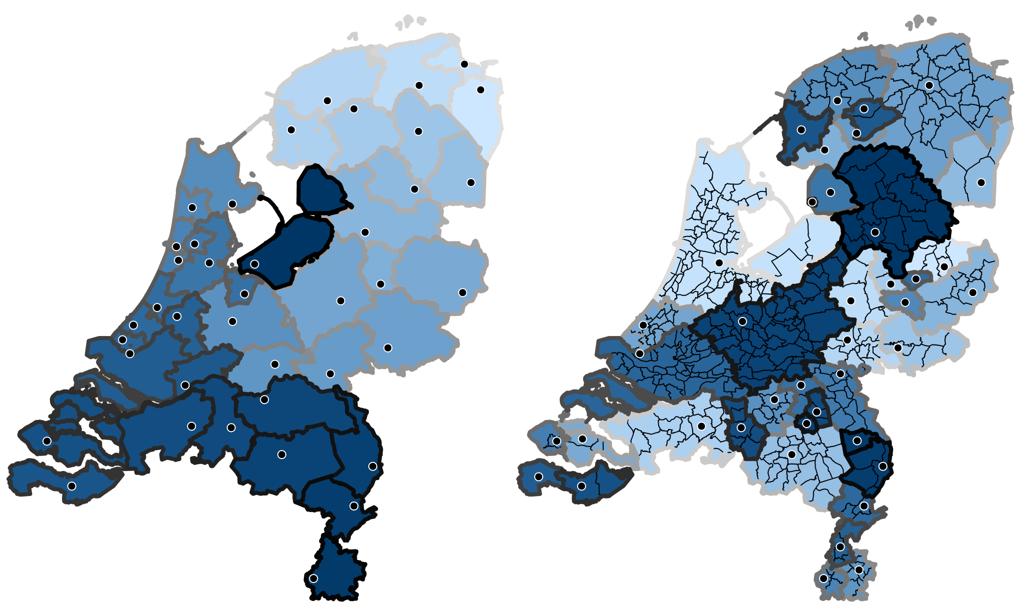
3. cluster_setk containing the source unit identifier variable (sourceunit), and one variable for each destination unit representing the absolute flows (destinationunit). If the researcher uses the option save_k, the cluster_setk data sets will be saved.

4 Applications

4.1 Application 1: Within-country regional clusters based on commuting flows

In the first application, a researcher uses `flowbca` to construct regional clusters based on individuals′ commuting flows from place of residence to place of work. The researcher aims to compare the levels of self-containment of forty NUTS 3 areas and twelve provinces to the levels of self-containment of forty and twelve clusters identified using `flowbca`, respectively. The Dutch NUTS 3 areas offer an interesting point of comparison, as they were identified, in 1970, based on journey-to-work and place-of-work statistics that reflected the employment outcomes and commuting behaviour of the Dutch population. Moreover, in research on European countries, NUTS 3 areas are often used as the regional classification to operate regional clusters (e.g. Ciccone 2002).

For this application, aggregate data on 7,131,000 commuting flows in 2014 were used, at the municipality level, retrieved from the CBS StatLine open databank of Statistics Netherlands (CBS 2017).[4] The algorithm starts from a set of 398 municipalities (K).[5] Note that this application uses the option `k()` of `flowbca`, as the researcher aims to identify a specific number of clusters. The optimization function is based on the directed relative flows approach. The directed flows approach was used, as commuting flows are by nature directed in the sense that they flow from one unit to another. Relative flows are preferred to absolute flows, as relative flows function as weights to account for the relative importance of a unit that allows smaller source units to be able to aggregate to bigger destination units. To visualize the identified clusters, the Stata commands mergepoly (Picard and Stepner 2015) and spmap (Pisati 2008) were used.



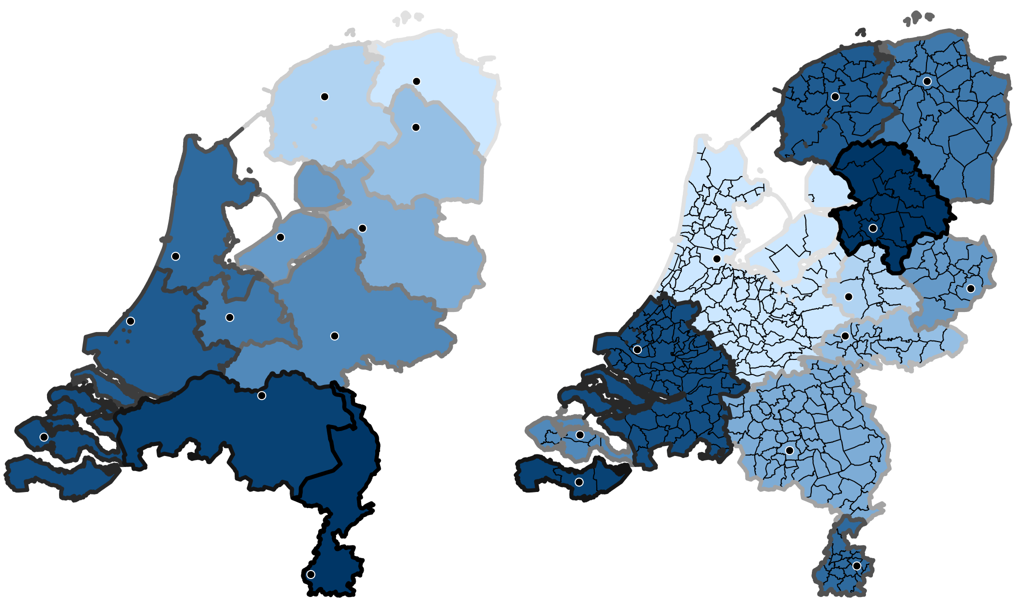
(a) NUTS 3 areas. $L_a = .699$; $L_w = .743$; $L_m = .462$; $K^* = 40$

(b) Forty clusters. $L_a = .695$; $L_w = .805$; $L_m = .385$; $K^* = 40$; $K = 398$

Figure 1: NUTS 3 areas and forty identified clusters. Notes: The NUTS 3 cores (black dots with a white circle) are defined as the municipality with the highest number of residents. The cores of the identified regional clusters are returned by the algorithm. Each distinct cluster is surrounded by a thick border and highlighted by a different color.

4. Note that the researcher could also exploit microdata to construct clusters. For instance, subgroup-specific clusters could be identified using subgroup-specific flows (Farmer and Fotheringham 2011).
5. Note that five municipalities, which represent small Wadden islands in the northern part of the Netherlands, were removed. These municipalities were removed as they would be identified as small self-contained clusters that artificially increase the average of the internal relative flows (L_a).

Figure 1 shows that the weighted average of the internal relative flows (L_w) of the forty identified regional clusters (see Figure 1(b)) is higher than the weighted average of the NUTS 3 areas (see Figure 1(a)). This implies that a higher number of individuals live and work in their identified cluster compared to their NUTS 3 area. Figure 2 shows that the average of the internal relative flows (L_a), the weighted average of the internal relative flows (L_w) and the minimum of the internal relative flows (L_m) are higher in the case of the twelve identified clusters than of the twelve pre-defined administrative provincial areas. All in all, this application shows that `flowbca` can be used to identify meaningful regional clusters that are characterized by a relatively high level of self-containment.



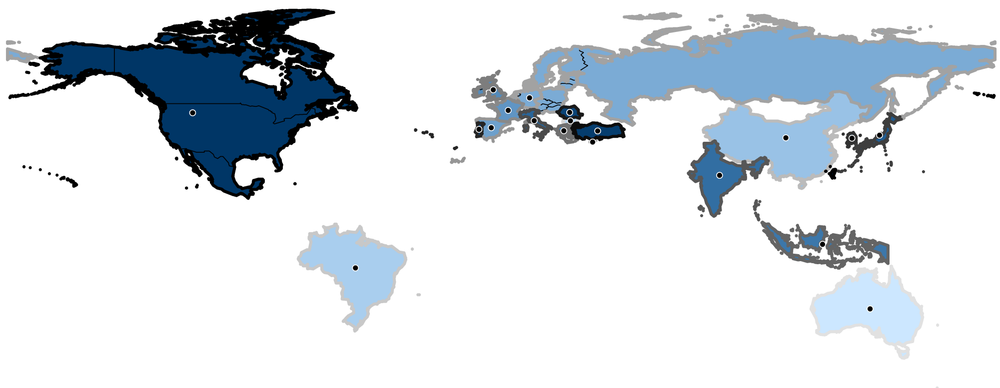
(a) Provial areas. $L_a = .833$; $L_w = .866$; $L_m = .570$; $K^* = 12$

(b) Twelve clusters. $L_a = .869$; $L_w = .901$; $L_m = .776$; $K^* = 12$; $K = 398$

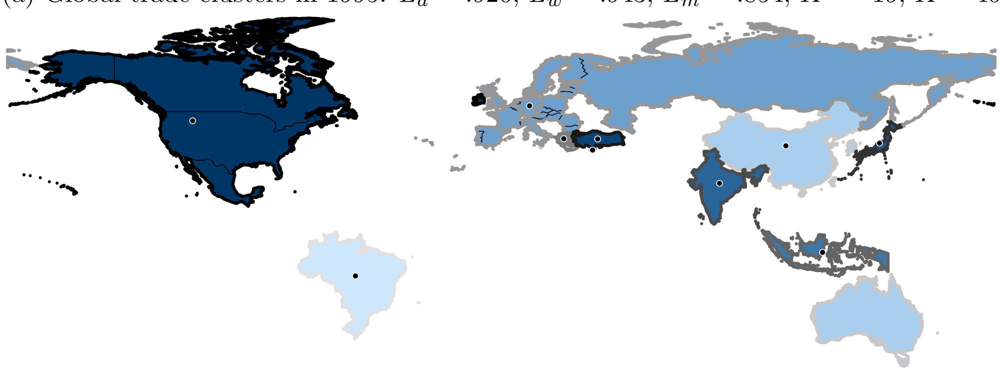
Figure 2: Provinces and twelve identified clusters. Notes: The provincial cores (black dots with a white circle) are the capital cities. The cores of the identified regional clusters are returned by the algorithm. Each distinct cluster is surrounded by a thick border and highlighted by a different color.

4.2 Application 2: Global regional clusters based on national trade flows

In the second application, a researcher uses `flowbca` to identify global regional clusters based on trade flows. The researcher aims to examine how interrelated countries are in terms of trade, and whether this relatedness changed over time. The researcher identifies global clusters of economic activity for the years 1995 and 2011, given a minimum flow threshold which is set equal to five per cent.



(a) Global trade clusters in 1995. $L_a = .920$; $L_w = .943$; $L_m = .854$; $K^* = 19$; $K = 40$



(b) Global trade clusters in 2011. $L_a = .938$; $L_w = .947$; $L_m = .898$; $K^* = 10$; $K = 40$

Figure 3: Global clusters based on trade flows. Notes: The cores of the identified clusters (black dots with a white circle) are returned by the algorithm. Each distinct cluster is highlighted by a different color. Global clusters were identified based on trade flows expressed in millions of dollars between countries from the 1995 and 2011 WIOD data. The flow threshold `q` was set equal to five per cent. Trade flows between countries were computed by aggregating all within-country flows. Nine countries with zero or negative flows were removed.

For this application, the World Input-Output Database (WIOD) was used (Dietzenbacher, Los, Stehrer, Timmer, and de Vries 2013). This data set consists of data on the trade flows between forty countries in the period 1995-2011. The algorithm starts from a set of forty countries (K). Note that this exercise uses the option `q()` of `flowbca`. The optimization function is based on the directed relative flows approach.

The set of clusters in 1995 and 2011 are compared to examine the change in global trade clusters over time. Figure 3 shows, given the flow threshold of five per cent, that the number of distinct global clusters decreases from nineteen to ten over the period 1995-2011. Consistent with globalization, the decrease in the number of identified clusters suggests that the trade flows within countries decreased relative to the trade flows between countries. Another observation is that the size of the two clusters in which China and Germany is the core, respectively, became larger over time.

4.3 Application 3: A social network based on friendship ties

In the third application, a researcher uses `flowbca` to detect groups of prison inmates based on friendship ties. The researcher aims to identify groups of inmates in which each group should have a minimum internal relative flow of at least fifty per cent. The minimum internal relative flow of fifty per cent implies that, in each group, at least fifty per cent of the inmates′ friendship ties should be with inmates in their own group.

For this application, the Gagnon and MacRae prison friendship data set was used (MacRae 1960). The level of interaction between inmates is approximated by multiple zero-one indicator variables that represent friendship ties, which equal one if a given "source" inmate indicates a friendship with a given "destination" inmate and zero otherwise. The algorithm starts from a set of 67 inmates (K). All inmates could indicate as few or as many friendship ties as desired. Inmates could not indicate a friendship with themselves. Note that this application utilizes the option `lm()` of `flowbca`. Relative flows were used to have the relative importance of each tie.

Table 1 provides information about the groups of inmates that were identified using `flowbca` for both the directed and undirected flows approach, respectively. The results show that the directed flows approach, compared to the undirected flows approach, leads to fewer and bigger groups of inmates. Another observation is that there are more isolated inmates if the optimization function is based on the undirected flows approach. Effectively, the undirected flows approach puts more weight on the situation where two inmates indicate each other as friend, and leads to more sparse groups.

Table 1: Number and size of the identified groups of inmates

Directed flows approach		Undirected flows approach	
Number of groups for a given size	Size (in # of inmates)	Number of groups for a given size	Size (in # of inmates)
1	38	1	11
1	12	1	9
1	7	1	7
1	5	1	6
1	4	3	5
		2	4
		1	3
		2	2
$L_a = .847$		$L_a = .689$	
$L_w = .846$		$L_w = .676$	
$L_m = .758$		$L_m = .5$	
$K^* = 5$		$K^* = 12$	
$K = 67$		$K = 67$	

Notes: The connected groups of inmates are based on friendship ties between inmates from the Gagnon and MacRae prison data set. The minimum internal relative flow `lm`, which each group should satisfy, was set equal to fifty per cent. The raw prison data set contains 67 inmates. No inmate was disconnected, i.e. the situation that an inmate did not specify nor was specified as friend by another inmate. However, 1 inmate and 4 inmates were identified as isolated using the directed and undirected flows approach, respectively. The isolated inmates were removed.

4.4 Application 4: Industrial clusters based on input-output flows

The final application is a case where a researcher uses `flowbca` to identify five U.S. industrial clusters based on input-output flows of goods between U.S. industries. The researcher examines whether the relatedness between industries changed over time, by comparing the set of industrial clusters in 1995 to the set of clusters in 2011. The WIOD was used to identify U.S. industrial clusters in the years 1995 and 2011. The algorithm starts from a set of 35 industries (K). Note that this exercise utilizes the option `k()` of `flowbca`. The optimization function is based on the directed relative flows approach.

Table 2 presents the results of application 4. The sectors "Construction" and "Public Administration and Defence; Compulsory Social Security" are the largest industrial clusters in 1995 and 2011, respectively. Hence, in 2011, the industry "Public Administration and Defence; Compulsory Social Security" uses relatively more goods that are produced in other production chains than the industry "Construction". Interestingly, `flowbca` identifies the industries "Food, Beverages and Tobacco", "Textiles and Textile Products" and "Transport Equipment" as the core of a cluster in both 1995 and 2011, which suggests that these clusters have been relatively self-contained over time.

Table 2: Industrial clusters based on U.S. input-output flows

1995		2011	
Core of cluster	Size (in # of units)	Core of cluster	Size (in # of units)
Construction	28	Public Administration and Defence; Compulsory Social Security	30
Food, Beverages and Tobacco	3	Food, Beverages and Tobacco	2
Textiles and Textile Products	2	Chemicals and Chemical Products	1
Chemicals and Chemical Products	1	Basic Metals and Fabricated Metal	1
Transport Equipment	1	Transport Equipment	1
$L_a = .602$		$L_a = .567$	
$L_w = .828$		$L_w = .837$	
$L_m = .335$		$L_m = .287$	
$K^* = 5$		$K^* = 5$	
$K = 35$		$K = 35$	

Notes: U.S. industrial clusters based on U.S. input-output flows of goods expressed in millions of dollars between 35 ISIC industries from the WIOD data. The minimum number of clusters `k` was set equal to five.

As Table 2 shows, the largest cluster is composed of many units and the other clusters are composed of very few units. Application 4 highlights the main limitation of `flowbca`. The main limitation of `flowbca` is that the algorithm identifies one relatively big cluster that is composed of many units, if the network is not sparse enough and thus not composed of multiple subnetworks. For example, consider the case that a researcher aims to identify clusters using random flows between units. It is likely that in each iteration a source unit is aggregated to the identical destination unit, as the destination unit represents a relatively big cluster due to the aggregations in the earlier iterations.

All in all, to identify meaningful clusters, the network should be sparse enough, e.g. by distance (in the case of within-country regional clusters), input-output flows of goods (in the case of industrial clusters) or social interaction (in the case of SNA). Otherwise, the algorithm will identify one relatively big cluster and several distinct clusters of very few units. Note that the use of more disaggregated units in the starting set of units would improve the accuracy of the cluster algorithm, as more detailed flow data is used.

5 Concluding remarks

In this article, we have introduced and illustrated the `flowbca` Stata command that can be used to identify clusters based on relational data of flows between disjoint units. Four applications in a wide range of research fields were provided to illustrate `flowbca` cluster identification capabilities. Given the increasing availability of relational data of various types of flows, `flowbca` can be of use to a variety of research fields. `flowbca` is flexible, because it allows for various optimization functions and stopping criteria. The program is accessible for the researcher, which will hopefully lead to a further development of the algorithm. Overall, the program is robust, user-friendly and well able to identify clusters that are characterized by a high level of self-containment.

6 Acknowledgements

We are grateful for the comments of seminar participants at the Utrecht University School of Economics. We also thank Rense Corten, Elena Fumagalli and Bastian Westbrock for insightful comments.

7 References

Bathelt, H., and P.-F. Li. 2014. Global cluster networks–foreign direct investment flows from Canada to China. *Journal of Economic Geography* 14(1): 45–71.

Beguerisse-Díaz, M., G. Garduño-Hernández, B. Vangelov, S. N. Yaliraki, and M. Barahona. 2014. Interest communities and flow roles in directed networks: the Twitter network of the UK riots. *Journal of The Royal Society Interface* 11(101): 1–12.

Brezzi, M., M. Piacentini, K. Rosina, and D. Sanchez-Serra. 2012. Redefining urban areas in OECD countries. In *Redefining "Urban"*, 19–58. Organisation for Economic Co-operation and Development. http://www.oecd-ilibrary.org/content/chapter/9789264174108-4-en.

CBS. 2017. CBS Statline. Centraal Bureau voor de Statistiek, Den Haag/Heerlen.

Chen, M. X., and M. O. Moore. 2010. Location decision of heterogeneous multinational firms. *Journal of International Economics* 80(2): 188–199.

Ciccone, A. 2002. Agglomeration effects in Europe. *European Economic Review* 46(2): 213–227.

Coombes, M. G., A. E. Green, and S. Openshaw. 1986. An efficient algorithm to generate official statistical reporting areas: The case of the 1984 travel-to-work areas revision in Britain. *Journal of the Operational Research Society* 37(10): 943–953.

Corten, R. 2011. Visualization of social networks in Stata using multidimensional scaling. *Stata Journal* 11(1): 52–63.

Dietzenbacher, E., B. Los, R. Stehrer, M. Timmer, and G. de Vries. 2013. The construction of world input–output tables in the WIOD project. *Economic Systems Research* 25(1): 71–98.

Duranton, G. 2015. Delineating metropolitan areas: Measuring spatial labour market networks through commuting patterns. In *The Economics of Interfirm Networks*, ed. T. Watanabe, I. Uesugi, and A. Ono, 107–133. No. 4 in Advances in Japanese Business and Economics, Springer Japan.

Farmer, C. J. Q., and A. S. Fotheringham. 2011. Network-based functional regions. *Environment and Planning A* 43(11): 2723–2741.

Ford Jr, L. R., and D. R. Fulkerson. 1962. *Flows in Networks*. Princeton University Press.

Fortunato, S. 2010. Community detection in graphs. *Physics Reports* 486(35): 75–174.

Leontief, W. W. 1986. *Input-output Economics*. Oxford University Press.

MacRae, D. 1960. Direct factor analysis of sociometric data. *Sociometry* 23(4): 360–371.

Miura, H. 2012. Stata graph library for network analysis. *Stata Journal* 12(1): 94–129.

Picard, R., and M. Stepner. 2015. MERGEPOLY: Stata module to merge adjacent polygons from a shapefile. https://ideas.repec.org/c/boc/bocode/s457574.html.

Pisati, M. 2008. SPMAP: Stata module to visualize spatial data. https://ideas.repec.org/c/boc/bocode/s456812.html.

Smith, D. A., and D. R. White. 1992. Structure and dynamics of the global economy: Network analysis of international trade 1965-1980. *Social Forces* 70(4): 857–893.

Timmer, M. P., E. Dietzenbacher, B. Los, R. Stehrer, and G. J. de Vries. 2015. An illustrated user guide to the World Input–Output Database: The case of global automotive production. *Review of International Economics* 23(3): 575–605.

About the authors

Jordy Meekes is a doctoral candidate in the Department of Economics at Utrecht University, the Netherlands.

Wolter H. J. Hassink is a professor of applied econometrics in the Department of Economics at Utrecht University and a research fellow of the IZA institute of labor economics.