Advances in Methodology and Statistics / Metodološki zvezki, Vol. 18, No. 1, 2021, 17–30
https://doi.org/10.51936/kcpy2449

Comparing different methods for one-mode homogeneity blockmodeling according to structural equivalence on binary networks

Aleš Žiberna

University of Ljubljana, Faculty of Social Sciences, Ljubljana, Slovenia

Abstract

One-mode homogeneity blockmodeling is an approach to clustering networks that searches for partitions of units in a network so that the resulting blocks are as homogeneous as possible. Block is a part of the network that contain (possible) ties from the units of one cluster to the units of another cluster (or ties within a cluster). Typically, sum of squared deviations from the mean is taken as the measure of variability (non-homogeneity). The paper presents the results of a simulation study that applied several methods for this problem to binary networks generated according to structural equivalence. Several versions of homogeneity generalized blockmodeling (using a relocation algorithm), a *k*-means-based algorithm, and an indirect approach are compared. Since all of the methods being compared try to optimize the same criterion function, this and the Adjusted Rand Index are the main criteria for the comparison. All methods (except the indirect approach, which is not iterative) were given the same amount of time to find the best possible solution. The overall conclusion is that the *k*-means approach is advised in most cases, except when smaller networks (200 units) are being partitioned into larger number of clusters, in which case the homogeneity generalized blockmodeling is preferred.

Keywords: blockmodeling, sum of squares, generalized blockmodeling, *k*-means approach, indirect approach

1. Introduction

One-mode homogeneity blockmodeling (Žiberna, 2007) is an approach for clustering networks. It searches for partitions of units in a network so that the resulting blocks, namely, the parts of the network that contain (possible) ties from the units of one group to the units of another group (or ties within a group), are as homogeneous as possible (where some measure of their variability is as low as possible). Typically, the sum of squared deviations from

*Corresponding author
Email address: ales.ziberna@fdv.uni-lj.si (Aleš Žiberna)
ORCID iD: 0000-0003-1534-6971 (Aleš Žiberna)

the mean (sum of squares for short) is taken as the measure of variability and hence only this measure is considered here since then *k*-means-like (Žiberna, 2020) approaches may be used. The aim of this paper is to investigate which strategies are best for finding the ideal partition using this criterion. Here, *k*-means and generalized blockmodeling approaches are considered, together with various strategies for certain aspects of the relocation algorithm used within generalized blockmodeling and the possibility of combining the two approaches. In addition, an indirect approach to blockmodeling is examined. Results of a similar simulation as presented here, but based on linked networks and without the indirect approach, has already been presented (Žiberna, 2020). However, given that most users of homogeneity blockmodeling approaches analyze one-mode networks, I believe these results are more relevant for most cases while they also incorporate use of an indirect approach. In this paper, only "simple" one-mode binary directed networks without loops (self-ties) are examined, that is, those which contain only one set of units and only one directed relation.

2. Blockmodeling criteria and approaches

Blockmodeling is a clustering method for clustering units in a network. Its aim is to partition network units into clusters and, at the same time, to partition the set of ties into blocks (Doreian et al., 2005, p. 29). Blockmodeling may also be "[v]iewed as a method of data reduction, [...] a valuable technique in which redundant elements in an observed system are reduced to yield a simplified model of relationships among types of elements (units)" (Borgatti & Everett, 1992). Several approaches to blockmodeling exist. One possible division entails stochastic approaches, namely stochastic blockmodeling (Anderson et al., 1992; Holland et al., 1983; Snijders & Nowicki, 1997), and deterministic approaches.

Only selected deterministic approaches are considered in this text because the aim is to only look at approaches that minimize the sum of squares within blocks. Deterministic approaches may be further split into indirect approaches1 (e.g. Breiger et al., 1975; Burt, 1976; see Doreian et al., 2005, pp. 25–26 for definition) and direct approaches. In this paper, one indirect and two direct approaches are examined. The indirect approach used relies on corrected squared Euclidian distance (Batagelj et al., 1992) and Ward hierarchical clustering (Ward, 1963). The direct approaches used are generalized blockmodeling (Doreian et al., 1994, 2005; Žiberna, 2007) and *k*-means based blockmodeling approaches (Brusco & Doreian, 2015a, 2015b; Žiberna, 2020). In the subsections below, the sum of squares criterion is presented first, followed by the approaches that are being considered.

2.1. Sum of squares criterion

In this paper, the methods aim to minimize within-block sum of squared deviations from the mean. They thereby aim to obtain the partition that induces blocks whose values are as homogenous as possible.

For the binary networks under study here, ideal blocks (those leading to value 0 of a criterion function) include complete blocks (all possible ties are present) and null blocks (no ties are present). However, such situations are rarely found in real networks unless the number of clusters is large relative to the number of units. Therefore, for binary networks, this criterion essentially means we are looking for blocks where the densities of the rows and likewise of the columns are as similar as possible to each other.

More precisely, the criterion function that is optimized is:

$$f(P) = \sum_{k=1}^{K} \sum_{l=1}^{K} v_{kl}$$

where v_{kl} is the sum of squared deviations from the mean for block kl. The notation is explained in Table 1.

Table 1: Nomenclature

n	number of units
U	a set of n units
$\mathbf{X}$	a real or binary matrix with elements x_{ij} and with dimensions $n \times n$
K	number of clusters
P	a partition of all units, $P = \{C_1, ..., C_K\}$, $\cup_{i=1}^{K} C_i = U$
Π	a set of all possible partitions
$\bar{x}$	the grand mean of $\mathbf{X}$, $\bar{x} = \sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} \frac{x_{ij}}{(n(n-1))}$ assuming to loops (self-ties)
$\bar{\mathbf{X}}$	a matrix with dimensions $K \times K$ with elements $\bar{x}_{kl}$ representing the mean of the block from cluster C_k to cluster C_l, where $\bar{x}_{kl} = \frac{\sum_{i \in C_k} \sum_{j \in (C_l \setminus i)} x_{ij}}{\lvert C_k \rvert \cdot (\lvert C_l \rvert - \delta(k,l))}$
$\delta(l,k)$	a function that returns 1 if its arguments are equal, 0 otherwise
$\mathbf{V}$	a matrix with dimensions $K \times K$ with elements $v_{kl} = \sum_{i \in C_k} \sum_{j \in \{C_l \setminus\}} \left(x_{ij} - \bar{x}_{kl}\right)^2$, the within-block sum of squares for block from cluster C_k to cluster C_l

2.2. Indirect blockmodeling

As mentioned, the indirect approach used is Ward's hierarchical clustering (Ward, 1963) applied to the corrected squared Euclidian distance (Batagelj et al., 1992) since, at least on classical data, this combination is often seen as a (usually inferior) alternative to homogeneity approaches based local optimization of sum of squares criteria, e.g. the k-means approach (Steinley, 2003). Some other combinations of distance (using "unsquared" corrected Euclidian distance) and other hierarchical methods (complete, single) are also used for comparison purposes.

This approach is not the focus of this research, but included to also check whether in the case of homogeneity criteria the partitions obtained from indirect approaches are inferior to those from direct ones as observed for the criterion function counting the number of inconsistencies (Batagelj et al., 1992).

2.3. Generalized blockmodeling

It was noted that the first direct approach to be used is generalized blockmodeling (Doreian et al., 2005). This is a very flexible approach for blockmodeling, as it can not only cluster units according to one type of equivalence, such as structural (Lorrain & White, 1971) or regular (White & Reitz, 1983) equivalence, but can be used with any equivalence that can be

expressed by specifying a set of allowed block types. While initially designed for only binary one-mode networks (Doreian et al., 1994), it was later extended to various other network types (Doreian et al., 2004; Doreian & Mrvar, 2009). For this paper, the extension to valued networks called sum of squares homogeneity blockmodeling (Žiberna, 2007) is especially important because this minimizes the sum of squares within-blocks criteria, which is the criterion used in this paper.

Particularly for the purpose of optimizing the sum of squares criteria introduced in Subsection 2.1, this approach basically amounts to using a relocation algorithm for optimizing this criterion. Since it is only a locally-optimal algorithm, the whole procedure is repeated (restarted) with several (typically randomly selected) starting points. Implementation of the relocation algorithm used here involves two possible "relocations":

1. *Transfer*: Transferring a unit from one cluster to another (if the unit is not the only unit in its cluster);
2. *Exchange*: Exchanging two units from different clusters.

The first option (transfer) is always used because it is relatively fast. The second option (exchange) on the other hand uses much more time. For binary two-mode blockmodeling has been shown (Brusco & Steinley, 2011) that it is better to use more restarts (repeat the algorithm with a new random starting point) without exchanges than to try to better optimize each partition using this option.

Another implementation detail is whether all allowed relocations (so either just transfers or moves and transfers) are first evaluated and the partition that improves the criterion function the most is selected as the partition for use in the next iteration.

The other possibility is that when the first partition which improves the criterion function is found, the algorithm moves to this new partition. When this option is used, the order in which the relocations are performed is very important. To prevent bias and allow as diverse search as possible, the order of relocations was random.[1]

2.4. k-means blockmodeling

k-means blockmodeling has its roots in the *k*-means algorithm (Hartigan, 1975; Hartigan & Wong, 1979; MacQueen, 1967), a well-known algorithm for partitioning units in "classical" data (units by variables) by minimizing the within-cluster sum of squares. The algorithm is locally-optimal. A thorough review is given by e.g. Steinley (2006). The original *k*-means algorithm therefore partitions one mode of a two-mode valued matrix. The method was first adapted to the two-mode clustering of two-mode matrices or networks (Baier et al., 1997; Brusco & Doreian, 2015a, 2015b; Van Mechelen et al., 2004; van Rosmalen et al., 2009; Vichi, 2001) and recently to the one-mode clustering of one-mode matrices/networks (Žiberna, 2020). This last variant of the algorithm is evaluated in this paper. The *k*-means algorithm can be initialized either with a random partition or with random centroids. The first variant is relied on here.

[1]For faster performance, the order was not completely random. In each iteration, the program goes through all clusters in random order. Within each cluster, it goes through all units from this cluster in random order. For each unit, the program then goes through all remaining clusters in random order. For each cluster, it first checks if the transfer of a selected unit to this cluster improves the criterion function. If it does not, the exchanges are allowed, and the second cluster has a higher id than the first one (to avoid examining the same exchange twice), it goes through all units in this cluster and evaluates it if the exchange of these two units (from different clusters) improves the criterion function. Whenever an improvement is found, the transfer or exchange is executed and the new partition is taken as the starting partition for the next iteration. All "random orders" are different in each iteration.

2.5. k-means plus relocation

Späth (1980, as cited in Steinley, 2006, p. 3) noted the results of the original *k*-means algorithm can sometimes (although presumably rarely in practice) be improved by relocating a single unit from one cluster to another. Similarly was found for the algorithm for the two-mode partitioning of two-mode networks (Brusco & Steinley, 2007). As sum of squares homogeneity blockmodeling optimizes the criterion function using a relocation algorithm as the *k*-means approach for one-mode networks, this relocation-based algorithm is used here in an attempt to improve the *k*-means solution. The fact that the generalized blockmodeling algorithm is relatively slow is not as problematic because in most cases optimizing a single good partition is relatively quick.

3. Simulation study design

The aim of this simulation study is to compare approaches to one-mode homogeneity blockmodeling on binary networks. The approaches under comparison described in subsections in Section 2 are indirect blockmodeling based on the corrected squared Euclidian distance and Ward's hierarchical clustering, homogeneity sum of squares blockmodeling, and *k*-means blockmodeling. Given that what mainly differentiates *k*-means-based algorithms from homogeneity (sum of squares) generalized blockmodeling algorithms (only when used with structural equivalence and no pre-specification) is the optimization algorithm, the generalized blockmodeling in this section is typically called a relocation-based algorithm.

3.1. Design of the study

In this simulation study, random networks were generated using the following generating procedure. First, the partition of units into the desired number of clusters was randomly generated with the constraint that each cluster had to have at least three units. For each network, a random image matrix (a matrix representing ties among clusters) was generated based on three constraints:

1. Only two types of blocks were generated, named 'null' (very sparse) and 'complete' (less sparse);
2. The expected number of 'complete' blocks in any row (or column) of the image matrix was 1.5 (regardless of the number of clusters);
3. The image matrix had to be such that no two clusters had the same pattern of ties (were not structurally equivalent in the image matrix/network).

The final network was generated based on such an image matrix. The probability of a tie in 'null' blocks was set to 0.05, while the probability of a tie in 'complete' blocks was set to one of five possible values: 0.08, 0.12, 0.16, 0.20, 0.24. The 'null' block density was chosen based on some real networks, which are usually sparse. The 'complete' block densities were chosen so that the recovery of the correct partition for at least the best methods in most scenarios runs from terrible to good. Some examples of generated networks are presented in Figure 1.

The design of the simulation allowed all possible combinations of the factors listed below (their levels) to be tested. For each combination, 10 random networks were generated and analyzed. The factors that varied were (the names in square brackets are used in the figures and tables):

1. total number of units [n]: 200, 400, 800, 1600;
2. the number of clusters [k]: 2, 4, 8, 16;

3. the probability of a tie in 'complete' blocks [Prob. of a tie in com. blocks]: 0.08, 0.12, 0.16, 0.20, 0.24.

Since all combinations of factors were tested and 10 random networks were generated for each combination, 800 ($= 4 \times 4 \times 5 \times 10$) networks were generated. These networks were then analyzed with the algorithms that all try to minimize the sum of squares criterion, that is, the sum of squared deviations from the block means.

The following methods were used (the names in square brackets are used in the figures and tables):

- indirect approach, described in Subsection 2.2.
  - Hierarchical clustering, Ward's method applied to corrected squared Euclidian distances [HclustWard2]
  - Hierarchical clustering, Ward's method applied to corrected Euclidian distances [HclustWard]
  - Hierarchical clustering, the Complete Linkage method applied to corrected Euclidian distances [HclustComp]
  - hierarchical clustering, the Single Linkage method applied to corrected Euclidian distances [HclustSingle]
- relocation-based algorithms – homogeneity sum of squares generalized blockmodeling according to structural equivalence as described in Subsection 2.3 and implemented in the `blockmodelingTest` package (Žiberna, 2019a). The following versions were used:
  - with only transfers allowed; the transfers are tried in random order and, as soon as an improvement is found, the algorithm moves to the improved partition [RL moves]
  - the same as above, except that in the both transfers and exchanges were tried [RL moves & ex.]
  - with only transfers allowed; all possible transfers are evaluated and the algorithm moves to the partition associated with the biggest improvement [RL moves all]
  - same as above, except that both all possible transfers and all possible exchanges are evaluated [RL moves & ex. all]
- the *k*-means-based algorithm described in Subsection 2.4, implemented in package `kmBlocks` (Žiberna, 2019b) for the R programming language:
  - Only *k*-means [k-means]
- the *k*-means-based followed by the relocation algorithm described in Subsection 2.5:
  - *k*-means followed by "RL move". 5/6 of the time was allocated to *k*-means, while the remaining $1/6$ was given to the "RL move" for optimizing the best partition found by "*k*-means" [k-means + RL move]
  - *k*-means followed by "RL move & ex." $5/6$ of the time was allocated to *k*-means, while the remaining 1/6 was given to "RL move & ex." for optimizing the best partition found by "*k*-means" [k-means + RL move & ex.].

In addition, the error of the original partition, namely, the partition used in generating the network, and the error for the partition where all units are in the same (one) cluster, were computed. The results for these partitions are not presented, but were only used to compute one of the performance measures: relative error (see the next subsection).

3.2. Implementation and evaluation measures

The simulation study was run on a desktop computer with an Intel® Core™ i7-7700 3.6GHz CPU with 16 GB of RAM. All algorithms were run through the R statistical software version

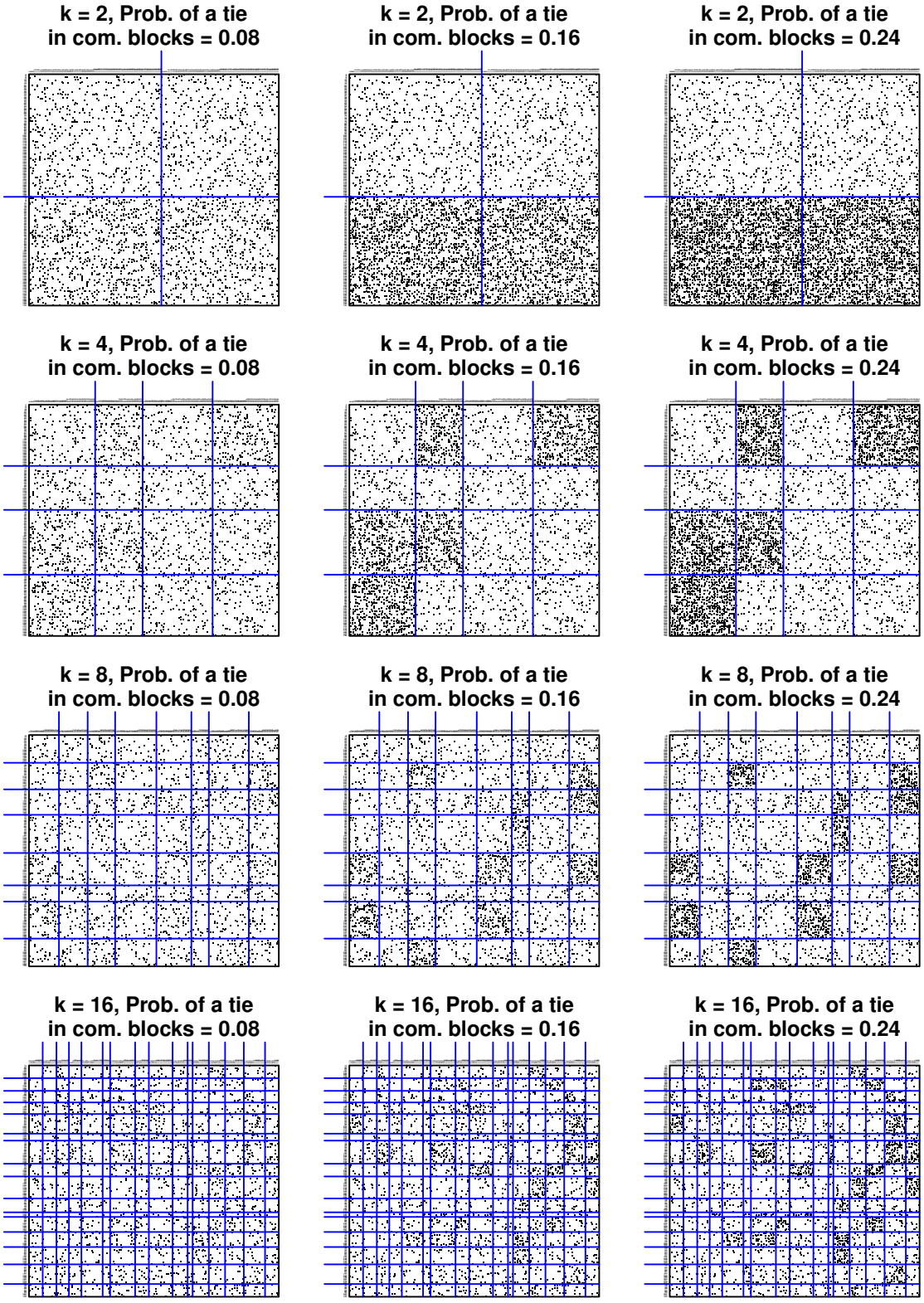

Figure 1: Examples of a generated network based on the original partition for networks with 200 units ($n = 200$), with all possible cluster sizes ($k = 2, 4, 8, 16$) and with the probability of 'complete' blocks [Prob. of a tie in com. blocks] equaling 0.08, 0.16, 0.24. The image matrix is the same for all densities with a given number of clusters.

3.5.3 (64-bit version).

The indirect approach was conducted by first computing the corrected Euclidian distance using the `sedist()` function from the `blockmodelingTest` R package (Žiberna, 2019a). On this, the hierarchical clustering was computed with the `hclust()` function (method `ward.D2`) from the `stats` package (R Core Team, 2019).[2]

For the relocation algorithm (homogeneity generalized blockmodeling), the implementation in the `blockmodelingTest` R package version 0.3.5.9000 was used (Žiberna, 2019a). The most time-consuming parts of the algorithm (refinement of the partitions) was programmed in C. For the *k*-means algorithm, the `kmBlocks` package was used (Žiberna, 2019b). The same part (refinement of partitions) was programmed in C++ using the `Rcpp` (Eddelbuettel & Francois, 2011) and `RcppArmadillo` (Eddelbuettel & Sanderson, 2014). The implementation of the other parts (generating random partitions, choosing the best result, etc., was implanted in R and practically the same for both approaches to make the results of these two approaches as similar as possible. It should be mentioned that the focus was more on the comparability of these two implementations (relocation algorithms and *k*-means) than on the efficiency of the two algorithms.

The analysis of all networks with all algorithms for a single combination of factors was run on a single thread, although multiple threads were used to run simulations for several factor combinations simultaneously.

Each iterative method was allocated at most 3 minutes for analyzing each network. The iterative methods (*k*-means and generalized blockmodeling) methods were implemented so that they finished after the time limit had been reached. Accordingly, if the time limit was reached, the refinement stopped and the current partition (and error) were saved together with those with already finished restarts (if any).[3] The methods "k-means + RL move" and "k-means + RL move & ex." often finished (substantially) sooner because the second phase (relocation algorithm) finished much faster than allowed.

This excludes the indirect approach, which is not an iterative one and where the execution time was not limited. The results show this approach needed much less time on networks with 800 units or less, although it also needed about 5 minutes on networks with 1600 units. However, since this method is not the main focus of the paper, this is not regarded as problematic.

The results of the methods were assessed based on two main measures: error (sum of squared deviations from the block means) and the Adjusted Rand Index (Hubert & Arabie, 1985). Especially in graphs (to make scales more comparable), relative error is used instead of original error. Some other measures based on these two (for ease of interpretation) and some auxiliary statistics were also computed or recorded. The following statistics were therefore saved (the names in square brackets indicate the names used in supplementary materials):

- Adjusted Rand Index [ARI]

[2]The `ward.D` method was also tested. `ward.D2` internally squares the distances and is therefore equivalent to using squared Euclidian distances, while `ward.D` uses original Euclidian distances. The results were very similar. The single linkage and complete linkage methods were also tested but, as expected, their results were inferior and are therefore not presented.

[3]As the check for the time limit is only performed at certain points in the code, it could happen that the total time was exceeded. In this case, the results of the last restart were used with the probability proportional to the time left for execution before the last restart relative to the time used by the last restart. This was implemented so that methods with faster restarts would not be favored (for bigger problems). In rare cases where this resulted in no restarts being used, the worst-case scenario, that is, when we assume only a single cluster, was taken as the solution for this method.

- Is the method the best method for a given network based on the ARI? [Is best ARI]
- Error [Error] – sum of squared deviations from the block means
- Relative error [Relative error]. The relative error compares the error of each method with the error of the original partition (used to generate the network) and the error that we would obtain if we had assumed only one cluster and therefore the whole network would represent one block (all reasonable methods should do better than that). It represents how much the error of a given method is worse than that of the original partition, relative to how much the error of the original is better than that in the case of only one cluster. It is computed as:

$$\text{relative error} = \frac{\text{err} - \text{err}_{\text{org}}}{\text{err}_{\text{one}} - \text{err}_{\text{org}}}$$

  where err is the error of the evaluated method, err_{org} is the error of the original partition, and err_{one} is the error in the case of only one cluster. Lower values are of course better, although values below 0 might be interpreted as overfitting.
- Is the method the best method for a given network based on the error? [Is best error]
- Number of restarts achieved[4] [Number of restarts]
- Elapsed time [Elapsed time]; the time used by the algorithm.

4. Results

Due to the magnitude of the results, not all are presented in this paper, although all results may be found in the interactive table and chart in the supplementary materials. In the paper, only the results for two measures—relative error and the Adjusted Rand Index—are presented.

For better readability, solely the results for a selection of methods are presented. At least one method is presented for each group of methods (indirect approaches, generalized blockmodeling/relocation-based, *k*-means, combination). Among indirect approaches, only the results for Ward's hierarchical clustering based on corrected squared Euclidian distance are presented. Here the selection was mainly based on theory (see Subsection 2.2). The results of using Ward's method on "unsquared" corrected Euclidian distance were in some cases slightly better, yet the difference is negligible. Other versions of the indirect approach performed worse.

In addition, the versions of relocation algorithm where all possible transfers or transfers and exchanges are evaluated before selecting the best one ("RL move all" and "RL move & ex. all") are omitted. As may be seen in the supplementary materials, these methods practically never performed better than the other versions of the relocation algorithm and usually (especially in the case of larger networks) performed much worse.

The results for mean relative error are presented in Figure 2, while the results for the mean Adjusted Rand Index (ARI) are shown in Figure 3. Based on the relative error (Figure 2), we may conclude that in most cases studied with only 2 groups the difference among the various approaches is minimal. For most other cases, hierarchical clustering is shown to perform worse than other methods. The difference is in most cases bigger when there is a lower probability of a tie in the 'complete' blocks. However, in the case of a large number of units (800 and especially 1600) and a relatively high probability of a tie in the 'complete'

[4]The last restart is taken into account (if not deleted as explained in the previous footnote) even if it ended prematurely.

blocks, the hierarchical approach performs better than the relocation algorithm,[5] while the *k*-means-based approaches are clearly superior.

As the number of units rises, the differences among approaches that include the *k*-means algorithm and those using only the relocation algorithm increase, where the algorithms which include *k*-means algorithm are the clear winners. The differences are slightly bigger as the probability of a tie in the 'complete' blocks increases. Only in cases with a relatively small number of units and large number of clusters do the approaches based on the relocation algorithm perform slightly better than those based on *k*-means. These are also cases where using the relocation algorithm after *k*-means is obviously beneficial. In most other cases, no clear suggestions regarding this can be given.

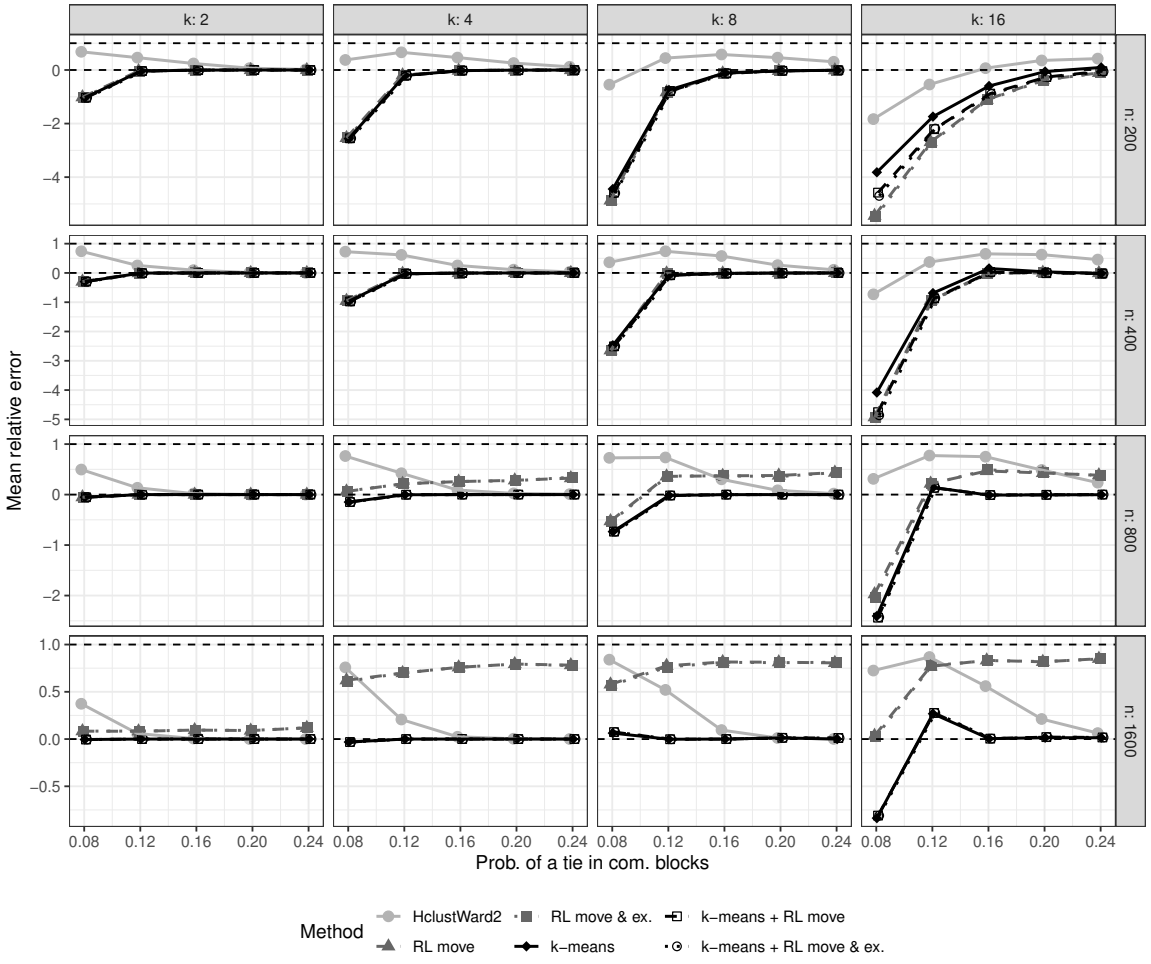

Figure 2: Values of the mean relative error for the different methods. Each line represents the results of a single method. The panels are determined by the values of the variables number of groups (k) and number of units (n) indicated (variables and their value) in the gray rectangles on the top and right sides of the figure.

In some cases, we can observe that the relative error is lower than 0, indicating that the algorithms found a partition with the error lower than the error of the partition that was used to generate the network. This is a case of overfitting and happens where there is not enough

[5]It should however be noted that on networks with 1600 units the hierarchical approach on average needed 5 minutes to complete, which is more time than was given for the other approaches.

data to recover the true structure. That is, this occurs in cases where the difference between the probability of a tie in 'null' and of that in 'complete' blocks is to low compared to the expected size of the blocks. As may also be observed in Figure 1, random tie generation causes that block densities are not exactly equal to the probability of a tie in a certain block.

Another interesting observation is that while for the relocation algorithms the relative error more often than not increases as the number of units increases, this is not the case with the *k*-means-based methods. Larger networks on one hand mean larger blocks which are, especially at the relative low densities used in this simulation, easier to find, while on the other hand they also increase the computational burden. In a setting with 800 units or more, *k*-means based algorithms clearly outperform the relocation algorithms.

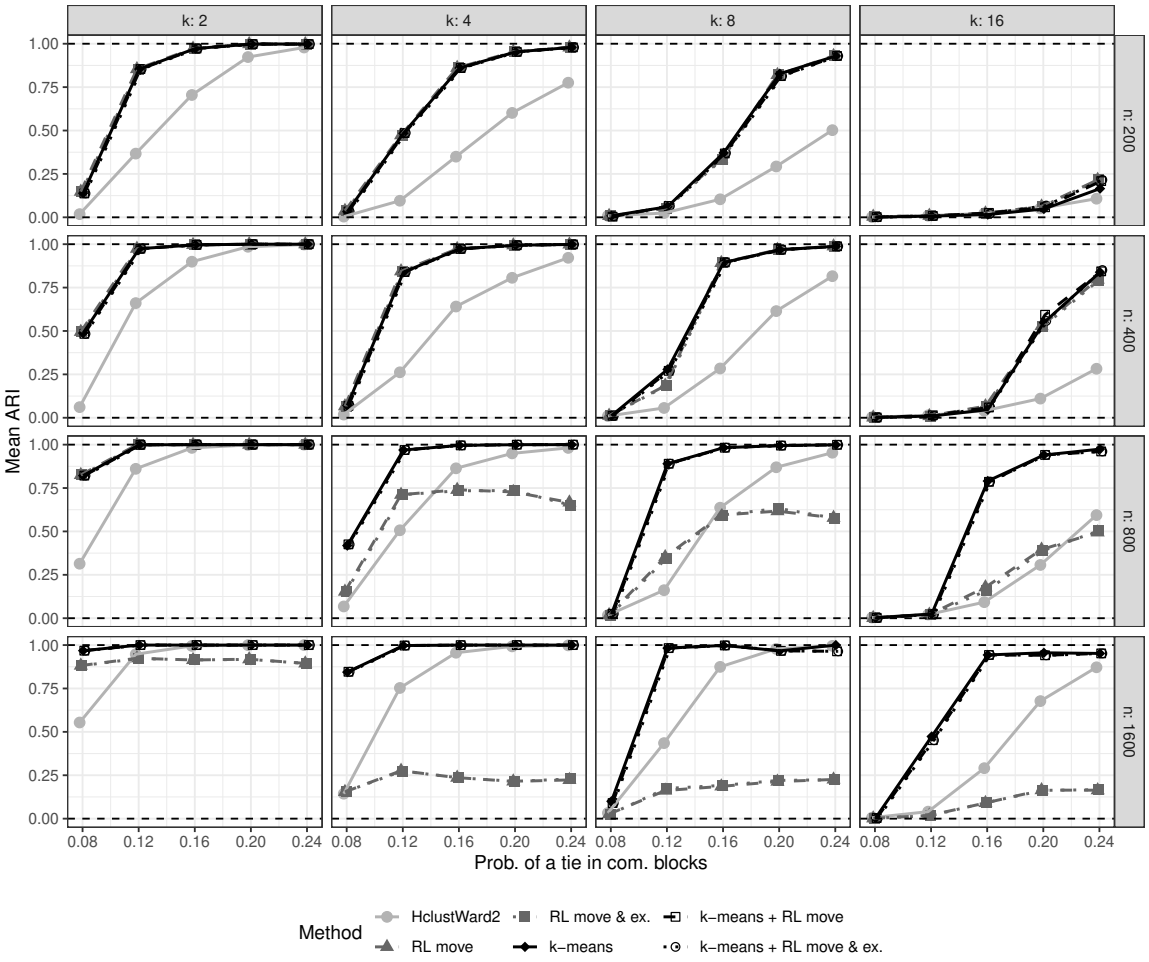

Figure 3: Values of the mean Adjusted Rand Index for the different methods. Each line represents the results of a single method. The panels are determined by the values of the variables number of groups (k) and number of units (n) indicated (variables and their value) in the gray rectangles on the top and right sides of the figure.

Results based on the mean Adjusted Rand Index (Figure 3) are even more favorable for the *k*-means versus relocation algorithm approaches. In almost all cases, one of the *k*-means based approaches is the best and the differences among them are negligible. Only in large k, small n cases (a large number of clusters and small number of units) are the approaches using the relocation algorithm slightly better, however ARI is relatively low here for all methods. As seen on the previous graph, indirect approach preforms well only in the case

of a large number of units (800 and especially 1600) and a relatively high probability of a tie in the 'complete' blocks. As expected, for all methods ARI rises as the probability of a tie in 'complete' blocks rises and when the number of clusters decreases (due to larger blocks). The effect of networks' size on ARI depends on the methods used. In general, it has a positive effect on *k*-means based approaches and the indirect approach, and generally a negative effect on the relocation-based approaches (with some exceptions). As discussed above with respect to relative error, larger networks mean larger blocks which are, especially at the relative low densities used in this simulation, easier to find, yet they also increase the computational burden. It seems that *k*-means based approaches are fast enough to use the advantages of larger networks (of the sizes analyzed here), that is, they are not hampered that much by the greater computational burden.

5. Conclusion

One can find several methods for one-mode blockmodeling of networks based on the minimum sum of squares criteria. This paper aimed to compare these methods using a simulation. The main aim was to compare approaches based on the relocation algorithm (homogeneity generalized blockmodeling (Žiberna, 2007)) and its variants with the *k*-means algorithm (Žiberna, 2020) and an approach where the relocation algorithm is used to optimize the partition produced by *k*-means. In addition, an indirect approach was considered.

The overall finding is that in most cases, the *k*-means based approaches, namely either *k*-means alone or *k*-means followed by the relocation algorithm, perform best both in terms of the optimized criterion and the similarity between the original partition (the one used for generating the network) and the one found by the algorithm. In addition, it is never much worse than the best approach and thus generally seems to be the safest approach. The difference is negligible for networks with 400 units or smaller, but becomes very big for large networks (1600 units), especially when there are 4 clusters or more (in case of large networks and a clear structure, the hierarchical approach can produce similar results as the *k*-means based approach). Only in the case of relatively small networks and a large number of clusters do pure relocation algorithms usually perform better. As expected, the use of indirect approaches is never the best approach, at least not in terms of the optimized criterion.

The results for whether it is better to use just *k*-means or *k*-means followed by relocation algorithms are inconclusive. The latter approach is clearly only superior for small networks with a relatively large number of clusters, where the use of the pure relocation algorithm is generally preferred.

With respect to relocation approaches, the use of versions of algorithm that move to a new partition as soon as an improvement is found is recommended because evaluating all possible transfers (or transfers and exchanges) is prohibitive for larger networks.

Of course, this study has its own limitations, mainly stemming from the fact that only binary networks of selected sizes (from 200 to 1600), numbers of clusters (2 to 16) and densities were evaluated. The study also assumed that the number of clusters is known. A number of unsettled questions remains that could be answered using simulation studies, for example, ways of determining the correct number of clusters and what is the best way for initializing the *k*-means algorithm, to name just a few.

References

Anderson, C. J., Wasserman, S., & Faust, K. (1992). Building stochastic blockmodels. *Social Networks*, *14*(1–2), 137–161. https://doi.org/10.1016/0378-8733(92)90017-2

Baier, D., Gaul, W., & Schader, M. (1997). Two-mode overlapping clustering with applications to simultaneous benefit segmentation and market structuring. In *Classification and knowledge organization* (pp. 557–566). Springer. https://doi.org/10.1007/978-3-642-59051-1_58

Batagelj, V., Ferligoj, A., & Doreian, P. (1992). Direct and indirect methods for structural equivalence. *Social Networks*, *14*(1–2), 63–90. https://doi.org/10.1016/0378-8733(92)90014-X

Borgatti, S. P., & Everett, M. G. (1992). Regular blockmodels of multiway, multimode matrices. *Social Networks*, *14*(1–2), 91–120. https://doi.org/10.1016/0378-8733(92)90015-Y

Breiger, R. L., Boorman, S. A., & Arabie, P. (1975). An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *Journal of Mathematical Psychology*, *12*(3), 328–383. https://doi.org/10.1016/0022-2496(75)90028-0

Brusco, M. J., & Doreian, P. (2015a). An exact algorithm for the two-mode *KL*-means partitioning problem. *Journal of Classification*, *32*(3), 481–515. https://doi.org/10.1007/s00357-015-9185-z

Brusco, M. J., & Doreian, P. (2015b). A real-coded genetic algorithm for two-mode *KL*-means partitioning with application to homogeneity blockmodeling. *Social Networks*, *41*, 26–35. https://doi.org/10.1016/j.socnet.2014.11.007

Brusco, M. J., & Steinley, D. (2007). A variable neighborhood search method for generalized blockmodeling of two-mode binary matrices. *Journal of Mathematical Psychology*, *51*, 325–338. https://doi.org/10.1016/j.jmp.2007.07.001

Brusco, M. J., & Steinley, D. (2011). A tabu-search heuristic for deterministic two-mode blockmodeling of binary network matrices. *Psychometrika*, *76*(4), 612–633. https://doi.org/10.1007/s11336-011-9221-9

Burt, R. S. (1976). Positions in networks. *Social Forces*, *55*(1), 93–122. https://doi.org/10.2307/2577097

Doreian, P., Batagelj, V., & Ferligoj, A. (1994). Partitioning networks based on generalized concepts of equivalence. *The Journal of Mathematical Sociology*, *19*(1), 1–27. https://doi.org/10.1080/0022250X.1994.9990133

Doreian, P., Batagelj, V., & Ferligoj, A. (2004). Generalized blockmodeling of two-mode network data. *Social Networks*, *26*, 29–53. https://doi.org/10.1016/j.socnet.2004.01.002

Doreian, P., Batagelj, V., & Ferligoj, A. (2005). *Generalized blockmodeling*. Cambridge University Press.

Doreian, P., & Mrvar, A. (2009). Partitioning signed social networks. *Social Networks*, *31*(1), 1–11. https://doi.org/10.1016/j.socnet.2008.08.001

Eddelbuettel, D., & Francois, R. (2011). Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, *40*(8), 1–18. https://doi.org/10.18637/jss.v040.i08

Eddelbuettel, D., & Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics & Data Analysis*, *71*, 1054–1063. https://doi.org/10.1016/j.csda.2013.02.005

Hartigan, J. A. (1975). *Clustering algorithms*. Wiley.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A *K*-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100–108. https://doi.org/10.2307/2346830

Holland, P. W., Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social Networks*, *5*(2), 109–137. https://doi.org/10.1016/0378-8733(83)90021-7

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*(1), 193–218. https://doi.org/10.1007/BF01908075

Lorrain, F., & White, H. C. (1971). Structural equivalence of individuals in social networks. *The Journal of Mathematical Sociology*, *1*(1), 49–80. https://doi.org/10.1080/0022250X.1971.9989788

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. Lecam & J. Neyman (Eds.), *Proceedings of the Fifth Berkeley symposium on mathematical statistics and probability* (pp. 281–297).

R Core Team. (2019). *R: A language and environment for statistical computing* (Version 3.5.3) [Computer software]. R Foundation for Statistical Computing. https://www.r-project.org/

Snijders, T. A. B., & Nowicki, K. (1997). Estimation and prediction for stochastic blockmodels for graphs with latent nlock structure. *Journal of Classification*, *14*, 75–100.

Steinley, D. (2003). Local optima in K-means clustering: What you don't know may hurt you. *Psychological Methods*, *8*(3), 294–304. https://doi.org/10.1037/1082-989X.8.3.294

Steinley, D. (2006). K-means clustering: A half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, *59*(1), 1–34. https://doi.org/10.1348/000711005X48266

Van Mechelen, I., Bock, H.-H., & De Boeck, P. (2004). Two-mode clustering methods: A structured overview. *Statistical Methods in Medical Research*, *13*(5), 363–394.

van Rosmalen, J., Groenen, P. J. F., Trejos, J., & Castillo, W. (2009). Optimization strategies for two-mode partitioning. *Journal of Classification*, *26*, 155–181. https://doi.org/10.1007/s00357-009-9031-2

Vichi, M. (2001). Double *k*-means clustering for simultaneous classification of objects and variables. In S. Borra, R. Rocci, M. Vichi, & M. Schader (Eds.), *Advances in classification and data analysis* (pp. 43–52). Springer. https://doi.org/10.1007/978-3-642-59471-7_6

Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*(301), 236–244. https://doi.org/10.2307/2282967

White, D. R., & Reitz, K. P. (1983). Graph and semigroup homomorphisms on networks of relations. *Social Networks*, *5*(2), 193–234. https://doi.org/10.1016/0378-8733(83)90025-4

Žiberna, A. (2007). Generalized blockmodeling of valued networks. *Social Networks*, *29*, 105–126. https://doi.org/10.1016/j.socnet.2006.04.002

Žiberna, A. (2019a). *blockmodelingTest: An R package for generalized and classical blockmodeling of valued networks* (Version 0.3.5.9000) [Computer software]. R-Forge. https://rdrr.io/rforge/blockmodelingTest/

Žiberna, A. (2019b). *kmBlock: k-means like blockmodeling of one-mode and linked networks* (Version 0.0.1.9102) [Computer software]. R-Forge. https://rdrr.io/rforge/kmBlock/

Žiberna, A. (2020). *k*-means-based algorithm for blockmodeling linked networks. *Social Networks*, *61*, 153–169. https://doi.org/10.1016/j.socnet.2019.10.006