10 Blockmodels

Marc Nunkesser and Daniel Sawitzki

In the previous chapter we investigated different types of vertex equivalences which lead us to the notion of a *position* in a social network. We saw algorithms that compute the sets of equivalent actors according to different notions of equivalence. However, which of these notions are best suited for the analysis of concrete real world data seems to depend strongly on the application area.

Practical research in sociology and psychology has taken another way: Instead of applying one of the equivalences of the previous chapter, researchers often use heuristical role assignment algorithms that compute approximations of strong structural equivalence. More recently, statistical estimation methods for stochastic models of network generation have been proposed.

Typically, researchers collect some relational data on a group of persons (the *actor set*) and want to know if the latter can be partitioned into positions with the same or at least similar relational patterns. The corresponding area of network analysis is called *blockmodeling*. Relational data is typically considered as a directed loopless graph G consisting of a node set $V = \{v_1, \ldots, v_n\}$ and R edge sets $E_1, \ldots, E_R \subseteq V^2 \setminus \{(v, v) \mid v \in V\}$. The following definition of a blockmodel sums up most of the views that can be found in the literature.

Definition 10.0.1. *A* blockmodel $BM = (\mathcal{P}, B_1, \ldots, B_R)$ *of* G *consists of two parts:*

1. *A partition* $\mathcal{P} = (P_1, \ldots, P_L)$ *of* V *into* L *disjoint subsets called the* positions *of* G*. For* $v \in V$*, the position number* k *with* $v \in P_k$ *is denoted by* $P(v)$*.*
2. *Matrices* $B_r = (b_{k,\ell,r})_{1 \le k,\ell \le L} \in \{0,1\}^{L \times L}$*,* $1 \le r \le R$*, called* image matrices *that represent hypotheses on the relations between the positions with respect to each relation.*

Thus, a blockmodel is a simplified version of G whose basic elements are the positions. If we demand that nodes of the same position have exactly the same adjacencies, the equivalence classes of the structural equivalence relation introduced in Definition 9.1.3 (denoted by $\simeq \in V^2$ in this chapter) give us a unique solution $\mathcal{P}_\simeq$ to our partitioning problem.

Because the field of blockmodeling is concerned with processing real world data possibly collected in experiments, it is assumed that there is some 'true' blockmodel underlying the observed graph which may not be reflected cleanly by G. This may be caused by measurement errors or natural random effects. $\mathcal{P}_\simeq$

U. Brandes and T. Erlebach (Eds.): Network Analysis, LNCS 3418, pp. 253–292, 2005.

does not catch these deviations, and is therefore expected to contain too many positions hiding the true blockmodel.

Hence, blockmodeling is much about building relaxations of structural equivalence which are able to tolerate random distortions in the data up to an appropriate degree. Corresponding blockmodels are expected to have a minimal number of positions while tolerating only small deviations from the assumption of structural equivalence. Historically, the first methods used in blockmodeling have been heuristic algorithms which were believed to give good trade-offs between these two criterions.

In blockmodeling, graphs are often viewed from an adjacency matrix point of view. Let $A_r = (a_{i,j,r})_{i,j}$ denote the adjacency matrix of E_r, i. e., $a_{i,j,r} = 1 \Leftrightarrow (v_i, v_j) \in E_r$. Then, a blockmodel is represented by a permuted version of A which contains nodes of the same position in consecutive rows and columns.

Definition 10.0.2. *The* $\mathcal{P}$-permuted adjacency matrix $A_r^* := \left(a^*_{i,j,r}\right)_{i,j} := \left(a_{\pi^{-1}(i),\pi^{-1}(j),r}\right)_{i,j}$ *is obtained by reordering rows and columns of* A_r *with respect to the permutation* $\pi \in \Sigma_n$ *defined by*

$$\pi(i) < \pi(j) :\Leftrightarrow \left[P(v_i) < P(v_j)\right] \vee \left[\left(P(v_i) = P(v_j)\right) \wedge (i < j)\right]$$

for all $1 \leq i < j \leq n$.

For $1 \leq k, \ell \leq L$, *the* $|P_k| \times |P_\ell|$*-submatrix*

$$A_r^{k,\ell} := (a_{\pi^{-1}(i),\pi^{-1}(j),r})_{(v_i,v_j)\in P_k \times P_\ell}$$

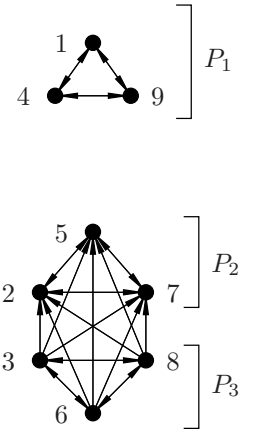
is called a block *and contains the connections between positions* k *and* ℓ *with respect to relation* r.

That is, the rows and columns of A_r^* are lexicographically ordered with respect to position and index. Nodes of the same position have consecutive rows and columns. A block $A_r^{k,\ell}$ represents the part of A_r^* that corresponds to the relation between P_k and P_ℓ. The entry $b_{k,\ell,r}$ of B_r should contain this information in distilled form. For $\mathcal{P}_{\simeq}$, each block is solely filled with ones or zeros (except the diagonal elements), and it makes sense to set $b_{P(v_i),P(v_j),r} := a_{i,j,r}$.

In blockmodels obtained by heuristic algorithms, the nodes of a position P_k do not necessarily have equal neighborhoods. Nevertheless, the adjacencies of nodes of the same position should be very similar in a good model, and $b_{k,\ell,r}$ should express trends existing in $A_r^{k,\ell}$. Therefore, a variety of methods has been employed to derive the image matrices from the blocks according to $\mathcal{P}$. If we consider B_r as an adjacency matrix of a graph having the positions as nodes, we obtain the so called *reduced graph* of E_r (compare Definition 9.0.3).

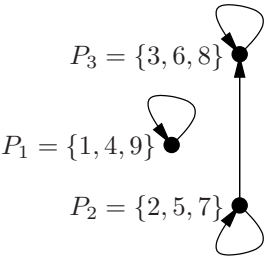
Figure 10.1(a) gives an example of a network G, its adjacency matrix A, and its three positions due to the structural equivalence relation $\simeq$. Figure 10.1(b) shows both the corresponding permuted adjacency matrix A^* and the reduced graph with positions as nodes.

E	1	2	3	4	5	6	7	8	9
1	-	0	0	1	0	0	0	0	1
2	0	-	0	0	1	0	1	0	0
3	0	1	-	0	1	1	1	1	0
4	1	0	0	-	0	0	0	0	1
5	0	1	0	0	-	0	1	0	0
6	0	1	1	0	1	-	1	1	0
7	0	1	0	0	1	0	-	0	0
8	0	1	1	0	1	1	1	-	0
9	1	0	0	1	0	0	0	0	-



(a) G's adjacency matrix A with corresponding graph and positions P_1, P_2, and P_3 due to the structural equivalence relation.

E	3	6	8	1	4	9	2	5	7
3	-	1	1	0	0	0	1	1	1
6	1	-	1	0	0	0	1	1	1
8	1	1	-	0	0	0	1	1	1
1	0	0	0	-	1	1	0	0	0
4	0	0	0	1	-	1	0	0	0
9	0	0	0	1	1	-	0	0	0
2	0	0	0	0	0	0	-	1	1
5	0	0	0	0	0	0	1	-	1
7	0	0	0	0	0	0	1	1	-



(b) $\mathcal{P}$-permuted adjacency matrix A^* and the corresponding reduced graph. Blocks in A^* are separated by lines. Due to the structural equivalence, they contain solely ones or zeros in non-diagonal positions.

Fig. 10.1. Example network $G = (V, E)$ and its blockmodel due to the structural equivalence relation

Contents. This chapter gives a survey on selected blockmodeling approaches which are either well-established and have been widely used, or which seem to be promising and to give novel perspectives on this quite old field in network analysis. We will restrict ourselves to graphs $G = (V, E)$ containing only one edge set corresponding to one actor relation. Most approaches can be easily adapted to the case of several (sometimes weighted) relations.

Section 10.1 presents blockmodeling approaches that are mainly based on heuristic assumptions on positional interplay without a concrete model of network generation. In contrast to these so called deterministic models, which in-

clude some of the oldest algorithms used in blockmodeling, Section 10.2 presents approaches based on stochastic models. They assume that the positional structure influences a randomized process of network generation and try to estimate the parameters of the corresponding probability distribution. In this way, we can both generate and evaluate hypotheses on the network positions. Conclusions on both kinds of methods and an overview of the relevant literature are given in Section 10.3

10.1 Deterministic Models

In this section, well-established blockmodeling approaches are presented which are mainly based on heuristic assumptions on positional interplay without a concrete stochastic model of the process that generates the network. Instead, certain relaxations of the structural equivalence relation are used to decide whether two nodes share the same position. Because the decision criterions are based upon static network properties, we call these approaches deterministic models.

In order to weaken the structural equivalence, we need to measure to what extend two nodes are equivalent. Therefore, Section 10.1.1 is devoted to two of the most popular measures. These need not to be metrics, but the techniques for multidimensional scaling discussed in Section 10.1.2 can be used to embed actors in a low-dimensional Euclidian space. Having pair-wise distance values for the actors, clustering based methods like Burt's algorithm (see Section 10.1.3) are popular ways to finally partition the actor set V into positions $\mathcal{P}$. Section 10.1.4 presents the CONCOR algorithm that is an alternative traditional method to obtain $\mathcal{P}$.

The methods up to this point have been mainly introduced in the 70's and represent classical approaches. They are only used to compute a partition $\mathcal{P}$ of the actor set; the image matrix B is typically obtained by applying some standard criterions to $\mathcal{P}$ discussed in Section 10.1.5. In Section10.1.6 we discuss different goodness-of-fit indices that are obtained by comparing the $\mathcal{P}$-permuted adjacency matrix A^* with the image matrix B of a concrete blockmodel. Finally, Section 10.1.7 introduces a generalized blockmodeling framework which integrates the steps of partitioning the actor set, computing B, and evaluating the resulting blockmodel. It represents the most recent blockmodeling approach in this section on deterministic models.

10.1.1 Measuring Structural Equivalence

We have already noted in the introduction that relations between actors in observed real-world networks may reflect an eventual underlying positional structure only in a distorted and inexact way. Therefore, blockmodeling algorithms have to tolerate a certain deviation from perfect structural equivalence, whose idea of equal neighborhoods seems to be reasonable in principle. Hence, it does not suffice to know if two nodes v_i and v_j are equivalent w. r. t. $\simeq$—we also want

to know some value $\delta_{i,j}$ describing how close a node pair (v_i, v_j) is to equivalence. In the following, $\delta_{i,j}$ will always denote a symmetric distance measure between the adjacency relations of node v_i and v_j with the properties $\delta_{i,i} = 0$ and $\delta_{i,j} = \delta_{j,i}$. Superscripts identify special measures.

In order to apply geometrical distance measures, we consider the concatenation of the ith row and ith column of A as a point in the $2n$-dimensional space $\mathbb{R}^{2n}$. Burt [107] was the first who proposed to use the *Euclidian distance* in blockmodeling:

Definition 10.1.1. *The* Euclidian distance $\delta^{\mathrm{e}}_{i,j}$ *between actors* v_i *and* v_j *is defined by*

$$\delta^{\mathrm{e}}_{i,j} := \sqrt{\sum_{k \neq i,j} (a_{i,k} - a_{j,k})^2 + \sum_{k \neq i,j} (a_{k,i} - a_{k,j})^2} \tag{10.1}$$

for $1 \leq k \leq n$.

Note that $\delta^{\mathrm{e}}_{i,i} = 0$, $\delta^{\mathrm{e}}_{i,j} = \delta^{\mathrm{e}}_{j,i}$, and $0 \leq \delta^{\mathrm{e}}_{i,j} \leq \sqrt{2(n-2)}$.

A second widely used measure of structural equivalence is the *correlation coefficient*, also known as *product-moment coefficient*. In contrast to the Euclidian distance, it does not directly compare entries in A, but their deviations from mean values of rows and columns.

Definition 10.1.2. *Let* $\bar{a}_{i,\cdot} := \sum_{1 \leq k \leq n} a_{i,k}/(n-1)$ *resp.* $\bar{a}_{\cdot,i} := \sum_{1 \leq k \leq n} a_{k,i}/(n-1)$ *be the mean of the values of the* i*th row resp.* i*th column of* A. *The* correlation coefficient *(or* product-moment coefficient*)* $c_{i,j}$ *is defined by*

$$\frac{\sum_{k \neq i,j} (a_{i,k} - \bar{a}_{i,\cdot})(a_{j,k} - \bar{a}_{j,\cdot}) + \sum_{k \neq i,j} (a_{k,i} - \bar{a}_{\cdot,i})(a_{k,j} - \bar{a}_{\cdot,j})}{\sqrt{\sum_{k \neq i,j} [(a_{i,k} - \bar{a}_{i,\cdot})^2 + (a_{k,i} - \bar{a}_{\cdot,i})^2]} \sqrt{\sum_{k \neq i,j} [(a_{j,k} - \bar{a}_{j,\cdot})^2 + (a_{k,j} - \bar{a}_{\cdot,j})^2]}} \tag{10.2}$$

for $1 \leq k \leq n$. *The matrix* $C = (c_{i,j})_{i,j}$ *is called the* correlation matrix *of* A.

That is, its numerator is the sum of products of v_i's and v_j's deviations from their respective row and column mean values. In the denominator, these deviations are squared and summed separately for v_i and v_j before their respective square roots are taken and the results are multiplied.

Note that $c_{i,j} \in [-1, 1]$. In statistics, the correlation coefficient is used to measure to what degree two variables are linearly related; an absolute correlation value of 1 indicates perfect linear relation, while a value of 0 indicates no linear relation. Especially, $c_{i,i} = 1$ and $c_{i,j} = c_{j,i}$. On the other hand, $|c_{i,j}| = 1$ does not imply $v_i \simeq v_j$, and $c_{i,j} = 0$ does not mean that the ith and jth row/column of A are not related at all—they are just not linearly related.

In order to derive a measure that fulfills the property $\delta_{i,i} = 0$, we normalize $c_{i,j}$ to $\delta^{\mathrm{c}}_{i,j} := 1 - |c_{i,j}|$.

Comparison of Euclidian Distance and Correlation Coefficient. Let us compare the two measures $\delta^{\mathrm{e}}_{i,j}$ and $\delta^{\mathrm{c}}_{i,j}$. We have already seen that the Euclidian distance $\delta^{\mathrm{e}}_{i,j}$ is directly influenced by the difference between the entries for v_i and v_j in A, while the normalized correlation coefficient $\delta^{\mathrm{c}}_{i,j}$ also incorporates the mean values $a_{i,\cdot}$, $a_{\cdot,i}$, $a_{j,\cdot}$, and $a_{\cdot,j}$. Thus, $\delta^{\mathrm{e}}_{i,j}$ measures the absolute similarity between the neighborhoods of v_i and v_j, while $\delta^{\mathrm{c}}_{i,j}$ measures the similarity of the mean deviations.

In order to make the formal relationship between $\delta^{\mathrm{e}}_{i,j}$ and $c_{i,j}$ better understandable, we temporarily assume that both (10.1) and (10.2) contain only the row-related sums.

Property 10.1.3. Let $\sigma_{i,\cdot} := \sqrt{\sum_{k\neq i}\left(a_{i,k}-\bar{a}_{i,\cdot}\right)^2/(n-1)}$ resp. $\sigma_{\cdot,i} := \sqrt{\sum_{k\neq i}\left(a_{k,i}-\bar{a}_{\cdot,i}\right)^2/(n-1)}$, $1 \leq k \leq n$, be the standard deviation of the ith row resp. ith column of A. Then, it holds

$$\left(\delta^{\mathrm{e}}_{i,j}\right)^2 = (n-2)\left[\left(\bar{a}_{i,\cdot}-\bar{a}_{\cdot,i}\right)^2+\sigma^2_{i,\cdot}+\sigma^2_{j,\cdot}-2c_{i,j}\sigma_{i,\cdot}\sigma_{j,\cdot}\right] \quad .$$

That is, the Euclidian distance grows with increasing mean difference $|\bar{a}_{i,\cdot}-\bar{a}_{j,\cdot}|$ and variance difference $|\sigma^2_{i,\cdot}-\sigma^2_{j,\cdot}|$, while these are filtered out by $c_{i,j}$. If the used blockmodeling method leaves freedom in choosing a measure, structural knowledge about G should influence the decision: If the general tendency of an actor to be related to others is assumed to be independent of his position, the use of $\delta^{\mathrm{c}}_{i,j}$ is expected to give a better insight in the positional structure than $\delta^{\mathrm{e}}_{i,j}$. For example, if the relational data was obtained from response rating scales, some actors may tend to give consistently higher ratings than others.

In the following sections, we will see how such symmetric measures $\delta_{i,j}$ are used in computing the actor set partition $\mathcal{P}$.

10.1.2 Multidimensional Scaling

Blockmodels and MDS. In the previous section we saw that the deterministic blockmodeling problem is connected to (dis-)similarity measures between the rows and columns of the adjacency matrix A that correspond to the actors. After we have decided upon a particular dissimilarity measure, we get for the set of actors a set of pairwise dissimilarities, from which we might want to deduce the positions and the image matrix of the blockmodel. This in turn can be considered as a reduced graph, which we already saw in the introduction. This process can be seen as one of information reduction from the initial dissimilarities to an abstract representation. Clearly, this is not the only way to represent the blockmodel. In this section, we will discuss in detail a slightly different approach, where the abstract representation maps the actors to points in the plane. The distances between the points should roughly correspond to the dissimilarities between the actors. Points that are close to each other with respect to the other points could then again be interpreted as positions. The underlying general problem is called *multidimensional scaling* (MDS): Given a

set of dissimilarities of actors, find a 'good' representation as points in some space (two-dimensional Euclidean space for our purposes). It has been used as an intermediate step for blockmodeling, where clustering algorithms are run on the points produced by the MDS algorithm (see Section 10.1.3), and it is also considered a result in itself that needs no further postprocessing. The result can then be seen as a concise visual representation of the positional structure of a social network. Let us define the problem formally:

Problem 10.1.4 (Multidimensional Scaling Problem (MDS)). Given n objects by their $n \times n$ dissimilarity matrix δ, a dimension d and a loss function $\ell\colon \mathbb{R}^{n\times n} \times \{S \subset \mathbb{R}^d | |S| = n\} \to \mathbb{R}^+$, construct a transformation $f\colon \{1,\ldots,n\} \to \mathbb{R}^d$ such that the loss $\ell(\delta, P)$ is minimal, for $P = f(\{1,\ldots,n\})$.

The loss function $\ell(\delta, P)$ measures how much the dissimilarity matrix of the objects $\{1,\ldots,n\}$ is distorted by representing them as a point set P in d dimensional Euclidean space. Obviously, different loss functions lead to different MDS problems. In this whole section, we set $d = 2$ for ease of presentation even if the first approach to be presented can be easily extended to higher dimensions. When discussing solutions to multidimensional scaling problems we will often directly talk about the point set $P = \{(p_x^1, p_y^1), \ldots, (p_x^n, p_y^n)\}$ that implicitly defines a possible transformation f. Then we also write $\delta[p, q]$ for the dissimilarity $\delta[f^{-1}(p), f^{-1}(q)]$ of the preimages of p and q for some points $p = p^i = (p_x^i, p_y^i)$ and $q = p^j = (p_x^j, p_y^j)$. We call any candidate set of points P for a solution a *configuration* and write $P = f(\delta)$ abusing notation slightly. In the next two sections we have selected out of the multitude of different MDS-approaches two algorithms that are particularly interesting: Kruskal's MDS algorithm and a recent algorithm with quality guarantee by Bădoiu. Kruskal's algorithm is probably the one that has been used most frequently in the blockmodeling context because it has become relatively established. On the other hand we are not aware of any study in blockmodeling in which Bădoiu's algorithm has been used. We present it here, because it is an algorithmically interesting method and has appealing properties like the quality guaranty.

Kruskal's MDS Algorithm. Historically, Kruskal's algorithm was among the first that gave a sound mathematical foundation for multidimensional scaling. Kruskal called his approach *nonmetric multidimensional scaling* to set it apart from earlier approaches that fall into the class of *metric scaling*. The latter approach tries to transform the dissimilarity matrix into distances by some class of parametric functions and then finds the parameters that minimize the loss function. This scenario is very similar to the classical estimation task and can be solved by least squares methods. In contrast to this parametric approach nonmetric multidimensional scaling makes no parametric assumptions about the class of legal transformations; the only condition that the transformation f should fulfill best-possible is the *monotonicity constraint* (MON)

$$\delta[p, q] < \delta[r, s] \Rightarrow \|p - q\|_2 \leq \|r - s\|_2 \tag{10.3}$$

for all $p, q, r, s \in P$. This constraint expresses that if a pair of objects is more similar than another pair then the corresponding pair of points must have a smaller (or equal) distance than the distance of the other pair of points. In (10.3), the only necessary information about the dissimilarities is their relative order.

The Stress. The key to Kruskal's algorithm is the right choice of a loss function that he calls *stress*. It is best introduced via *scatter diagrams*. Given a dissimilarity matrix and a candidate configuration of points P, we can plot the distances $d_{ij} = \|p^i - p^j\|_2$ versus the dissimilarities δ in a scatter diagram like in Figure 10.2(a).

Obviously, the configuration in Figure 10.2(a) does not fulfill the monotonicity constraint, because when we trace the points in the order of increasing dissimilarity, we sometimes move from larger to smaller distances, i.e. we move left. Let us call the resulting curve the *trace* of the configuration. A trace of a configuration that fulfills MON must be a *monotone curve* like in Figure 10.2(b). The idea is now to take the minimum deviation of the trace of a configuration from a monotone curve as the loss function. Clearly, if the trace itself is monotone this deviation is zero. More precisely, we define the *raw stress* of a configuration as

$$\min \left\{ \sum_{i<j} (d_{ij} - \hat{d}_{ij})^2 \Big| (\hat{d}_{ij})_{ij} \text{ fulfill MON} \right\} .$$

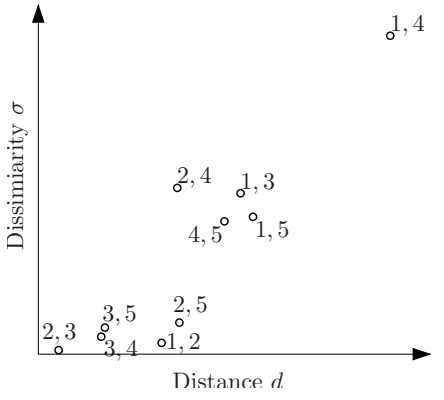
This means that the error is measured only at y-coordinates of points in the scatter diagram. At these y-coordinates, we search for points $\hat{d}_{ij}$ that together fulfill MON and minimize the squared error of distances to the corresponding points of the configuration. The raw stress has some disadvantages, for example it is not invariant under uniform stretching or shrinking of the dissimilarities. Therefore, stress is defined as follows.

Definition 10.1.5. *Given a dissimilarity matrix δ and a configuration of points P, the* stress *of P is defined by*

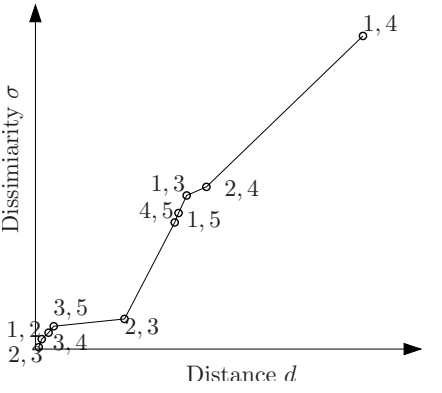
$$S(P) = \min \left\{ \frac{\sum_{i<j} (d_{ij} - \hat{d}_{ij})^2}{\sum_{i<j} d_{ij}^2} \Big| (\hat{d}_{ij})_{ij} \text{ fulfill MON} \right\} . \tag{10.4}$$

Note that the values of δ do not enter in (10.4); however, their order occurs implicitly via MON. In Figure 10.2(c) there is an example of a configuration together with a monotone curve that minimizes the stress, in Figure 10.2(d) the corresponding values of $\hat{d}_{ij}$ are shown as squares.

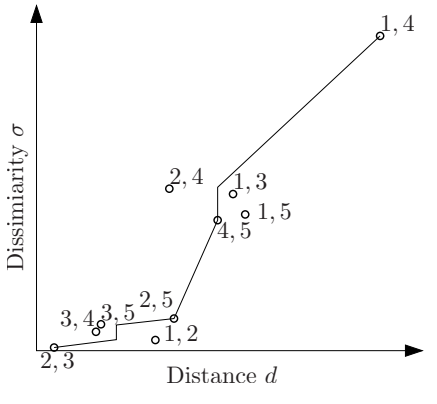
The Algorithm. To complete the description of the algorithm we need to know two further details: How is the stress computed and how is a configuration with minimum stress found? Assume we have answered the first question such that we have a procedure to compute the stress of any given configuration. For a



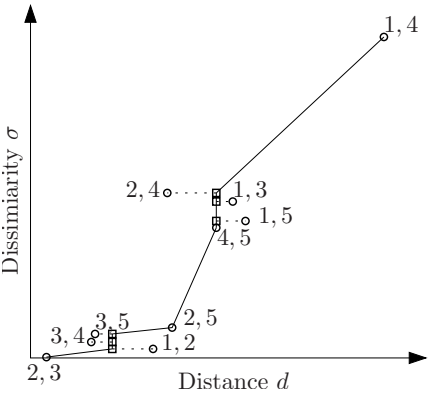
(a) A scatter diagram.



(b) A configuration that satisfies MON yields a monotone curve.



(c) The stress of a configuration is measured with respect to a monotone curve.



(d) The stress is defined by the $\hat{d}_{ij}$ values that correspond to the squares (and circles if they coincide with d_{ij}).

Fig. 10.2. Elements of Kruskal's MDS-Algorithm

given configuration it returns the correct values $\hat{d}_{ij}$. These values correspond to a *local stress function*

$$S_\ell(P) = S_\ell((p_x^1, p_y^1), \ldots, (p_x^n, p_y^n)) = \sum_{i<j} (d_{ij} - \hat{d}_{ij})^2 \Big/ \sum_{i<j} d_{ij}^2 \ , \qquad (10.5)$$

where we have still $d_{ij} = \|p^i - p^j\|_2$. The problem of finding a configuration with minimum local stress turns out to be the numerical problem of minimizing a func-

tion of $2n$ variables with respect to a given objective function, the stress. Therefore, any standard method for function minimization can be used. Kruskal proposes the *method of steepest descent* that starts with an arbitrary point in search space, computes the gradient of the local stress $(\partial S_\ell/\partial p_x^1, \partial S_\ell/\partial p_y^1, \ldots, \partial S_\ell/\partial p_y^n)$, and moves towards the negative direction of the gradient. Then, it recomputes the local stress in the new configuration and iterates until a local minimum is found. This need not be the global minimum. In this sense the algorithm is a heuristic without performance guarantee (just as any other general algorithm for minimization of non-convex functions). To understand that the algorithm is really as straight-forward as it sounds, observe that it is indeed possible to calculate the partial derivatives of a local stress function. In general, also other methods for function minimization could be used.

As for the computation of the $\hat{d}_{ij}$ we will briefly sketch the algorithm. It relies on the following observation.

Observation 10.1.6. *The $\hat{d}_{ij}$ that minimize the stress for a given configuration have the following form: The ordered list of dissimilarities can be partitioned into consecutive blocks $\{b_1, \ldots, b_k\}$ such that within each block, $\hat{d}_{ij}$ is constant and equals the average of the d_{ij} values in the block.*

Note that the $\hat{d}_{ij}$ values in Figure 10.2(d) have this form. From this observation it is clear that the problem can be solved by finding the correct partition. This is achieved by starting from the finest possible partition (each point in one block) and then iteratively joining an arbitrary pair of neighboring blocks for which the monotonicity constraint is violated.

MDS with Quality Guarantee. In this section we present a relatively new approach to multidimensional scaling by Bădoiu [105] that relies more on the combinatorial structure of the problem. As before the algorithm constructs an embedding of a given dissimilarity matrix into the plane. In this case the dissimilarity matrix is also called distance matrix, because Bădoiu's algorithm searches for a point set that not only qualitatively mirrors the order relation on the distances/dissimilarities, but also its objective is that the distances in the embedding should approximate the distances given by the matrix δ as precisely as possible. It is an approximation algorithm in the sense that the loss of the constructed embedding is bounded by $c\varepsilon$ if an optimal embedding has loss ε. Note that this is not the same as having a constant loss with respect to the original dissimilarity measure which is impossible in general. This algorithm is a quite recent result and was the first to give such guarantees. Its success stems from a clever choice of the loss function combined with beautiful insights into the combinatorial nature of the problem. Unfortunately, it is slightly too complicated to be presented here in its entirety. However, we will see the important parts and explain the ideas for the missing parts. All missing proofs can be found in [105].

The Loss Function. The loss function that is employed here is one that uses the L_∞-norm to measure the distance in $\mathbb{R}^2$. Remember that the infinity norm of a

vector its component with maximum absolute value. The loss is the maximum deviation of embedded distances from original distances.

$$\ell(\delta, f(\delta)) = \max_{1 \le i < j \le n} \left\{ \left| \delta[i,j] - \|f(i) - f(j)\|_\infty \right| \right\} \tag{10.6}$$

$$= \max_{p,q \in P} \left\{ \left| \delta[p,q] - \|p - q\|_\infty \right| \right\} \tag{10.7}$$

The first equation is in terms of the objects, the second in terms of the configuration $P = f(\delta)$. We call this loss function *distortion*. It measures the maximum additive error of the embedding. Let us have a closer look at the properties of this loss function. Assume we know the distortion $\varepsilon^\star = \min_f\{\ell(\delta, f(\delta))\}$ of the optimal solution and search the corresponding point set $P^\star$. Then, for each pair of points $p, q \in P^\star$ it must hold that

$$-\varepsilon^\star \le \delta[p,q] - \max\{|p_x - q_x|, |p_y - q_y|\} \le \varepsilon^\star \ .$$

Note that the infinity norm destroys the symmetry suggested by the absolute value in (10.6) in the following sense. For the lower bound it must hold that

$$-\varepsilon^\star \le \delta[p,q] - |p_x - q_x| \ \text{ and } \ -\varepsilon^\star \le \delta[p,q] - |p_y - q_y| \ , \tag{10.8}$$

whereas for the upper bound it must hold that

$$\delta[p,q] - |p_x - q_x| \le \varepsilon^\star \ \text{ or } \ \delta[p,q] - |p_y - q_y| \le \varepsilon^\star \ . \tag{10.9}$$

We sum these two equations up in the following simple observation.

Observation 10.1.7. *Let $P^\star$ be the point set with minimum distortion $\varepsilon^\star$. For any two points $p, q \in P^\star$ the lower bound $-\varepsilon^\star \le \delta[p,q] - |p_z - q_z|$ must hold for both x- ($z = x$) and y-coordinate ($z = y$). The upper bound $\delta[p,q] - |p_z - q_z| \le \varepsilon^\star$ must hold for either x- or y-coordinates.*

The observation also suggests that x- and y-coordinates can be treated independently to a certain extend.

The Algorithm. The general idea of the algorithm is to do the following:

1. Guess $\varepsilon^\star = \min_f\{\ell(\delta, f(\delta))\}$
2. Find x-coordinates of an embedding with distortion $\varepsilon' \le c_1 \cdot \varepsilon^\star$.
3. For these x-coordinates find y-coordinates such that the resulting point set P has distortion no more than $\varepsilon'' \le c_2 \cdot \varepsilon'$.

We will see that the resulting point set P has distortion $\varepsilon'' \le 30\varepsilon^\star$. Guessing the right $\varepsilon^\star$ is done by a binary search in the end. The most interesting part of the algorithm is how the y-coordinates are found. For this reason we will discuss this part in detail. Then we will sketch how the x-coordinates are found.

The y-coordinates. Let us assume that we are given x-coordinates $X = \{p_x^1, \ldots, p_x^n\}$ of a point set P with the property that for all $p, q \in P$ it holds

$$-\varepsilon' \leq \delta[p,q] - \max\{|p_x - q_x|, |p_y - q_y|\} \leq \varepsilon' \ . \tag{10.10}$$

Let us call this assumption the *quality assumption*. It will not be possible to exactly recover the y-coordinates in P. But we will construct y-coordinates such that the resulting point set P' has the property

$$-5\varepsilon' \leq \delta[p,q] - \max\{|p_x - q_x|, |p_y - q_y|\} \leq 5\varepsilon'$$

for all $p, q \in P'$. We call such a solution a 5-approximation solution. In doing so, we see finding the y-coordinates as a problem in its own right, i.e. we only want to know how the distortion grows with respect to ε'.

From Observation 10.1.7 it is clear that all x-coordinate pairs (p_x, q_x) have to fulfill the lower bound. In the special case where all such pairs fulfill also the upper bound, it follows by the same observation that it suffices to find y-coordinates such that all y-coordinate pairs (p_y, q_y) fulfill the lower bound. In terms of the absolute value this means $|p_y - q_y| \leq \delta[p,q] + \varepsilon'$. It is easy to express this condition as linear constraints because $|x| \leq c$ is equivalent to $x \leq c$ and $-x \leq c$. The linear constraints become

$$-\varepsilon' - \delta[p,q] \leq p_y - q_y \leq \delta[p,q] + \varepsilon' \tag{10.11}$$

for all $p, q \in P'$. Note that in this special case we actually recover the 'correct' y-coordinates that fulfill (10.10). It follows that we only need to care about pairs (p_x, q_x) that do not fulfill the upper bound. We introduce the notion of edges between such points that model how bad a pair (p_x, q_x) exceeds the upper bound.

Definition 10.1.8. *If for a pair (p_x, q_x) it holds*

$$\delta[p,q] - |p_x - q_x| > 3\varepsilon' \ , \tag{10.12}$$

there is a strong edge *between p and q. If*

$$3\varepsilon' \geq \delta[p,q] - |p_x - q_x| > \varepsilon' \ , \tag{10.13}$$

there is a weak edge *between p and q. We denote the set of all strong edges by E_{s}, the set of all weak edges by E_{w}.*

In the special case where all edges are weak edges we can again find y-coordinates via linear programming with constraints of Type (10.11). The result is then at least a 3-approximation.

The set of strong edges E_{s} together with the points P form (a drawing of)[1] a graph G. For the correctness of the algorithm, it is important that the connected components of G can be separated by vertical lines, i.e., they do not overlap. The graph G does not have this property. Therefore, we define the edge set E' and claim that the resulting graph G' has the desired property.

[1] We will simply refer to the drawing of the graph G as *the graph G* because it will always be clear that we are discussing embeddings in the plane, and it will also be clear which drawing we refer to, namely the one given by P.

Definition 10.1.9. *Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the connected components of G and l_i (r_i) be the leftmost (rightmost) point in C_i. Let $\tilde{E}_\mathrm{w} \subset E_\mathrm{w}$ be the set of weak edges that have exactly one endpoint in some component C_i and the other one between l_i and r_i: $\tilde{E}_\mathrm{w} = \big\{\{p,q\} \in E_\mathrm{w} \mid \exists i\colon p \in C_i, q \notin C_i, l_i \leq q_x \leq r_i\big\}$. We define E' as $E_\mathrm{s} \cup \tilde{E}_\mathrm{w}$.*

The resulting graph G' has the desired property:

Claim 10.1.10. The connected components of G' can be separated by vertical lines that do not intersect any vertex. Moreover, every weak edge in G' is adjacent to at least one strong edge (see Figure 10.3).

The (easy) proof uses the definitions of strong and weak edges and the triangle inequality.

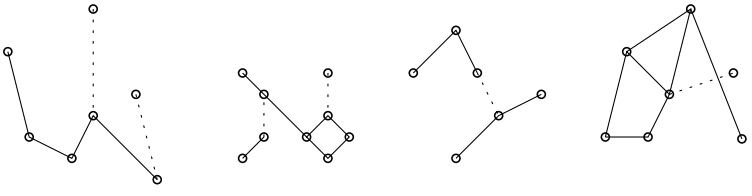


Fig. 10.3. Structure of G'. Solid lines represent strong edges, dotted lines weak edges. The four connected components do not overlap. Each weak edge is adjacent to at least one strong edge

Now that we know the structure of the graph G' that is constructed from the strong and weak edges it is interesting to see how exactly these edges can help to find an embedding. From Observation 10.1.7 we know that for a strong edge $\{p,q\}$ the y-coordinates p_y and q_y have to fulfill both the upper and the lower bound. If we try to express the upper bound similarly to (10.11), we run into the problem that $|x| \geq c$ is equivalent to $x \geq c$ or $-x \geq c$, which we cannot express as linear constraints of a linear program,[2] which have to be fulfilled simultaneously. But if we know whether $q_y \geq p_y$ or $p_y > q_y$, this problem vanishes and we can again use linear programming.

Definition 10.1.11. *For an edge $e = \{p,q\} \in E'$, $p_x \leq q_x$ we say that e is* oriented up *if $q_y \geq p_y$, we say that it is* oriented down *if $p_y > q_y$.*

Lemma 10.1.12. *If we know the orientation of all strong edges, we can compute a 3-approximation via linear programming.*

Proof. We construct the following linear program.[3]

[2] More generally, the constraint $|x| \geq c$ is non-convex, because the function $-|x|$ is not convex. On the other hand only convex optimization problems can be solved efficiently, which is a hint that we cannot express it in any way in a linear program. See [91] for more information on convex optimization.

[3] In the original paper [105] there are inconsistencies both in the definition of orientation and in the linear program.

$$
\begin{aligned}
&\min \delta \\
&\text{s. t.} \\
&\qquad -\delta \le \delta[p,q] - (q_y - p_y) \le \delta \qquad \begin{cases} \text{if } \{p,q\} \in E \\ \text{is oriented up} \end{cases} \\
&\qquad -\delta \le \delta[p,q] - (p_y - q_y) \le \delta \qquad \begin{cases} \text{if } \{p,q\} \in E \\ \text{is oriented down} \end{cases} \\
&-\delta[p,q] - \delta \le \qquad q_y - p_y \qquad \le \delta[p,q] + \delta \text{ if } (p,q) \notin E
\end{aligned} \tag{10.14}
$$

By the quality assumption there is a solution P that fulfills (10.10). This solution leads to a solution to the linear program with $\delta = \varepsilon'$. On the other hand, a solution with optimal value $\delta \le \varepsilon'$ to the linear program is only guaranteed to have distortion lower than $3\varepsilon'$: For all edges $\{p,q\} \in E'$ the first two inequalities guarantee that the distortion is at most δ. For all pairs $\{p,q\} \notin E_{\mathrm{w}}$ and $\{p,q\} \notin E_{\mathrm{s}}$ the third inequality bounds the distortion by δ; but for all $\{p,q\} \in E_{\mathrm{w}} \setminus E'$ the only guarantee for an upper bound is via the weakness of the edges. Thus, the guaranteed upper bound is $3\varepsilon'$ (see Definition 10.1.8). □

After Lemma 10.1.12 it is clear that it is useful to find out the orientation of the edges in E'. The following lemma states how to perform this task for one connected component of E'.

Lemma 10.1.13. *By fixing the orientation of one arbitrary edge in a connected component of G' we also fix the orientation of all other edges in this connected component.*

Proof. We show that the orientation of an edge $e = \{v,w\}$ fixes the orientation of all adjacent strong edges or of all edges if e itself is strong. Without loss of generality let v, w be oriented up. As $\{v,w\} \in E'$, both the upper and the lower bound must hold for the y-coordinate. Thus it holds (by the quality assumption) that

$$w_y - v_y + \varepsilon' \ge \delta[v,w] \ge w_y - v_y - \varepsilon' \ . \tag{10.15}$$

Furthermore, $w_y - v_y > w_x - v_x$ because $u, w \in E'$ and the upper bound must be established. For an adjacent edge $\{w,t\}$ that is oriented up, we get

$$
\begin{aligned}
\delta[v,t] \overset{\text{Obs } 10.1.6}{\ge} (t_y - v_y) - \varepsilon' &= (t_y - w_y) + (w_y - v_y) - \varepsilon' \\
&\overset{(10.15)}{\ge} \delta[v,w] + \delta[w,t] - 3\varepsilon' \ .
\end{aligned} \tag{10.16}
$$

As $\{w,t\}$ is a strong edge $\delta[w,t] - 3\varepsilon' > 0$, it holds

$$\delta[v,t] > \delta[v,w] \ . \tag{10.17}$$

As $\{v,w\}$ is (at least) a weak edge and $\{w,t\}$ is a strong edge, we get by combining Equations (10.12), (10.13), and (10.16)

$$\delta[v,t] > (w_x - v_x) + |t_x - w_x| + \varepsilon' \ge |t_x - v_x| + \varepsilon' \ . \tag{10.18}$$

In the other case where $\{w,t\}$ is oriented down we get

$$\begin{aligned}\delta[v,t] \leq \|t-v\|_\infty + \varepsilon' &\leq \max\{|t_x - v_x| + \varepsilon', |(w_y - v_y) - (w_y - t_y)| + \varepsilon'\} \\ &\leq \max\{|t_x - v_x| + \varepsilon', \delta[v,w] + \varepsilon' - \delta[w,t] + \varepsilon' + \varepsilon'\} \\ &\leq \max\{|t_x - v_x| + \varepsilon', \delta[v,w]\} \ . \quad (10.19)\end{aligned}$$

Where the first inequality follows by the quality assumption, the second by the orientation of the edges, the third by the fact that both are edges and Observation 10.1.7, and the fourth because $\{w,t\}$ is a strong edge.

Equations (10.17) and (10.18) together contradict (10.19); therefore it is possible to find out the orientation of edge $\{w,t\}$. A similar argument shows that in the case where $\{w,t\}$ is a weak edge and $\{v,w\}$ is strong we can find out the orientation of $\{w,t\}$. As in a connected component each weak edge is connected to a strong edge, we can iteratively find the orientation of all edges in it by fixing one. □

The previous two lemmata together already yield a 3-approximation solution if G' consists of a single connected component.

If G' consists of more than one connected component the algorithm arbitrarily fixes the orientation of one edge in each connected component. In the case where all these relative orientations are accidentally chosen correctly, we still have a 3-approximation. Surprisingly, even if the relative orientations are chosen incorrectly we still have a 5-approximate solution. The intuition behind this result is that between connected components there are no strong edges (but potentially weak edges) and therefore by choosing the wrong relative orientation between the components not too much distortion is created. The following lemma makes this statement precise.

Lemma 10.1.14. *There is a 5-approximate solution for every relative orientation between the edges of the components.*

Sketch of Proof. The idea of the proof is to show how we can transform the optimal solution (i.e. the solution in which the orientations are chosen optimally) into a solution with arbitrary relative orientation. To this end, we scan through the components $\{C_1, \ldots, C_k\}$ from left to right and flip in the ith step components $\{C_i, \ldots, C_k\}$ by an appropriately chosen horizontal line if the orientations in C_i in the arbitrary and the optimal solution disagree. For this choice to be possible it is necessary that the components can be separated by vertical lines. Then it needs to be established that this (clever) choice of flips does not create too much additional distortion. □

The x-coordinates. We will see a sketch of the method to find the x-coordinates. We start again by stating the *quality assumption* (q. a.) that the optimal embedding has error $\varepsilon^\star$. Now let the diameter be given by the points p and q and assume it is defined by $q_x - p_x$. As the origin of the coordinate system is arbitrary, we can fix p at $(0,0)$.

Let A be the set of points $v \in P \setminus \{p, q\}$ with $\delta[p, q] + k\varepsilon^\star \geq \delta[p, v] + \delta[v, q]$ for some constant k.

$$A = \{v \in P \setminus \{p, q\} \mid \delta[p, q] + k\varepsilon^\star \geq \delta[p, v] + \delta[v, q]\}$$

Points in A fulfill the following two inequalities

$$v_x \overset{\text{q. a.}}{\leq} \delta[p, v] + \varepsilon^\star \ , \tag{10.20}$$

$$v_x \overset{2\times\text{q. a.}}{\geq} \delta[p, q] - \delta[v, q] - 2\varepsilon^\star \overset{v\in A}{\geq} \delta[p, v] - (k + 2)\varepsilon^\star \ . \tag{10.21}$$

If we fix v_x at the arithmetic mean of the two right hand sides $v_x = (2\delta[p, v] - (k + 1)\varepsilon^\star)/2 = \delta[p, v] - ((k + 1)\varepsilon^\star)/2$, the additive error with respect to the optimal value for v_x is bounded by $(k+3)\varepsilon^\star/2$. If all points $v \in P \setminus \{p, q\}$ are in A, the problem is solved. In the case $P \setminus A \neq \emptyset$, the algorithm makes a (lengthy) case distinction that we will not present in detail. The general idea is to partition the set $P \setminus A$ into finer sets B, C, and D. Then, similar to the case of the problem with the y-coordinates, equations are derived that hold under the assumptions that a point p' is in B, C, or D. As the equations are again contradictory, it is possible to find out to which of the sets p' belongs. From this membership it is then possible to find a good approximation of the x-coordinate.

This completes the presentation of Bădoiu's algorithm. To sum up, it achieves its goal of guaranteeing a constant loss with respect to the optimal embedding by connecting the MDS-problem to a discrete structure—the graph G' together with an orientation on it. This makes possible the use of a combinatorial algorithm. Note that on bad instances the distortion of the constructed embedding can still be very high if even the optimal embedding has high distortion, see the bibliography for references on this problem.

10.1.3 Clustering Based Methods

In the preceding sections, we have discussed how to derive measures of structural equivalence from the adjacency matrix A of G and how to refine them by multidimensional scaling. We will now investigate clustering based methods which use such a measure $\delta_{i,j}$ for computing an actor set partition $\mathcal{P}$ that hopefully corresponds to the true positional structure of G.

Having a symmetric distance measure $\delta_{i,j}$, we could of course apply general clustering techniques in order to identify subsets of actors which are expected to represent one position. Chapter 8 gives an overview over this broad area of network analysis. Nevertheless, in the area of blockmodeling a rather simple clustering heuristic has been implemented and applied by most researchers.

In general, we speak of *hierarchical clustering* if the clustering algorithm starts with clusters $P_1, \ldots, P_n$ with $P_i := \{v_i\}$ before it iteratively joins pairs of clusters with minimal distance $d(P_k, P_\ell)$. Different measures $d\colon \mathcal{P}(V) \times \mathcal{P}(V) \to \mathbb{R}$ for the inter-cluster distance have been proposed. This clustering framework generates a hierarchy of subsets and finally results in a single cluster containing

all actors of V. Then, the researcher has to select a minimum distance β that has to be between two clusters resp. positions.

Formally, we start with a partition $\mathcal{P}_1 = \{\{v_1\}, \ldots, \{v_n\}\}$. In general, we have a current partition $\mathcal{P}_x$ and compute $\mathcal{P}_{x+1}$ by joining two different clusters $P_{k^*}, P_{\ell^*} \in \mathcal{P}_x$, i. e., $\mathcal{P}_{x+1} := (\mathcal{P}_x \setminus \{P_{k^*}, P_{\ell^*}\}) \cup \{P_{k^*} \cup P_{\ell^*}\}$ for $(P_{k^*}, P_{\ell^*}) := \arg\min_{P_k, P_\ell \in \mathcal{P}_x} d(P_k, P_\ell)$. The result is a sequence $\mathcal{P}_1, \ldots, \mathcal{P}_n$ of partitions. The researcher has to choose a threshold value β which is used to discard cluster unions incorporating cluster pairs of larger distance than β. After having pruned the hierarchy in this way, the resulting actor subsets are taken as the positions of the blockmodeling analysis.

Cluster Distance Measures. There are four popular ways how to define the cluster distance d (see [18]). All of them have been justified by successful analyses of positional structures and may be selected depending on the relational data of G. However, the single linkage hierarchical clustering is not considered to be very good because of chaining effects. Nevertheless, it is able to discover well-separated shape clusters.

Single linkage $d^{\mathrm{s}}(P_k, P_\ell)$. In case of single linkage, we set $d^{\mathrm{s}}(P_k, P_\ell) := \min\{\delta_{i,j} \mid v_i \in P_k,\, v_j \in P_\ell\}$. That is, the smallest distance between two members v_i of P_k and v_j of P_ℓ is taken as distance between the clusters P_k and P_ℓ.

Complete linkage $d^{\mathrm{c}}(P_k, P_\ell)$. In case of complete linkage, we demand that every pair $(v_i, v_j) \in P_k \times P_\ell$ has at most distance $d^{\mathrm{c}}(P_k, P_\ell)$, i. e., $d^{\mathrm{c}}(P_k, P_\ell) := \max\{\delta_{i,j} \mid v_i \in P_k,\, v_j \in P_\ell\}$.

Average linkage $d^{\mathrm{a}}(P_k, P_\ell)$. In contrast to the maximum or minimum actor-wise distances of the two previous measures, the average linkage takes the average actor distances into account. The average linkage distance d^{a} is defined by

$$d^{\mathrm{a}}(P_k, P_\ell) := \frac{1}{|P_k| \cdot |P_\ell|} \cdot \sum_{v_i \in P_k,\, v_j \in P_\ell} \delta_{i,j} \quad .$$

Average group linkage $d^{\mathrm{g}}(P_k, P_\ell)$. Finally, the average group linkage considers the average distance between all actor-pairs of the join of P_k and P_ℓ. This result $P_k \cup P_\ell$ contains $\binom{|P_k|+|P_\ell|}{2}$ actor pairs, and it is

$$d^{\mathrm{g}}(P_k, P_\ell) := \sum_{v_i \in P_k,\, v_j \in P_\ell} \delta_{i,j} \Big/ \binom{|P_k| + |P_\ell|}{2} \quad .$$

Burt's Algorithm. We finally want to mention a special well-established hierarchical clustering approach to blockmodeling that was presented by Burt [107] in 1976. Basically, he uses the Euclidian distance $\delta^{\mathrm{e}}_{i,j}$ together with the single

linkage cluster distance d^{s}. Furthermore, Burt assumes that the vector $\delta^{\mathrm{e}}_{i,\cdot} := \left(\delta^{\mathrm{e}}_{i,j}\right)_j$ of the observed actor distances between v_i and the other actors is composed mainly of two components: First, a position-dependent vector $p_k \in \mathbb{R}^n$ which contains the hypothetical distances of an ideal member of position $k := P(v_i)$ to all other actors. Second, $\delta^{\mathrm{e}}_{i,\cdot}$ is influenced by an additive error component $w_i \in \mathbb{R}^n$ as small as possible which is (besides of the covariance) used to explain the deviations of $\delta^{\mathrm{e}}_{i,\cdot}$ from p_k. In detail, Burt's model states

$$\delta^{\mathrm{e}}_{i,\cdot} = \mathrm{cov}\left(\delta^{\mathrm{e}}_{i,\cdot}, p_k\right) \cdot p_k + w_i \ ,$$

where $k := P(v_i)$ and $\mathrm{cov}\left(\delta^{\mathrm{e}}_{i,\cdot}, p_k\right)$ is the covariance between $\delta^{\mathrm{e}}_{i,\cdot}$ and p_k. That is, vectors $\delta^{\mathrm{e}}_{i,\cdot}$ and $\delta^{\mathrm{e}}_{j,\cdot}$ for $P(v_i) = P(v_j)$ may only differ by their mean, while the remaining deviation w_i resp. w_j should be small for a good blockmodel.

Burt gives methods to compute the unknown components p_k, $1 \le k \le L$ and w_i, $1 \le i \le n$, from the distances $\delta_{i,j}$ minimizing the error components w_i. These results can then be used for further interpretation of the blockmodel or to evaluate its plausibility by means of the magnitudes of the error components.

10.1.4 CONCOR

Besides clustering based methods, the CONCOR algorithm represents the most popular method in traditional blockmodeling. It was presented by Breiger, Boorman, and Arabie [99] in 1975 and has been extensively used in the 70's and 80's.

CONCOR is a short form of *convergence of iterated correlations*. This stems from the observation in sociological applications that the iterated calculation of correlation matrices of the adjacency matrix A typically converges to matrices of special structure. In detail, the algorithm computes the symmetric *correlation matrix* $C_1 := \left(c^{(1)}_{i,j}\right)_{i,j} := (c_{i,j})_{i,j}$ of A corresponding to Definition 10.1.2. Then, it iteratively computes the correlation matrix $C_{s+1} := \left(c^{(s+1)}_{i,j}\right)_{i,j}$ of $C_s := \left(c^{(s)}_{i,j}\right)_{i,j}$. This process is expected to converge to a matrix $\mathcal{R} := (r_{i,j})_{i,j}$ consisting solely of -1 and $+1$ entries. Furthermore, it has been observed that there typically exists a permutation $\pi \in \Sigma_n$ on the set of actor indices $\{1, \ldots, n\}$ and an index i^* such that the rows and columns of $\mathcal{R}$ can be permuted to a matrix $\mathcal{R}^* := \left(r^*_{i,j}\right)_{i,j} := \left(r_{\pi(i),\pi(j)}\right)_{i,j}$ with $r^*_{i,j} = 1$ for $(i,j) \in (\{1, \ldots, i^*\} \times \{1, \ldots, i^*\}) \cup (\{i^*+1, \ldots, n\} \times \{i^*+1, \ldots, n\})$ and $r^*_{i,j} = -1$ for $(i,j) \in (\{i^*+1, \ldots, n\} \times \{1, \ldots, i^*\}) \cup (\{1, \ldots, i^*\} \times \{i^*+1, \ldots, n\})$ (see Figure 10.1.4).

$$\mathcal{R}^* = \left(\begin{array}{c|c} +1 & -1 \\ \hline -1 & +1 \end{array}\right)$$

Fig. 10.4. Layout of matrix $\mathcal{R}^*$

Assume that the actor set partition $\mathcal{P} = \{P_1, \ldots, P_L\}$ reflects the true positional structure of G. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be disjoint subsets of $\mathcal{P}$ with $\mathcal{P}_1 \cup \mathcal{P}_2 = \mathcal{P}$ such that actors of different positions $P_k \in \mathcal{P}_x$ and $P_\ell \in \mathcal{P}_x$, $x \in \{1, 2\}$, are more similar to each other than actors of positions $P_{k'} \in \mathcal{P}_1$ and $P_{\ell'} \in \mathcal{P}_2$. That is, we assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ are the result of some clustering method dividing $\mathcal{P}$ into two subsets.

The CONCOR algorithm is based on the assumption that the correlation coefficient $c_{i,j}^{(s)}$ between actors v_i, v_j of the same part $\mathcal{P}_x$ converges to 1, while this index converges to -1 if v_i and v_j are placed in different halves of $\mathcal{P}$. Therefore, the algorithm is iterated until for some s^* matrix C_{s^*} is close enough to $\mathcal{R}$; then, the actors V are divided into $V_1 := \{v_{\pi(1)}, \ldots, v_{\pi(i^*)}\}$ and $V_2 := \{v_{\pi(i^*+1)}, \ldots, v_{\pi(n)}\}$. Now, V_x, $x \in \{1, 2\}$, should correspond to $\bigcup_{k \in \mathcal{P}_x} P_k$.

In order to finally obtain the positional subsets $P_1, \ldots, P_L$ of V, CONCOR is recursively applied to the induced subgraphs $G_x := (V_x, E \cap (V_x \times V_x))$ for $x \in \{1, 2\}$, until the user decides to stop. One criterion for this could be the speed of convergence to $\mathcal{R}$; in most papers reporting on applications of CONCOR, this decision is taken heuristically depending on G. That is, we get a subdivision tree of V (often called *dendrogram*) whose leaves correspond to the final output $\mathcal{P} = \{P_1, \ldots, P_L\}$.

Criticism. Although this method is well-established and was applied in many blockmodel analyses of social data, there has also been a lot of criticism of CONCOR. Doreian [160], Faust [199], and Sim and Schwartz [520] applied it to several hypothetical networks whose positional structure was known and experienced CONCOR to be unable to recover the correct blockmodel.

Breiger, Boorman, and Arabie proposed the CONCOR algorithm without a mathematical justification for its procedure or an idea what it exactly computes. In [508], Schwartz approaches this problem by investigating CONCOR's mathematical properties. Experiments show that for most input graphs G, the result matrix C_{s^*} has rank 1. It can be easily proved that this property implies the special structure of $\mathcal{R}$ (that is, $r_{i,j} \in \{-1, 1\}$ and the existence of π and i^*). Schwartz also gives concrete counterexamples for which this does not hold. Furthermore, the only eigenvector of such a rank 1 matrix seems almost always to correspond to the *first principal component* obtained by the statistical method of principal component analysis (PCA) [335]. That is why there seems to be no substantial reason to use CONCOR instead of a PCA, whose properties are well-understood.

10.1.5 Computing the Image Matrix

It was already mentioned that the partition $\mathcal{P}_\simeq$ of the structural equivalence classes causes the $\mathcal{P}_\simeq$-permuted adjacency matrix $A^* = \left(a^*_{i,j}\right)_{i,j}$ to consist solely of 0- and 1-blocks $A^{k,\ell}$, $1 \leq k, \ell \leq L$. It has also been argued that $\simeq$ is not suited to retrieve the hidden positional structure of real-world graphs G. Therefore,

heuristic methods based on some relaxation of $\simeq$ have been introduced in the preceding sections.

Let $\mathcal{P}$ be an actor set partition produced by such a heuristic blockmodeling method. The corresponding $\mathcal{P}$-permuted matrix A^* is expected to consist of blocks $A^{k,\ell}$ containing both zeros and ones. In order to decide if position P_k is adjacent to P_ℓ in the reduced graph represented by the image matrix $B = (b_{i,j})_{i,j} \in \{0,1\}^{L\times L}$, several criterions have been proposed in the literature. We describe the three most popular ones for the case $k \neq \ell$.

Zeroblock Criterion. The zeroblock criterion corresponds to the assumption that two positions $P_k, P_\ell \in \mathcal{P}$ are only non-adjacent if the k–ℓ block $A^{k,\ell}$ of the $\mathcal{P}$-permuted matrix A^* solely contains zeros, i. e., $b_{k,\ell} = 0 :\Leftrightarrow \forall (v_i, v_j) \in P_k \times P_\ell \colon (v_i, v_j) \notin E$. If the zeroblock criterion is used, the image matrix B corresponds to the adjacency matrix of the role graph introduced in Definition 9.0.3.

Oneblock Criterion. In contrast to the zeroblock criterion, the oneblock criterion corresponds to the assumption that two positions $P_k, P_\ell \in \mathcal{P}$ are only adjacent if $A^{k,\ell}$ solely contains ones, i. e., $b_{k,\ell} = 1 :\Leftrightarrow \forall (v_i, v_j) \in P_k \times P_\ell \colon (v_i, v_j) \in E$.

α-Density Criterion. In most cases, we do not assume that a single entry in a block $A^{k,\ell}$ decides about the relation between positions $P_k, P_\ell \in \mathcal{P}$. We would rather accept small deviations from perfect 0- or 1-blocks and, therefore, want to know to which block type $A^{k,\ell}$ is more similar.

First, we define a supporting identifier for the number of non-diagonal elements of a block.

Definition 10.1.15. *The* block cardinality $S_{k,\ell}$ *of block* $A^{k,\ell}$ *is defined by* $S_{k,\ell} := |P_k| \cdot |P_\ell|$ *if* $k \neq \ell$ *and* $S_{k,\ell} := |P_k| \cdot (|P_\ell| - 1)$ *if* $k = \ell$.

Definition 10.1.16. *The* block density $\Delta_{k,\ell}$ *of block* $A^{k,\ell}$ *is defined by*

$$\Delta_{k,\ell} := \frac{1}{S_{k,\ell}} \cdot \sum_{v_i \in P_k,\, v_j \in P_\ell} a_{i,j} \quad .$$

This definition excludes the diagonal elements of A. Using the α-density criterion, we set $b_{k,\ell}$ to zero iff $\Delta_{k,\ell}$ is smaller than a certain threshold value α, i. e., $b_{k,\ell} = 0 :\Leftrightarrow \Delta_{k,\ell} < \alpha$. Often, the over-all density of the adjacency matrix A is used as threshold α, i. e., $\alpha := \sum_{1\leq i,j\leq n} a_{i,j}/(n(n-1))$. That is, two positions P_k and P_ℓ are decided to be connected if the relative edge number in their induced subgraph is at least as high as the relative edge number of the whole graph G.

10.1.6 Goodness-of-Fit Indices

Due the heuristical nature of the algorithms discussed in this section on deterministic models, it makes sense to apply several different methods on the same

data and to compare the results. In order to decide which result to accept as the best approximation of the true positional structure of the input graph G, quality or *goodness-of-fit* indices are needed to evaluate the plausibility of a blockmodel.

So let us assume that $B = (b_{i,j})_{i,j}$ is an image matrix produced by some blockmodeling method for graph G with adjacency matrix A and corresponding actor set partition $\mathcal{P}$.

Density Error. A blockmodel can be evaluated by comparing $A = (a_{i,j})_{i,j}$ with a hypothetical ideal adjacency matrix induced by B. Such an ideal matrix would have only 1-entries in blocks $A^{k,\ell}$ with $b_{k,\ell} = 1$ resp. 0-entries if $b_{k,\ell} = 0$ (excluding diagonal elements $a_{i,i}$). In detail, we compute the sum of error differences between the block densities $\Delta_{k,\ell}$ and the elements of B.

Definition 10.1.17. *The* density error e_{d} *of image matrix B is defined by*

$$e_{\mathrm{d}} := \sum_{1 \le k,\ell \le L} |b_{k,\ell} - \Delta_{k,\ell}| \ .$$

It is $e_{\mathrm{d}} \in \left[0, L^2\right]$. The smaller e_{d}, the more structural equivalent are actors of same position. Therefore, the blockmodel with the smallest density error is expected to be a better representation of the positional structure of G.

Carrington-Heil-Berkowitz Index. A second widely used goodness-of-fit index is the Carrington-Heil-Berkowitz index [112], which is tailored for evaluating blockmodels that have been created using the α-density criterion (see Section 10.1.5). Remember that we define $b_{i,j} = 0$ iff the block density $\Delta_{k,\ell}$ is smaller than the threshold value α. The choice of $b_{k,\ell}$ seems to be more reliable if the difference $|\Delta_{k,\ell} - \alpha|$ is large. The best possible difference for $b_{k,\ell} = 0$ is α, while it is $1 - \alpha$ for $b_{k,\ell} = 1$.

The Carrington-Heil-Berkowitz index is the normalized weighted sum of squared ratios of the observed difference $|\Delta_{k,\ell} - \alpha|$ to the ideal one α resp. $1 - \alpha$. Again, let $S_{k,\ell}$ be the block cardinality of block $A^{k,\ell}$ defined in Definition 10.1.15.

Definition 10.1.18. *Let $t_{k,\ell} := 1$ for $b_{k,\ell} = 0$ and $t_{k,\ell} := 1/(1-\alpha)$ for $b_{k,\ell} = 1$. The* Carrington-Heil-Berkowitz index e_{b} *of image matrix B is defined by*

$$e_{\mathrm{b}} := \sum_{1 \le k,\ell \le L} \left(\frac{\Delta_{k,\ell} - \alpha}{t_{k,\ell} \cdot \alpha} \right)^2 \cdot \frac{S_{k,\ell}}{n(n-1)} \ .$$

That is, the summand for block $A^{k,\ell}$ is weighted by the ratio $S_{k,\ell}/(n(n-1))$ it contributes to the whole matrix A. It is $e_{\mathrm{b}} \in [0, 1]$, and a value of 1 indicates perfect structural equivalence. A value close to 0 stems from all $\Delta_{k,\ell}$s being close to α. Then, many values $b_{k,\ell}$ are expected to be wrong because the α-density criterion classified them just due to little random deviations of $\Delta_{k,\ell}$ around α. Hence, the corresponding blockmodel is assumed to be bad.

10.1.7 Generalized Blockmodeling

Batagelj, Ferligoj, and Doreian [42, 45, 206] present an approach called *generalized blockmodeling*. They consider blockmodeling as an optimization problem on the set of partitions $\Pi := \{\mathcal{P} = (P_1, \ldots, P_L) \mid V = P_1 \uplus \cdots \uplus P_L\}$, where L is part of the input.

In the classical blockmodeling framework, the entry $b_{k,\ell} \in \{0, 1\}$ of the image matrix B represents a hypothesis on the existence of a connection between positions P_k and P_ℓ. That is, the blocks of a good blockmodel are assumed to be filled mainly either with ones or with zeros. In contrast, generalized blockmodeling is not just based on a relaxation of structural equivalence, but allows positions to be related by a variety of different connection types $\mathcal{T}$. Hence, the entries $b_{k,\ell}$ of B now take values in $\mathcal{T}$.

The optimization problem that has to be solved is defined by an error measure $D\colon \Pi \to \mathbb{R}$ with $D(\mathcal{P}) := \sum_{1 \le k,\ell \le L} d(P_k, P_\ell)$ summing up blockwise errors $d\colon \mathcal{P}^2 \to \mathbb{R}$. The final result is an optimal partition $\mathcal{P}^* := \arg\min_{\mathcal{P} \in \Pi}\{D(\mathcal{P})\}$. The authors use a local search heuristic to find $\mathcal{P}^*$. Starting from an initial (possibly random) partition, a current partition $\mathcal{P}$ is iteratively improved by replacing it with the best partition $\mathcal{P}' \in \mathcal{N}(\mathcal{P})$ from the *neighborhood* $\mathcal{N}(\mathcal{P})$ of $\mathcal{P}$. This neighborhood is defined by all partitions resulting from one of two operations applied on $\mathcal{P}$:

1. A *transition* moves some node v from its position P_k to another position P_ℓ.
2. A *transposition* exchanges the positional membership of two distinct nodes $v \in P_k$ and $v_j \in P_\ell$, $k \neq \ell$.

This optimization method does not depend on D and leaves freedom for the definition of the blockwise error measure d. It is assumed that each connection type $T \in \mathcal{T}$ has a set $I(T)$ of ideal blocks that fit T perfectly. For any block $A^{k,\ell}$ according to the current partition $\mathcal{P}$, the type-specific error measure $\delta(A^{k,\ell}, T) \in \mathbb{R}$ gives the minimal distance of $A^{k,\ell}$ to any block of $I(T)$. Then, we assume that P_k and P_ℓ are related by some connection type T with minimal distance to $A^{k,\ell}$, that is, $b_{k,\ell} := \arg\min_{T \in \mathcal{T}}\{\delta(A^{k,\ell}, T)\}$. Some priority order on $\mathcal{T}$ can be used to determine $b_{k,\ell}$ if the nearest connection type is not unique. Alternatively, $\mathcal{P}$ can be optimized for a pre-defined image matrix B or a whole class of image matrices (see, e. g., [44, 162]). From B, the blockwise errors are obtained by $d(P_k, P_\ell) := \delta(A^{k,\ell}, b_{k,\ell})$.

Some Proposed Connection Types. The generalized blockmodeling framework can be used with arbitrary user-defined connection types. In [45], Batagelj, Ferligoj, and Doreian propose a set of nine types motivated from different existing blockmodeling approaches, which will be briefly discussed in the following. In order to simplify the descriptions, we assume that blocks contain no diagonal elements of A.

Complete and null. This corresponds to the zeroblock- and oneblock-criterion in classical blockmodeling. An ideal complete (null) block $A^{k,\ell}$ contains solely

ones (zeros) and represents the case that all nodes P_k are connected to all nodes P_ℓ (no node of P_k is connected to any node of P_ℓ). If all blocks are either ideal complete or ideal null, the partition corresponds to the equivalence classes of a structural equivalence relation.

Row-dominant and col-dominant. An ideal row-dominant (col-dominant) block contains at least one row (column) entirely filled with ones. That is, there is at least one actor in the row position connected to all of the other group (there is at least one actor in the column position to which every actor from the row position is connected).

Row-regular, col-regular, and regular. An ideal row-regular (col-regular) block contains at least one one in each row (column). That is, every actor in the row position is connected to at least one of the column position (every actor in the column position is connected from at least one of the row position). A block is called regular if it is both row-regular and col-regular. If all blocks are ideal regular, the partition corresponds to a regular role assignment (see Definition 9.0.3).

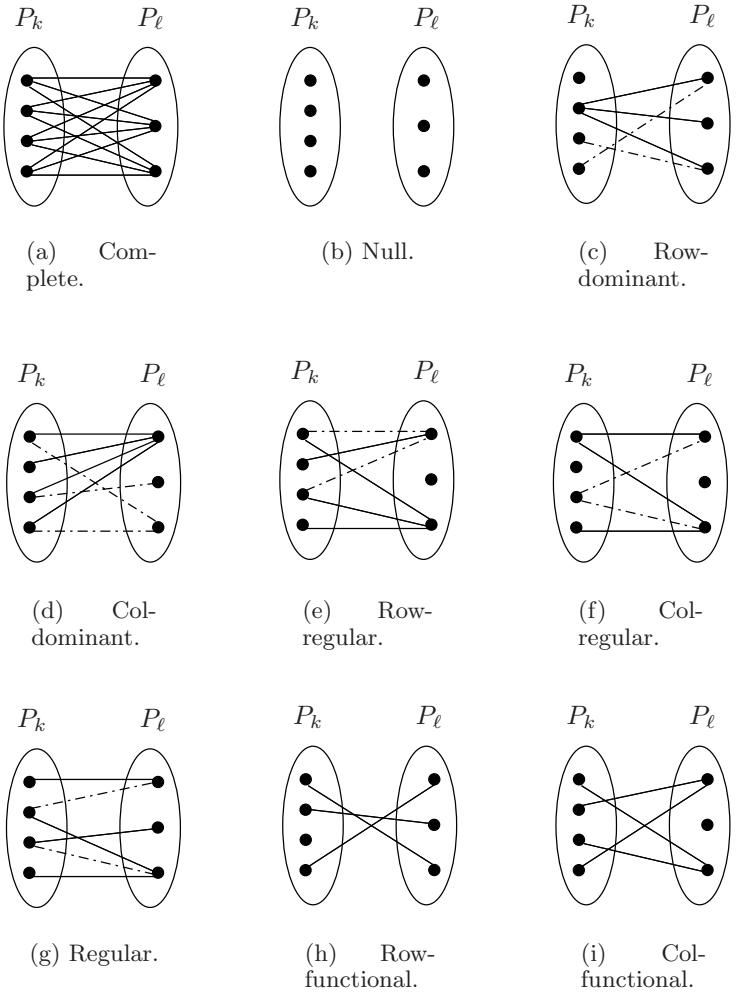
Row-functional and col-functional. An ideal row-functional (col-functional) block contains exactly one one in each column (row). That is, every actor in the column position is connected to exactly one of the row position (every actor in the row position is connected from exactly one of the column position).

Figure 10.1.7 illustrates ideal subgraphs of each connection type, while Table 10.1.7 lists the definitions of deviation functions $\delta(P_k, P_\ell, T)$ for each of the nine types. These sum up elements of rows resp. column which do not fit the ideal block of a particular connection type.

Generalized blockmodeling has been successfully applied to several networks (see, e. g., [163]). The method seemed to be limited to networks of at most hundreds actors.

10.2 Stochastic Models

Recently, many researchers have advocated the use of stochastic models instead of deterministic models because they make explicit the assumptions on the model and enable us to make precise statements on the validity of hypotheses about social networks. In Section 10.2.1 we present the p_1 model that was the first stochastic model to become established in social network analysis. Then we investigate in the context of the p_1 model how hypothesis testing can be done for stochastic models in Section 10.2.2. In Section 10.2.3 we explore the use of the p_1 model for blockmodeling. We also describe stochastic models that are more adapted to the specific setting of blockmodeling in Section 10.2.4. Finally, we present an advanced stochastic model in Section 10.2.5.



(a) Complete. (b) Null. (c) Row-dominant.

(d) Col-dominant. (e) Row-regular. (f) Col-regular.

(g) Regular. (h) Row-functional. (i) Col-functional.

Fig. 10.5. Examples of ideal subgraphs for the different block types of generalized blockmodeling as proposed by Batagelj, Ferligoj, and Doreian. Dashed lines are not necessary for the blocks to be ideal

10.2.1 The p_1 Model

If we want to understand blockmodeling from a statistical point of view, we need to make an assumption on a model that generates the data. In the setting of parameterized statistics, this is a parameterized probability distribution. As the data in blockmodeling is a directed graph, we need to understand suitable distributions on graphs. Historically, the p_1 model was one of the first such

Table 10.1. Deviation functions for the connection types of generalized blockmodeling as proposed by Batagelj, Ferligoj, and Doreian. (For blocks containing diagonal elements, the formulas have to be slightly modified)

T	$\delta(P_k, P_\ell, T)$
complete	$\lvert P_k\rvert \cdot \lvert P_\ell\rvert - c$
null	c
row-dominant	$(\lvert P_\ell\rvert - M_\mathrm{r}) \cdot \lvert P_k\rvert$
col-dominant	$(\lvert P_k\rvert - M_\mathrm{c}) \cdot \lvert P_\ell\rvert$
row-regular	$(\lvert P_k\rvert - N_\mathrm{r}) \cdot \lvert P_\ell\rvert$
col-regular	$(\lvert P_\ell\rvert - N_\mathrm{c}) \cdot \lvert P_k\rvert$
regular	$(\lvert P_k\rvert - N_\mathrm{r}) \cdot \lvert P_\ell\rvert + (\lvert P_\ell\rvert - N_\mathrm{c}) \cdot \lvert N_\mathrm{r}\rvert$
row-functional	$c - N_\mathrm{r} + (\lvert P_k\rvert - N_\mathrm{r}) \cdot \lvert P_\ell\rvert$
col-functional	$c - N_\mathrm{c} + (\lvert P_\ell\rvert - N_\mathrm{c}) \cdot \lvert P_k\rvert$

c	Number of ones in $A^{k,\ell}$.
N_r	Number of non-null rows in $A^{k,\ell}$.
N_c	Number of non-null column in $A^{k,\ell}$.
M_r	Maximal row-sum in $A^{k,\ell}$.
M_c	Maximal column-sum in $A^{k,\ell}$.

distributions that has been used in social network analysis. Its main advantages are its intuitive appeal and its simplicity.

Generally, we want to express for each graph x on n nodes with an $n \times n$ adjacency matrix A the probability that it is drawn from the set of all possible graphs on n nodes $\mathcal{G}_n$. If we define a random variable X that assumes values in $\mathcal{G}_n$, we could express any distribution by defining $\Pr[X = x]$ explicitly for all $x \in \mathcal{G}_n$. Of course, this direct approach becomes infeasible already for moderately big n. We are interested in an simple, 'intuitive' distribution. Therefore, it is natural to try to connect $\Pr[X = x]$ to the presence or absence of individual edges x_{ij} in x, which we express by a $\{0, 1\}$-random variable:

$$X_{ij} = \begin{cases} 1 & \text{if edge } x_{ij} \text{ present in } x, \\ 0 & \text{otherwise.} \end{cases}$$

Note that in contrast to the part on deterministic models concrete graphs are called x in this part and the edges are referred to as x_{ij}. The reason for this change in notation is that graphs are now seen as an outcome of a draw from a distribution that is represented by a random variable X. Probably one of the easiest ways to specify a distribution is to set $\Pr[X_{ij} = 1] = 1/2$ for all $i, j \in \{1, \ldots, n\}, i \neq j$ and to assume that all X_{ij} are independent. This is equivalent to giving all graphs in $\mathcal{G}_n$ the same probability, i.e. $\Pr[X = x] = 2^{-n(n-1)}$. Obviously, this model is too simple to be useful. It is not possible to infer anything from it as the distribution is not parameterized. A very simple parameterization is to set $\Pr[X_{ij} = 1] = a_{ij}$. If we assume independence of all

X_{ij} we get

$$\Pr[X = x] = \prod_{1 \le i,j \le n} a_{ij}^{x_{ij}} (1 - a_{ij})^{1 - x_{ij}} \ .$$

One reason why this closed form is so simple is that we have assumed independence. On the other hand this model has serious drawbacks: First, by the independence assumption it is impossible to infer how likely it is that a relation from a to b is reciprocated. Unfortunately, this question is at the heart of many studies in social network analysis. Second, the model has too many parameters, which cannot be estimated from a single observation (i.e. the observed social network), this problem is often referred to as this model not being *parsimonious*.

The p_1 model that we derive now from a first 'tentative' distribution overcomes these drawbacks. In order to model reciprocation effects let us assume statistical dependence of the variables X_{ij} and X_{ji} for all $1 \le i < j \le n$ which are together called the *dyad* $D_{ij} := X_{ij} \times X_{ji}$. Let the rest of the variables still be independent, i.e. the probability of an edge from a to b is only dependent on the existence of an edge from b to a. The resulting distribution, which we call p_t (for tentative), is easy to specify in terms of dyads. We set

$$\begin{aligned} \Pr[D_{ij} = (1,1)] &= m_{ij} \ , \\ \Pr[D_{ij} = (1,0)] &= a_{ij} \ , \\ \Pr[D_{ij} = (0,0)] &= n_{ij} \ , \end{aligned}$$

with $m_{ij} + a_{ij} + a_{ji} + n_{ij} = 1$. Here, m_{ij} stands for the probability of a mutual relation, a_{ij} for an asymmetric relation and n_{ij} for no relation between actors i and j. For the probability of a given graph x we get

$$p_t(x) = \Pr[X = x] = \prod_{i<j} m_{ij}^{x_{ij}(1-x_{ij})} \prod_{i \ne j} a_{ij}^{x_{ij}(1-x_{ji})} \prod_{i<j} n_{ij}^{(1-x_{ij})(1-x_{ji})} \ .$$

This formula completely specifies p_t. We have still the problem of too many variables, which we will address soon. From a statistical point of view, it is desirable to find out into which class of distributions p_t falls, so that the standard theory can be applied. For p_t we show that it belongs to the *exponential family* of distributions:

Definition 10.2.1. *A distribution of a random variable X belongs to the* s-dimensional exponential family *iff its probability density or frequency function can be written as*

$$f(x, \eta) = \exp\left[\sum_{i=1}^{s} \eta_i T_i(x) - A(\eta)\right] h(x) \ ,$$

where the η_i are parameters, A is a real-valued function of the parameters, the T_i are real-valued statistics, and the factor $h(x)$ is any function depending only on x.

To see that p_t has indeed this form we first transform it by taking logarithms and exponentiating:

$$p_t(x) = \exp\left[\sum_{i<j} \rho_{ij} x_{ij} x_{ji} + \sum_{i \neq j} \theta_{ij} x_{ij}\right] \prod_{i<j} n_{ij} \ , \tag{10.22}$$

where $\rho_{ij} = \ln\left[(m_{ij} n_{ij})/(a_{ij} a_{ji})\right]$ and $\theta_{ij} = \ln\left[a_{ij}/n_{ij}\right]$. The distribution p_t is in the exponential family: the η are all θ and ρ parameters, the statistics are the x_{ij} and the $x_{ij}x_{ji}$, the function $A(\eta)$ is $\sum_{i<j} \log n_{ij}$ and finally $h(x)$ is just the constant 1. The dimension is $2n^2$. Equation (10.22) is a reparameterization of p_t that is now expressed in terms of ρ_{ij} and θ_{ij}. The parameters ρ and θ are so-called *log-odds ratios*. Intuitively, $\exp(\rho_{ij})$ divides the symmetric cases by the asymmetric cases and therefore ρ_{ij} measures the tendency for reciprocation. The odds ratio $\exp(\theta_{ij})$ divides a case where there is an edge from i to j by a case where there is no edge. Therefore, θ_{ij} is an indicator of the probability of an edge from i to j.

To overcome the problem of too many parameters (that can be read off from the high dimension of p_t) we constrain the parameters in the following way:

$$\rho_{ij} = \rho \quad \forall i < j$$

and

$$\theta_{ij} = \theta + \alpha_i + \beta_j \quad \forall i \neq j \tag{10.23}$$

with $\sum_i \alpha_i = \sum_i \beta_i = 0$. The constraints imply that a global reciprocation parameter ρ is assumed and that the density from i to j is split up into three additive components: θ, a global density parameter, α_i, actor i's *expansiveness* (or *productivity*), and β_j, actor v_j's *attractiveness*. The resulting distribution is the p_1 distribution:

$$\begin{aligned} p_1(x) &= \exp\left[\rho m' + \theta \sum_{i,j} x_{ij} + \sum_i \alpha_i \sum_j x_{ij} + \sum_j \beta_j \sum_i x_{ij}\right] \cdot \prod_{i<j} n_{ij} \\ &= \exp\left[\rho m' + \theta e + \sum_i \alpha_i \Delta_{\text{out}}(i) + \sum_j \beta_j \Delta_{\text{in}}(j)\right] \cdot \prod_{i<j} n_{ij} \ . \end{aligned} \tag{10.24}$$

Also p_1 belongs to the exponential family: The statistics are the number of mutual edges m', the total number e of edges, and $\Delta_{\text{in}}(i)$ and $\Delta_{\text{out}}(i)$, i.e. for all i the in- and out-degrees of node i. The dimension is $2n + 2$, significantly lower than for p_t. Equation 10.24 shows that all statistics except for m' can be expressed as so-called *margins*, i.e. as a sum over the variables where some indices are fixed and others go over the complete range.

After having deduced the p_1 model, the most natural question is how we can estimate the parameters $\underline{\theta} = (\rho, \theta, \alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n)$ from an observed

graph. The standard estimation procedure for p_1 is maximum likelihood (ML-) estimation which yields the parameters that maximize the probability $p_1(x \mid \underline{\theta})$ for the observed x. The general approach to find the ML-estimator is to differentiate the probability density function for the parameters and to search for maxima. In this context the density function is called *likelihood function* $\ell_x(\underline{\theta})$ because it is seen as a function in the parameters and not in the data values. The theory of exponential families (that is beyond the scope of this book, see [385] for details) directly gives the result that the maximum likelihood estimation can be found as the solution of the *likelihood equations* in which the (sufficient) statistics are equated to their expected values.[4] In our case the sufficient statistics are all statistics that define p_1. Therefore we get

$$m' \stackrel{!}{=} E[m'] = \sum_{i<j} m_{ij} \ , \tag{10.25}$$

$$\Delta_{\text{in}}(i) \stackrel{!}{=} E[\Delta_{\text{in}}(i)] = \sum_{j} (m_{ij} + a_{ij}) \ \ \forall i \in \{1, \ldots, n\} \ , \tag{10.26}$$

$$\Delta_{\text{out}}(j) \stackrel{!}{=} E[\Delta_{\text{out}}(j)] = \sum_{i} (m_{ij} + a_{ij}) \ \ \forall j \in \{1, \ldots, n\} \ . \tag{10.27}$$

Note that for ease of presentation the variables θ and ρ_{ij} have been transformed back. Theoretically, any standard method that solves such a system of linear equations (like the Newton-method) can be applied. However, the structure of these equations can lead to nontrivial convergence problems. Therefore, specific algorithms have been developed; one of them is the *generalized iterative scaling* algorithm. In fact after a transformation of the variables also standard iterative scaling can be used. As this transformation is also needed in the next section it is presented here. Let

$$Y_{ijk\ell} = \begin{cases} 1 & \text{if } X_{ij} = k, X_{ji} = \ell \text{ for } k, \ell \in \{0, 1\}, \\ 0 & \text{otherwise.} \end{cases}$$

With this representation *all* statistics in the p_1 model can be expressed as margins of the variables. In particular, $m' = 1/2 \sum_{i,j} y_{ij11}$. For a single dyad we get

$$\Pr[Y_{ijk\ell} = 1] = \exp\left[k\alpha_i + k\beta_j + \ell\alpha_j + \ell\beta_i + (k + \ell)\theta + k\ell\rho + \lambda_{ij}\right] \ ,$$

where the λ_{ij} are chosen such that $\sum_{k,\ell} Y_{ijk\ell} = 1$ and $\sum_i \alpha_i = \sum_i \beta_i = 0$. It can be verified that this is equivalent to the p_1 model by expressing the original parameters m_{ij}, a_{ij} and n_{ij} in terms of the new parameters. Besides, a little calculation reveals that indeed

$$\prod_{i<j,k,\ell} \Pr[Y_{ijk\ell} = 1] = p_1(x) \ .$$

[4] Roughly, this result can be obtained by maximizing $\ell(\cdot)$ in setting $\partial\ell/\partial\underline{\theta}$ to zero and observing that p_1 is a convex function being in the exponential family.

The new representation allows to apply the theory of *generalized linear models* and *categorical data analysis*[5] to p_1.

The p_1 model incorporates the possibility to do goodness-of-fit tests and general hypothesis testing.

10.2.2 Goodness-of-Fit Indices

One of the major advantages of statistical models over the 'ad-hoc' deterministic models is the (at least theoretical) possibility to make precise statements on both how appropriate a model is for the observed data and how justified hypothesis on the social network are.

We review basic facts from statistics that are necessary to understand this. Whether we want to evaluate the goodness-of-fit of the model or whether we are interested in verifying claims about the social network, we are always in a similar setting in which we have two alternative hypothesis, the *null hypothesis* H_0 and the *alternative hypothesis* H_A. Already the names suggests that we usually treat these two hypothesis asymmetrically, which will become clear later in this section. To give an example H_0 might state that the observed social network is from a p_1 distribution with a given parameter set $\{\theta = \theta', \rho = \rho', \alpha_1 = \alpha'_1, \ldots, \alpha_n = \alpha'_n, \beta_1 = \beta'_1, \ldots, \beta_n = \beta'_n\}$, whereas H_A could state that this is true except for the reciprocation parameter, which is different: $\{\theta = \theta', \rho \neq \rho', \alpha_1 = \alpha'_1, \ldots, \alpha_n = \alpha'_n, \beta_1 = \beta'_1, \ldots, \beta_n = \beta'_n\}$ In this example H_0 is called a *simple hypothesis* because it completely specifies the distribution, whereas H_A does not specify ρ and is therefore called a *composite hypothesis*. In general composite hypotheses specify that the parameters can come from a subset of all possible parameters. A *test statistic* T is a random variable that maps the observed data x to a value $T(x)$, often with $T(x) \in [0, 1]$. The set of values of T for which H_0 is accepted (rejected) is denoted by *acceptance region* (resp. *rejection region*). Often the rejection region is of the form $\{x \mid T(x) < c\}$ or $\{x \mid T(x) > c\}$, then the value c that separates the rejection region from the acceptance region is called the *critical value*. In the ideal case all x for which H_0 holds are mapped to values in the acceptance region and all other x are mapped to values in the rejection region. In almost all nontrivial cases errors occur. These errors can be of two types:

1. H_0 is true, but the test rejects it. This is called a *type I error*, its probability α is called the *significance level* of the test.
2. H_0 is false, but is accepted. This (usually less detrimental) error is called a *type II error*. Let its probability be β, then we call $1 - \beta$ (the probability that H_0 is false and rejected by the test) the *power* of the test.

The asymmetry of H_0 and H_A is reflected in the usual test procedure: A significance level α is fixed (typically at small values like $0.01, 0.05$ or 0.1) and an appropriate test T is chosen. Obviously, tests with higher power for the fixed

[5] To be more precise the transformation shows that p_1 is a *loglinear model of homogeneous association* or of *no three-factor interaction*, see [3, 213].

significance level are preferable. The choice of the significance level reflects how detrimental the researcher assesses a type I error. The critical value c is set according to this significance level and finally $T(x)$ is computed on the observed data x. If $T(x) > c$ the null hypothesis is rejected, otherwise it is accepted. In order to set the critical value c according to the significance level we need to find a c for which $\Pr[T(x) > c] \leq \alpha$ under the assumption that the null hypothesis is true. Therefore, it is in general necessary to know the distribution of the test statistic under the null hypothesis. This distribution is the so-called *null distribution*.

Finding a good test, i.e. finding a test T with high or even maximum power among all possible tests, is a complicated problem beyond the scope of this book. However we present a paradigm from which many tests are constructed: Given two hypothesis H_0 and H_A expressed by the subsets of parameters ω_0 and ω_A to which they restrict the likelihood function $\ell(\theta)$, then the statistic

$$\Lambda^* = \frac{\sup_{\theta \in \omega_0} \ell_x(\theta)}{\sup_{\theta \in \omega_A} \ell_x(\theta)}$$

is called the *likelihood ratio test statistic*. High values of Λ^* suggest that H_0 should be accepted, low values suggest it should be rejected. The *likelihood ratio test* rejects for values below some critical value c and accepts above it. One reason why this test is often used is that it can be shown to be optimal for simple hypotheses (in this case the supremum is over a single value $\underline{\theta_0}$ resp. $\underline{\theta_A}$).

Lemma 10.2.2 (Neyman-Pearson). *Let H_0 and H_A be simple hypotheses given by the two parameter vectors $\underline{\theta_0}$ resp. $\underline{\theta_A}$. If the likelihood ratio test that rejects H_0 for $\frac{\ell_x(\underline{\theta_0})}{\ell_x(\underline{\theta_A})} < c$ and rejects it otherwise has significance level α then any other test statistic with significance level $\alpha' \leq \alpha$ has power less than or equal to that of the likelihood ratio test.*

Note that in the case of composite hypothesis nominator and denominator are the ML-estimates from the respective restricted parameter sets ω_0 and ω_A. For distributions involving exponentiation like the exponential family it is often easier to work with the ratio of the logarithms. In this case we get the statistic G^2 that is called *log likelihood ratio statistic*:

$$G^2 = -2 \log \lambda \ .$$

The factor of -2 has the reason that with this definition, G^2 has an approximate chi-square distribution in many cases.

Testing with p_1. We now investigate how to apply the general setting above to p_1 models. For goodness-of-fit evaluation we would state H_0 as "the data is generated by a p_1 model". Intuitively, H_A should express that "the data is generated by some other (more complicated) model". Making this statement precise is difficult, we need to define a family of distributions p_s that is a meaningful superset

of p_1 with two properties: First, for the likelihood ratio tests we need to be able to do ML-estimation in p_s. Second, we need to determine the null distribution of the likelihood ratio test statistic, in order to set a meaningful critical value. For p_1 both problems are nontrivial. One possibility to extend p_1 to a more general distribution is to allow for *differential reciprocity*, i.e. instead of setting $\rho_{ij} = \rho$ every actor gets a reciprocity parameter ρ_i and we set $\rho_{ij} = \rho + \rho_i + \rho_j$. Let us ignore the estimation problems for this model and assume that we can calculate the ML-estimates for given data. Then the likelihood ratio is the maximum likelihood of the p_1 model over the maximum likelihood of this extended model (which cannot be smaller because it contains the p_1 model). The value of this ratio indicates how justified the assumption of a global reciprocity parameter in the p_1 model is.

10.2.3 Blockmodels and p_1

The p_1-model has been extensively used for blockmodels. Recall that the p_1-model estimation yields—apart from the global density and reciprocation estimates θ and ρ—an expansiveness and an attractiveness estimate α_i respectively β_i for each actor.

One prominent approach is from Anderson, Faust, and Wasserman [30]. They propose to interpret the stochastic equivalence of two actors as them having the same α_i and β_i values . From this they derive the following blockmodeling procedure.

1. Fit a p_1-model to the observed digraph G, giving a set of parameters $\{\theta, \rho, \alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n\}$.
2. Attribute the point $q_i = (\alpha_i, \beta_i)$ to each actor $i \in \{1, \ldots, n\}$.
3. Cluster the points into k clusters and return the clusters as a partition $\mathcal{P}$ for the blockmodel.

Alternatively Anderson, Faust, and Wasserman suggest to take the points as a result of the blockmodel. The parameter k is an input parameter to the blockmodeling procedure. For the clustering any of the clustering methods from Section 10.1.3 or Chapter 8 can be used. Once the partition $\mathcal{P}$ has been found we can test its quality by the testing methods of the previous section: Let the null-hypothesis H_0 be that $\mathcal{P}$ is indeed the partition and therefore all actors in each $P_k \in \mathcal{P}$ are stochastically equivalent and have $\alpha_i = \alpha_j \, \forall i, j \in P_k$. For a maximum likelihood ratio test we need to evaluate G^2 which can be shown to be

$$2 \sum_{i<j,k,\ell} y_{ijk\ell} \log \frac{y_{ijk\ell}}{\hat{y}^{\mathcal{P}}_{ijk\ell}} \ ,$$

where $y_{ijk\ell}$ are the observed data and $\hat{y}^{\mathcal{P}}_{ijk\ell}$ are the ML-estimates for $\Pr[Y_{ijk\ell} = 1]$ under the side constraints given by $\mathcal{P}$. The null distribution of G^2 is a chi-squared distribution, the degrees of freedom of which are a function of the number of partitions.

The above model has some serious drawbacks that will become clear in a typical example, in which the social network consists of a class of pupils in a primary school. The pupils are asked for their best friends. The answers are encoded in the directed graph, i.e. an edge from pupil i to pupil j means that pupil i sees pupil j as one of his best friends. Typically, this setting results in a graph with two 'clusters', the boys and the girls. Both among the boys and among the girls a high 'within'-density of directed edges can be observed. Between the two clusters there are usually considerably less edges, corresponding to a low 'between'-density. From the discussion about the different types of equivalences it should be clear that the two groups should be taken into consideration for the blockmodel and that the partition should be into boys and girls. Unfortunately, the p_1-model attributes a single expansiveness and attractiveness parameter to each boy and girl and is thus unable to model the difference between 'within'- and 'between'-densities. This is a serious drawback because the different densities reflect the blockmodel. To overcome these shortcomings Wang and Wong proposed a refinement of the p_1-model [567]. In particular, if the partition $\mathcal{P}$ is already known in advance (like in the school class example) we can define indicator variables $d_{ijk\ell}$ that represent $\mathcal{P}$ as follows.

$$d_{ijk\ell} = \begin{cases} 1 & \text{if actor } i \text{ is in } P_k \text{ and actor } j \text{ is in } P_\ell, \\ 0 & \text{otherwise.} \end{cases}$$

Recall that in the derivation of p_1 we set θ_{ij} as in Eq. (10.23). We incorporate the knowledge about $\mathcal{P}$ in the new model by setting

$$\theta_{ij} = \theta + \alpha_i + \beta_j + \sum_{k,\ell} d_{ijk\ell}\lambda_{k\ell} \quad \forall i \neq j \ .$$

Here, λ_{ij} are the newly introduced *block parameters* that model the deviations from the average in the expansiveness and attractiveness between two specific partitions P_k and P_ℓ. In the school class example we would get a negative λ between the boys and girls and positive λs within the two partitions. Maximum likelihood estimation in this model can again be done via generalized iterative scaling, a transformation into a loglinear model of homogeneous association like for p_1 is not known however. For reasons of parsimony it is often preferably to restrict subsets of the λ_{ij}s to be equal.

10.2.4 Posterior Blockmodel Estimation

In the model of Wang and Wong that we saw in the previous section we had to content ourselves with blockmodels that needed the partition of actors as input and served only as a means of testing hypotheses on this partition. Such an approach is called *a priori* blockmodeling, because the partition constitutes a priori knowledge. It is justified whenever it is clear from the nature of the sociological question or by attributes of the actors (gender, age etc.) what the partition of interest is. In this section we consider a stochastic approach by Nowicki and Snijders to *a posteriori* blockmodeling, where the partition is unknown

[453]. This model does not have the drawbacks of other a posteriori approach we saw by Anderson, Faust, and Wasserman.

As in the p_1-model Nowicki and Snijders consider dyads, i.e., ordered pairs of actors (vertices) between which relations are given. Here a relation can be the presence or absence of directed edges between two actors, but more generally, the model allows the relation from vertex v_i to vertex v_j to take on any value from a finite set A, which is similar to allowing multiple edge sets as in Definition 10.0.1. Therefore, dyads (v_i, v_j) can take values x_{ij} in a set $\mathcal{A} \subset A \times A$, the values of all dyads together form the *generalized adjacency matrix*. For ease of presentation we will continue with directed graphs, thus we assume $\mathcal{A} = \{0,1\}^2$; for example $x_{ij} = (0,1)$ stands for the asymmetric dyad (v_i, v_j) with an edge from v_j to v_i. The crucial concept that models the partition is that vertices v_i have *colors* c_i from a set $\chi = \{1, \ldots, L\}$ that are not observed (latent). The authors call this model a *colored relational structure*. It is given by a generalized adjacency matrix x and a coloring $c = (c_1, \ldots, c_n)$.

The stochastic model that generates the data is now defined as follows. We model the coloring by independent identically distributed (i. i. d.) random variables C_i for each vertex $v_i \in V$. Thus we set

$$\Pr[C_i = k] = \theta_k$$

for each color $k \in \chi$. The joint distribution of a given coloring $c = \{c_1, \ldots, c_n\}$ is

$$\Pr[C_1 = c_1, \ldots, C_n = c_n] = \prod_{1 \le i \le n} \theta_{c_i} \ .$$

As we have seen in the discussion on different types of equivalences, we assume in blockmodeling that all actors in one block behave similarly. Therefore, it is assumed here that the type of relation between two actors i and j depends only on their colors:

$$\Pr[X_{ij} = a \mid C = c] = \eta(i, j, a) \ ,$$

where the η parameterize the distribution (we have to require $\forall i, j \in C$: $\sum_{a \in \mathcal{A}} \eta(i, j, a) = 1$). We obtain the following conditional distribution of relations x and colors c given the parameters:

$$\Pr[x, c \mid \theta, \eta] = \prod_{i=1}^{n} \theta_{c_i} \cdot \prod_{a,b} \left(\prod_{1 \le i < j \le L} \eta(i,j,(a,b))^{|\{x_{k\ell} | x_{k\ell}=(a,b), c_k=i, c_\ell=j\}|} \right) \cdot \prod_{a=b} \left(\prod_{1 \le i \le L} \eta(i,i,(a,b))^{|\{x_{k\ell} | x_{k\ell}=(a,b), c_k=c_\ell=i\}|/2} \right) \cdot \prod_{a \neq b} \left((\prod_{1 \le i \le L} \eta(i,i,(a,b))^{|\{x_{k\ell} | x_{k\ell}=(a,b), c_k=c_\ell=i, k<\ell\}|} \right) \quad \forall a, b \in \{0,1\} \ . \tag{10.28}$$

Note that this formula basically multiplies for each vertex the probability of its color θ_i and between all color classes the probabilities for the observed relations between the two classes. The first double product does the multiplication for different color classes, the last two double products do this for all monochromatic dyads. From a statistical point of view such a model falls into the class of *mixture models*.

We will briefly describe one way of statistical inference for such models. Assume some black box allows us to get a sample of values (θ, η, x) from the distribution given by the density function $f(\theta, \eta, c \mid x)$. Thus we have at our disposal a set of triplets $\{(\theta^0, \eta^0, x^0), (\theta^1, \eta^1, x^1), \ldots, (\theta^{K-1}, \eta^{K-1}, x^{K-1})\}$. This sample provides us with information about the underlying model parameters and hidden data. For example, the value

$$\frac{1}{K} \sum_{k=0}^{K-1} [x_i^k = x_j^k]$$

indicates how likely it is that actors v_i and v_j are in the same color class. Indeed, if the sample consists of *independent* draws from the distribution given by $f(\theta, \eta, c \mid x)$ it follows directly from the law of large numbers that the above value is a meaningful approximation of the random indicator variable $[C_i = C_j]$.

$$\Pr[C_i = C_j] = E\big[[C_i = C_j]\big] \approx \frac{1}{K} \sum_{k=0}^{K-1} [c_i^k = c_j^k] \tag{10.29}$$

For convenience we restate the (weak)[6] law of large numbers, which also makes precise the sense in which the $\approx$ symbol is to be understood.

[6] The type of convergence shown here is called *convergence in probability*. Other versions of this theorem exist in which stronger types of convergence are shown.

Theorem 10.2.3 (Weak Law of Large Numbers). *Let $X_1, X_2, \ldots, X_n$ be a sequence of independent random variables with $E[X_i] = \mu$ and $\mathrm{Var}[X_i] = \sigma^2$. Then for any $\varepsilon > 0$,*

$$\Pr\left[\left|\mu - \frac{1}{n}\sum_{i=1}^{n} X_i\right| > \varepsilon\right] \to 0$$

for $n \to \infty$.

The proof follows straight forward by one application of the Chebyshev inequality and can be found in any textbook on probability theory, for example [492].

This theorem makes no statement on the speed of convergence. In the case of independent random variables, the central limit theorem makes such a statement. Unfortunately we will see that our black box does not give us independent samples. With the same approach as above estimates for θ and η can be obtained. The value

$$\frac{1}{K}\sum_{k=0}^{K-1} \theta_{c_i}^k \tag{10.30}$$

gives an estimate of the probability of the color class containing actor i. Finally,

$$\frac{1}{K}\sum_{k=0}^{K-1} \eta(c_i, c_j, a) \tag{10.31}$$

is an estimate of the probability that between actor v_i's color class and actor v_j's color class a relation of type a holds.

These slightly awkward constructions are necessary, because there is an identifiability problem in the estimation process: It is not meaningful to talk about color class i because arbitrary permutations of the color class labels can lead to identical results. Similarly, it is not meaningful to estimate the probability $\Pr[C_i = j]$ (instead of 10.29). All functions in (10.29), (10.30), and (10.31) are invariant under permutations and therefore circumvent the identifiability problems.

Up to now we have gently ignored the question how we get the sample from $f(\theta, \eta, c \mid x)$. To this end a method called *Gibbs sampling* can be used. While the method itself is easy to describe its precise mathematical foundations are beyond the scope of this book. The general approach for Gibbs sampling from a distribution $f(x_1, \ldots, x_d)$ with prior distributions $\pi(x_i)$ for all variables is to start with a random point $(x_1^0, x_2^0, \ldots, x_d^0)$ as the first point of the sample. The next point is identical to the first, except in the first coordinate. The first coordinate is drawn from the full conditional distribution $f(x_1 \mid x_2 = x_2^0, \ldots, x_d = x_d^0)$. Usually it is much easier to get a sample from such a full conditional distribution then from the general one. In the ith step, the new point is the same as the last, except for the ($i \mod d$)th coordinate, which is drawn from the distribution $f(x_{i \bmod d} \mid x_1, \ldots, x_{(i \bmod d)-1}, x_{(i \bmod d)+1}, x_d)$. Often only every dth point is taken, so that the next point potentially differs in all coordinates from

the present one. In our case the Gibbs sampler works as follows: Given values (x^t, θ^t, η^t) the next values are determined as

1. $(\theta^{t+1}, \eta^{t+1})$ is drawn from $f(\theta, \eta \mid x^t, y)$.
2. For each value $i \in \{1, \ldots, n\}$ the color x_i^{t+1} is drawn from

$$f(x_i \mid \theta^t, \eta^t, x_1^t, \ldots, x_{i-1}^t, x_{i+1}^t, \ldots, x_d^t) \ .$$

It can be verified that the full conditional distributions used here have a comparatively easy form. The Gibbs sampler has the property that the sample $\{(x^0, \theta^0, \eta^0), \ldots, (x^{K-1}, \theta^{K-1}, \eta^{K-1})\}$ approximates the distribution $f(\theta, \eta, c \mid x)$ for large K. It is obvious from this description that the sample points are highly dependent, because the values $(x^{t+1}, \theta^{t+1}, \eta^{t+1})$ are constructed from (x^t, θ^t, η^t). The sequence of samples forms a Markov chain. Fortunately, the general theory of Markov chains comprises the so-called *ergodic theorem* that is in a sense the counterpart of the law of large numbers for dependent samples that are produced by a Markov chain. For a precise statement of the theorem too much terminology for Markov chains is required, therefore we leave the presentation at that intuitive level and refer the interested reader to the bibliography.

To sum up, Nowicki and Snijders propose to see blockmodeling as actors getting colors from a distribution defined by the θ parameters. The probabilities of relations between actors are influenced by their colors. As the colors are latent variables, the task is to predict them from the observations (namely the relations between the actors) and to estimate the parameters η that govern how the actors in the color classes relate to each other. The prediction and estimation is done by Gibbs sampling from the conditional distribution $f(\theta, \eta, c \mid x)$ and then evaluating invariant functions on the sample from which information about the coloring and the parameters can be inferred.

10.2.5 p^* Models

In Sects. 10.2.1 and 10.2.3, we have seen how a stochastic model of social network generation can be used to evaluate an a priori blockmodel and to compute an a posteriori blockmodel. The simple structure of the node-wise parameters α and β allows to define stochastic equivalence and, therefore, to express a blockmodel in terms of a restricted parameter space of the p_1 model. Moreover, the parameters of such simple models can be estimated exactly and efficiently.

On the other hand, we made quite strong assumptions on the network generation process. Even the basic assumption that the dyads are drawn independently of each other has been heavily criticized since the p_1 model was proposed in social network analysis. In 1996, Wasserman and Pattison [570] introduced a more powerful family of random graph distributions called p^* *models*, whose applications to blockmodeling will be discussed in the following. The aim of p^* models is to have greater flexibility in expressing the dependencies among the relations X_{ij} between the actors. To state this more formally we make the following definition.

Definition 10.2.4 (Conditional Independence). *Let X, Y and Z be (sets of) random variables. We say that X is* conditional independent *of Y given Z if*

$$\Pr[X \mid Y, Z] = \Pr[X \mid Z] \text{ for } \Pr[Z] > 0 \ .$$

This is written as $X \amalg Y \mid Z$.

In the p_1 model we have $\{X_{ij}\} \amalg X \setminus \{X_{ij}\} \mid \{X_{ji}\}$, i.e. the edge from i to j only depends on the edge from j to i. In order to model more complicated dependencies than this, we introduce a graph that represents these dependencies.

Definition 10.2.5. *Let $W := \{X_{ij} \mid 1 \leq i, j \leq n\}$ with $X_{ij} \in \{0, 1\}$ be the set of random variables of the edges of a random graph X on n nodes. Thus as before X is the random variable that assumes values in $\mathcal{G}_n$.*

- *A distribution on X is called* random field *if all graphs $x \in \mathcal{G}_n$ get positive probability.*
- *The undirected graph $\mathcal{I}_X = (W, F)$, $F \subseteq W^2$, is called the* dependency graph *of X if for all $X_{ij} \in W$ it holds that*

$$\{X_{ij}\} \amalg W \setminus \{X_{ij}\} \mid \mathcal{N}(X_{ij})$$

  where $\mathcal{N}(X_{ij})$ are all random variables adjacent to X_{ij} in $\mathcal{I}_X$.
- *A random field that can be expressed via a dependency graph is called a* Markov field.

The p_1 model can be seen as a Markov field (or *Markov graph*) with a dependency graph that consists of all edges $\{X_{ij}, X_{ji}\}$. The idea of p^* is to try to find explicit distributions for arbitrary dependency graphs. The Hammersley-Clifford Theorem [59, 589] states that this is always possible in the sense that for each Markov field there is a distribution that can be expressed by an (almost) closed form. We state the theorem in a simplified version:

Theorem 10.2.6 (Hammersley-Clifford). *Let $\mathcal{I}_X = (W, F)$ be the dependency graph of a Markov graph X. Let $\mathcal{C}$ be the set of cliques of $\mathcal{I}_X$. Then, there exist* potentials $\{\lambda_c \in \mathbb{R} \mid c \in \mathcal{C}\}$ *such that*

$$\Pr[X = x] = \frac{\exp\left(\sum_{c \in \mathcal{C}} \lambda_c \cdot \prod_{X_{ij} \in c} x_{ij}\right)}{\kappa} \ ,$$

where $\kappa := \sum_{z \in \mathcal{G}_n} \exp\left(\sum_{c \in \mathcal{C}} \lambda_c \cdot \prod_{X_{ij} \in c} x_{ij}\right)$ is a normalization constant.

Observe that the products over the cliques are one if and only if all edges in the clique in the dependency graph are present, otherwise they equal zero. Given a dependency graph $\mathcal{I}_X$ that expresses our assumptions about independence between relations in the observed graph x, we get a minimal parameter set for the graph distribution consisting of the potentials λ_c of all cliques of $\mathcal{I}_X$. Distributions of this kind are called p^* *models*.

Estimating the Potentials. Estimation in p^* models has been a topic of vivid research discussions in the last years. Several estimation methods have been proposed, the most prominent ones being the *pseudolikelihood method* and the *Markov Chain Monte Carlo (MCMC) method* which we saw briefly already in Section 10.2.4. Both methods are mathematically involved and have serious drawbacks as discussed in [529] and references therein, therefore we will not present them here. See the bibliography for detailed references on the methods.

Using p^* Models for Blockmodeling. Up to now, we have only seen a stochastic model for graph generation. Due to the clique-wise potentials, there is no obvious counterpart to the stochastic equivalence in p_1 blockmodels. We present an approach proposed in [472].

Consider an a priori blockmodel with actor set partition $\mathcal{P} := \{P_1, \ldots, P_L\}$ for an observed graph x. Let $\mathcal{C}$ be the set of cliques of the dependency graph $\mathcal{I}_X = \{W, F\}$. We call the subgraph of x on the nodes of a clique $c \in \mathcal{C}$ the *configuration* $C(c, x)$.

Definition 10.2.7. *Two configurations $C(a, x)$ and $C(b, x)$, $a, b \in \mathcal{C}$, are called* isomorphic *if there is a bijective map ϕ: $a \to b$ satisfying*

$$\phi(X_{ij}) = X_{i'j'} \Leftrightarrow (x_{ij} = x_{i'j'}) \wedge (x_{ji} = x_{j'i'}) \\ \wedge (P(v_i) = P(v_{i'})) \wedge (P(v_j) = P(v_{j'})) \qquad \forall X_{ij} \in W \ .$$

We can incorporate the blockmodel into the parameter estimation by forcing $\lambda_a = \lambda_b$ for isomorphic configurations $C(a, x)$ and $C(b, x)$. Then, the plausibility of a blockmodel can be scored by computing the likelihood ratio statistic G^2 using ML-estimates for both the unrestricted and restricted parameter spaces (see Sects. 10.2.1 and 10.2.3).

10.3 Chapter Notes

We have seen that the tight concepts of roles and equivalences discussed in Chapter 9 are not suited to analyze real-world data occurring in psychology and sociology. Therefore, a variety of methods has been developed since the 70's that realize some kind of relaxation of the strict structural equivalence.

Traditional methods in blockmodeling are mainly based on measures of similarity of actor relationships, which are then used to compute the partition of actors into positions. These measures can be turned into metrics using techniques for multidimensional scaling in order to refine the relational data or, alternatively, to enable a visual interpretation. Often, clustering based methods are used to compute the actor set partition. We have seen also the popular but heavily criticized CONCOR algorithm, which works with iterated correlations of adjacency matrices. Afterwards, different criterions may be used to decide on relations between the positions and, therefore, to obtain a simplified representation of the original data. With generalized blockmodeling, an integrated

optimizational approach has been presented which solves both the partitioning problem and the image matrix computation by minimizing a common error function.

Second, stochastic models have been introduced which assume certain kinds of stochastic generation processes for the observed relational data. They represent the more recent developments in blockmodeling. Both simple models offering exact and efficient estimation methods and more complex, realistic models have been presented. For the latter, different approaches to parameter estimation have been discussed which do not offer both exactness and efficiency, but which have been successfully applied to social network data. We have seen that the adaptation and application to blockmodeling follows the introduction of a new stochastic model originally proposed as general explanation for observed data.

Finally, we conclude that the area of blockmodeling seems to be strongly application driven. Researchers from psychology and sociology are in need of methods to analyze the positional structure of observed networks and serve themselves from different scientific areas like computer science and statistics to obtain methods giving them the desired analytic results. Hence, the approaches and techniques are quite heterogenous. At the moment, most researchers in this area seem to use rather the traditional methods discussed in Section 10.1 than the more recent methods of Section 10.2.

Further information on the properties of the correlation coefficient and its relationship to the Euclidean distance can be found in [492, 528].

Kruskal's Multidimensional Scaling algorithm was published in two seminal articles as early as 1964 [372, 373]. Cox and Cox discuss Multidimensional Scaling in a recent book [134] from a statistical point of view, it also contains among other topics a presentation of Kruskal's algorithm and its relation to other MDS-methods. The approximation algorithm for metric embedding is by Bădoiu [105]. The part of the algorithm that finds the x-coordinates is in the appendix of the paper and can be found at `http://theory.lcs.mit.edu/~mihai/`. More on the related metric embedding problems can be found in [320] and the references therein.

A variety of applications of the CONCOR algorithm to social network data can be found in [31, 98, 100, 230, 364, 424, 434].

An implementation of generalized blockmodeling is included in the Pajek software available via `http://vlado.fmf.uni-lj.si/pub/networks/pajek/default.htm`.

Exponential models are discussed in [385]. The p_1 model was introduced by Holland and Leinhardt in [302]. The goodness-of-fit test against differential reciprocity is advocated by Fienberg and Wasserman in [213]. In this paper the authors also show how to understand p_1 as a special case of a so-called general linear model. Loglinear models of homogeneous association can be found in the textbook by Agresti [3]. The Neyman-Pearson Lemma is proved in [384]. A gentle introduction into testing and statistics in general can be found in the book by Rice [492].

First applications of p_1 to blockmodeling can be found in [30, 212, 301]. The refinement of the p_1-model presented here is by Wang and Wong [567]. The a posteriori blockmodeling approach is by Nowicki and Snijders [453, 530]. In these papers, the identifiability problems are discussed in more detail. Furthermore, methods to test the adequacy of the obtained class structure are handled therein.

A proof of the strong law of large numbers can be found in [492]. The ergodic theorem is discussed in [245]. More information on the related Markov Chain Monte Carlo methods, Gibbs sampling, and mixture models can be found in [243, 245].

The p^* models can be seen as an application of Markov random graphs to social sciences. Markov random graphs were introduced by Frank and Strauss [222]. They were made popular in social network analysis by a sequence of papers by Pattison, Robins, and Wasserman [472, 494, 570]. Recently, Snijders has analyzed them in detail and pointed out estimation problems together with categorical problems which call into question the appropriateness of p^* to many social network problems [529]. More information on the two estimation methods for p^* can be found in [245, 529, 589]. Finally, [29, 200] contain more social network analyses using p^* models.