arXiv:1803.10354v3 [math.CO] 28 Feb 2019

AN OPTIMIZATION PARAMETER FOR SERIATION OF NOISY DATA[*]

MAHYA GHANDEHARI[†] AND JEANNETTE JANSSEN[‡]

Abstract. A square symmetric matrix is a Robinson similarity matrix if entries in its rows and columns are non-decreasing when moving towards the diagonal. A Robinson similarity matrix can be viewed as the affinity matrix between objects arranged in linear order, where objects closer together have higher affinity. We define a new parameter, Γ_1, which measures how badly a given matrix fails to be Robinson similarity. Namely, a matrix is Robinson similarity precisely when its Γ_1 attains zero, and a matrix with small Γ_1 is close (in the normalized ℓ^1-norm) to a Robinson similarity matrix. Moreover, both Γ_1 and the Robinson similarity approximation can be computed in polynomial time. Thus, our parameter recognizes Robinson similarity matrices which are perturbed by noise, and can therefore be a useful tool in the problem of seriation of noisy data.

Key words. Robinson similarity matrices, Robinsonian matrices, unit interval graphs, seriation, linear embeddings of graphs

AMS subject classifications. 68R01, 05C85, 05C50

1. Introduction.

Many real-life networks, such as online social networks, biological networks and neural networks, are manifestations of an underlying (hidden) spatial reality. For example, members of a social network can be identified with nodes placed in a metric space, whose coordinates represent the interests, backgrounds, and other significant features of the users. The formation of the network is then modeled as a stochastic process, where the probability of a link occurring between two nodes decreases as their metric distance increases. A fundamental and challenging problem in the analysis of a social network (or any other spatial network) is to uncover its "hidden spatial layout", *i.e.* to identify the metric space representation of the network. Analysis and visualization of data becomes considerably more tractable when the dataset is presented according to its spatial reality.

The classical seriation problem, introduced by Robinson in [22], can be viewed as the special case of the spatial layout problem, restricted to one dimension. The objective of the seriation problem is to order a set of items so that similar items are placed closer to each other. The seriation question translates in a natural way into a question regarding symmetric matrices. A symmetric matrix is a *Robinson similarity* matrix, or *Robinson matrix* for short, if its entries are non-decreasing when moving towards the main diagonal in each row or column. A symmetric matrix A is said to be *Robinsonian*, if it becomes a Robinson matrix after simultaneous application of a permutation π to its rows and columns. In that case, the permutation π is called a *Robinson ordering* of A. If the entries of the symmetric matrix $A = [A_{i,j}]$ represent similarity of items i and j, then the Robinson ordering represents a linear arrangement of the items so that similar items are placed closer together.

The problem of recognizing Robinsonian matrices, and finding their Robinson orderings, can be solved in polynomial time. See [19] for the first polynomial time algorithm for this problem, and [24, 20, 15, 14] for more recent efficient algorithms. Most of these algorithms are based on a similar principle; namely the connection

[*]Submitted to the editors DATE.

Funding: The authors gratefully acknowledge supports from UDRF and NSERC.

[†]Department of Mathematical Sciences, University of Delaware, Newark, DE (mahya@udel.edu).

[‡]Department of Mathematics and Statistics, Dalhousie University, Halifax, Canada (Jeannette.Janssen@dal.ca).

between Robinsonian similarity matrices and unit interval graphs ([15, 14]) or interval (hyper) graphs ([19, 24, 20]). A spectral algorithm based on reordering the matrix according to the components of the second eigenvector of the Laplacian, or the Fiedler vector, was given in [1], and was then applied to the *ranking problem* in [11].

The seriation problem has diverse and significant applications, from its origin in archeological studies to recent applications to ecology and sociology. For a historical overview of the problem and its diverse applications, see [16]. In most of these applications, it is natural to expect the data to be noisy. In that case, the optimal reordering of a data-derived matrix may not be itself a Robinson matrix, but it will be close to one. The question then becomes, to which extent a given matrix resembles a Robinson matrix. All of the algorithms mentioned in the previous paragraph only apply to noise-free Robinsonian similarity matrices.

In the presence of noise, the goal of the seriation problem is to find an "almost Robinson" ordering of a given matrix, *i.e.* an ordering for which the reordered matrix is closest to being Robinson. This question turned out to be much more challenging than the error-free analogue. In fact, it is shown in [4] that the problem of finding a reordering and a Robinson matrix which is the best ℓ^∞-approximation is NP-hard. In [5] a factor 16 approximation algorithm is given for the case of ℓ^∞. NP-hardness for a number of related problems is established in [2], where approximation by ℓ^p distance is considered. Specifically, it is shown that for an integer p, the problem of finding *proper strong Robinson relations* within specified ℓ^p distances of a given matrix is NP-complete. A proper strong Robinson relation corresponds to an appropriate relabelling of the original matrix together with a Robinson matrix with certain additional, stronger properties. In [10] (together with the references therein) a statistical approach to the problem of seriation with noise is developed, where the error of Robinson approximation is measured by the Frobenius norm.

If the appropriate labelling is given, then the problem becomes more tractable. Now the problem is that of finding a Robinson matrix closest to a given matrix. For the ℓ^∞-norm, it is known that this problem can be solved in polynomial time since an explicit closed form for the optimal solution can be easily given (cf. [23]). The problem of finding the best ℓ^1-approximation can be formulated as a linear program: Minimize the linear function $\|A-R\|_1$ subject to the constraint that R is Robinson similarity. The constraint can be expressed with $O(n^3)$ inequalities, and thus the problem can be solved in polynomial time.

In this article, we develop new methods and algorithms which can be used for seriation of noisy data. Our focus here is on formalizing the notion of a matrix being "almost Robinson." To do so, we introduce a parameter, which we call Γ_1, that measures how much the local structure of a matrix resembles being a Robinson similarity. Namely, Γ_1 sums the magnitude of local violations to the Robinson similarity property, and achieves the value 0 precisely when the matrix is Robinson. This parameter is a natural tool for the seriation problem, and has been used in practice as a heuristic to measure the amount of deviation from a Robinson form (see [3, 13]). Moreover, Γ_1 is simple to formulate, and can be computed in linear time. The main goal of this paper is to show that if the number and magnitude of local violations is small, then the matrix is indeed close, in the sense of ℓ^1-norm, to a Robinson matrix. Precisely, we prove in Theorem 2.2 that for every given matrix A, there exists a Robinson matrix R so that $\|A-R\|_1 \leq 26\Gamma_1(A)^{1/3}$. In addition, we give a polynomial time algorithm to compute the Robinson approximation R which fulfills the above inequality.

A proposed application of this work is a novel scheme for treating the seriation problem of noisy data, in which the selection of the best permutation is guided by parameter Γ_1. The traditional formulation is: *Given a matrix A, find a permutation π and a Robinson matrix R so that $\|A^\pi - R\|_p$ is minimized.* (Here A^π refers to the matrix obtained by permuting rows and columns of A according to π.) This approach requires the simultaneous optimization of both the matrix R and the permutation π. Using the results in this paper, we can instead reduce the noisy seriation problem to the following problem:

Given a symmetric matrix A, find a permutation of its rows and columns so that $\Gamma_1(A)$ is minimized.

Once such a permutation is found, our algorithm can be used to compute the appropriate Robinson approximation. While this approach is not an approximation algorithm per se, we do bound the performance of this approach in terms of the optimal outcome. Namely, as will be shown in Lemma 2.3, the best possible Robinson approximation has normalized ℓ_1 distance at least $\frac{1}{4}\Gamma_1(A)$ from A. Therefore, our results implicitly bound the Robinson approximation achieved by our algorithm in terms of the optimal solution.

The results of this article are fundamentally different from any previous results on ℓ^∞-fitting Robinsonian structures (see for example [4]). Indeed, when matrices grow large in size, the ℓ^1-norm provides us with a more suitable notion of "closeness". This fact becomes apparent when we analyze a growing sequence of graphs which are convergent in the sense of Lovász-Szegedy [18]. Indeed, this article (and the choice of notation for the parameter Γ_1) was motivated by our previous work [6], where we introduce a parameter Γ which characterizes Robinson *graphons*. Graphons are symmetric functions on $[0,1]^2$ with values in $[0,1]$, which can be thought of as the "blueprint" of a random graph whose vertices are randomly sampled from the interval $[0,1]$. A matrix $A = [A_{i,j}]$ can be interpreted as a graphon in a natural way, by splitting the unit square into subsquares of size $\frac{1}{n} \times \frac{1}{n}$, and setting the graphon equal to $A_{i,j}$ everywhere in the (i,j)-th subsquare. A *Robinson graphon* is a graphon which is non-decreasing along every horizontal or vertical line towards the main diagonal.

Our main result in [6] is that Γ becomes a continuous parameter, when the space of graphons is equipped with the box-norm. Therefore Γ provides us with a parameter to measure Robinson resemblance, which can be efficiently approximated. The parameters Γ_1 and Γ are closely related, even though box-norm continuity does not hold for Γ_1 anymore. In future work [12], we employ these parameters simultaneously in order to develop (continuous) methods for seriation of noisy data.

Finally, we mention applications to graphs and networks. Binary Robinson matrices correspond to unit interval graphs. More precisely, a graph is a unit interval graph if and only if the adjacency matrix is Robinsonian, that is, there exists a labelling of the vertices so that the resulting adjacency matrix is a Robinson matrix. The parameter Γ_1 can be directly applied to a (labelled) graph, and if Γ_1 is sufficiently small, our algorithm constructs a unit interval graph that is close, in edit distance, to the original graph.

When applied to real-life networks, the parameter Γ_1 can be used to measure how closely the matrix conforms to a linear model. In previous work, the authors investigated this question in the context of graph limits [6, 7]. In a *linear graph model* the vertices of the graph are placed on a line, and the links are formed stochastically so that vertices that are closer together are more likely to connect. The linear layout can refer to a time line, in case of graphs derived from archeology, or the food chain, in case of food webs. But a linear layout may also point to the presence of a strong hidden variable that influences link formation, such as hierarchy, in a professional social network, or age in a friendship graph. If a graph conforms to a linear model, then we expect the adjacency matrix of the graph to be "almost-Robinson." Parameter Γ_1 may therefore serve as a measure of the "linearity" of a given network.

The rest of this article is organized as follows. In Section 2, we introduce the necessary notations and definitions, and we state our main result, namely Theorem 2.2. In Section 3, we present Algorithm 3.1 which finds a Robinson approximation for the special case where A is a binary matrix (*i.e.* a graph adjacency matrix). The values of every cell in the Robinson approximation is decided based on the entries of A in the upper right and the lower left regions defined by that cell (see Figure 2). Algorithm 3.1 is very simple to state, however one needs to use careful approximations and counting tricks to prove that the algorithm generates an output which indeed is a good ℓ^1-approximation for the input matrix. Section 4 provides us with an adaptation of Algorithm 3.1 to general matrices. This is indeed a natural generalization, as every matrix with entries in $[0, 1]$ decomposes into a convex combination of binary matrices. Fortunately, the parameter Γ_1 distributes over such decompositions, even though it is not a linear parameter in general. This allows us to apply Algorithm 3.1 to the components of the decomposition in stages. In Section 5, we develop an algorithm that represents a preprocessing step for Algorithm 3.1. This preprocessing step is designed to transform the input matrix A, so that Algorithm 3.1 generates a better-approximating output matrix R. The trade-off here is that the preprocessing step increases the complexity of the algorithm, which still remains polynomial. We finish the paper by some concluding remarks and future directions.

2. Definitions and main results. For a given positive integer n, let $\mathcal{A}_n$ denote the set of all symmetric $n \times n$ matrices with entries in $[0, 1]$. Note that the restriction on the range of the entries is not a limitation, since it can always be achieved by shifting and scaling of the matrix. A matrix $A \in \mathcal{A}_n$ is called *binary* if it has entries only from $\{0, 1\}$. We refer to the position in the i-th row and j-th column of A as the (i, j)'th *cell*. For a matrix A of size n, we define its (normalized) ℓ^1-norm to be $\|A\|_1 = \frac{1}{n^2}\sum_{i,j=1}^{n} |A_{i,j}|$.

DEFINITION 2.1. *An $n \times n$ symmetric matrix A is a* Robinson matrix *if, for all $1 \le i < j < k \le n$,*

$$A_{i,j} \ge A_{i,k} \text{ and } A_{j,k} \ge A_{i,k}. \tag{2.1}$$

In this section, we define a parameter, denoted by Γ_1, which measures how badly a matrix fails to be Robinson. The choice of notation for Γ_1 is due to the fact that it simply adds the magnitude of violations to (2.1). Precisely, given a symmetric matrix A of size n,

$$\Gamma_1(A) = \frac{1}{n^3} \sum_{1 \le i < k < j \le n} [A_{i,j} - A_{i,k}]_+ + [A_{i,j} - A_{k,j}]_+,$$

where $[x]_+ = x$ if $x \ge 0$, and 0 otherwise. It is clear that $\Gamma_1(A) = 0$ if and only if A is a Robinson matrix. Note also that, for a binary matrix A which is not Robinson, $\Gamma_1(A) \ge \frac{1}{n^3}$. Moreover, the computation of $\Gamma_1(A)$ for an $n \times n$ matrix A involves a simple summation which can be executed in $O(n^3)$ steps. Finally note that, due to the normalization factor $\frac{1}{n^3}$, $\Gamma_1(A) \in [0, 1)$ whenever $A \in \mathcal{A}_n$.

We are now ready to state our main result (Theorem 2.2), whose proof takes a large part of the paper and finishes only at the end of Section 5.

THEOREM 2.2. *For every $A \in \mathcal{A}_n$, there exists a Robinson matrix $R \in \mathcal{A}_n$ so that*

$$\|A - R\|_1 \le 26\,\Gamma_1(A)^{1/3}.$$

Moreover, R can be computed in polynomial time. In addition, if A is binary, then there exists a binary matrix R satisfying the conditions of the theorem.

The distance between A and the Robinson approximation obtained in Theorem 2.2 is bounded in terms of $\Gamma_1(A)$, and thus the algorithm does not necessarily give the best possible approximation. However, as stated in the following simple lemma, there is a close relationship between $\Gamma_1(A)$ and the distance between A and the best possible Robinson approximation.

LEMMA 2.3. *For every pair of matrices A and R in $\mathcal{A}_n$, if R is a Robinson matrix then*

$$\|A - R\|_1 \ge \frac{1}{4}\Gamma_1(A).$$

Consequently, we have

$$\min\{\|A - R\|_1 : R \in \mathcal{A}_n \text{ is Robinson similarity}\} \ge \frac{1}{4}\Gamma_1(A).$$

Proof. Since the function $[\,\cdot\,]_+$ is sub-additive, we observe that

$$\Gamma_1(A) = \Gamma_1(A - R + R) \le \Gamma_1(A - R) + \Gamma_1(R) = \Gamma_1(A - R),$$

as R is a Robinson matrix and thus $\Gamma_1(R) = 0$. Moreover, for every symmetric matrix B of size n, we have

$$\begin{aligned}
\Gamma_1(B) &= \frac{1}{n^3} \sum_{1\le i<k<j\le n} [B_{i,j} - B_{i,k}]_+ + [B_{i,j} - B_{k,j}]_+ \\
&\le \frac{1}{n^3} \sum_{1\le i<k<j\le n} |B_{i,j}| + |B_{i,k}| + |B_{i,j}| + |B_{k,j}| \\
&\le \frac{4}{n^3} \sum_{k=1}^{n} \sum_{1\le i,j\le n} |B_{i,j}| \le 4\|B\|_1.
\end{aligned}$$

Letting $B = A - R$ finishes the proof. ∎

Applying Theorem 2.2 to binary matrices, we obtain an interesting corollary for graphs. For a graph G, let the *augmented adjacency matrix* B_G denote the matrix which is obtained from the adjacency matrix of G by replacing its diagonal entries by 1. Then, B_G is a Robinson matrix precisely when the 1-entries of each row and each column are all consecutive. This is called the *symmetric consecutive ones property* (*i.e.* C1P for rows and columns simultaneously), and it is known to characterize *unit interval graphs*, or equivalently as shown in [21], *proper interval graphs* (see [8, 9, 17]). Precisely, a graph is a unit interval graph if and only if there exists a linear order on its vertices with respect to which B_G has the consecutive ones property, *i.e.* is a Robinson matrix.

The parameter Γ_1 of the augmented adjacency matrix counts the number of triples (i, j, k), $1 \le i < j < k \le n$, for which vertices i and k are adjacent, but vertex j is not adjacent to either i or k. On the other hand, the (unnormalized) ℓ^1-distance

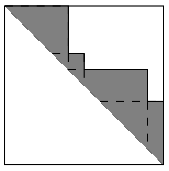

FIG. 1. *The black region is convex around the diagonal.*

$\|B_G - B_{\widehat{G}}\|_1$ between the augmented adjacency matrices B_G and $B_{\widehat{G}}$ of two labeled graphs G and $\widehat{G}$ of the same order corresponds to the edit distance between the graphs themselves. The *edit distance* between two graphs G and $\widehat{G}$, denoted by $ed(G, \widehat{G})$, is the minimum number of edge deletions and edge additions that need to be performed on G to transfer it to $\widehat{G}$. Applying these concepts, Theorem 2.2 directly leads to the following corollary.

COROLLARY 2.4. *For every graph G on vertex set $V = \{1, 2, \dots, n\}$, there exists a unit interval graph $\widehat{G}$ on vertex set V so that*

$$\frac{ed(G, \widehat{G})}{n^2} \le 26\,\Gamma_1(G)^{1/3}.$$

3. Robinson similarity approximation for binary matrices. In this section, we present an algorithm that finds a Robinson approximation for the special case where A is a binary matrix, and can thus be interpreted as the adjacency matrix of a graph. The algorithm can be intuitively understood as follows. We divide all cells of the matrix into *black* and *white* cells, and convert all zeros in the black cells to ones, and all ones in the white cells to zeros. The black region is *convex around the diagonal*, in the sense that, if a cell is black, then so are all other cells closer to the diagonal in the same row or column. In other words, the binary matrix whose support is precisely the black region is a Robinson matrix, which is indeed the Robinson approximation that the algorithm returns (See Figure 1).

The decision on whether to assign a cell to the black or white region depends on the entries in the *upper right (UR)* and *lower left (LL)* regions defined by the cell. Precisely, for any cell (a, b), $1 \le a < b \le n$, we define

$$\begin{aligned} \mathrm{UR}(a, b) &= \{(i, j) : i < a < b < j\}, \\ \mathrm{LL}(a, b) &= \{(i, j) : a \le i \le j \le b\}. \end{aligned}$$

Roughly speaking, a cell will be black if it has enough ones in its upper right region, and it is white when it has enough zeros in its lower left region. So, we need the following notations (also shown in Figure 2):

$$\begin{aligned} 1_{\mathrm{UR}}(a, b) &= |\mathrm{UR}(a, b) \cap \{(i, j) : A_{ij} = 1\}|, \\ 0_{\mathrm{LL}}(a, b) &= |\mathrm{LL}(a, b) \cap \{(i, j) : A_{ij} = 0\}|. \end{aligned}$$

In addition, define $1_{\mathrm{UR}}(a, b) = 0_{\mathrm{LL}}(a, b) = 0$ when $a \in \{0, n+1\}$ or $b \in \{0, n+1\}$ or $a > b$.

Algorithm 3.1 has complexity $\theta(n^2)$, and is therefore linear in the size of the input. A cell is considered black if $R_{i,j}$ is set to one, and white if $R_{i,j}$ is set to zero. If (i, j)

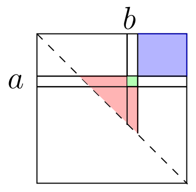

FIG. 2. *Regions* $\mathrm{UR}(a,b)$ *(blue) and* $\mathrm{LL}(a,b)$ *(red)*

Algorithm 3.1 Robinson approximation of a binary matrix

```
input: binary matrix $A \in \mathcal{A}_n$, threshold $t > 0$
output: binary Robinson matrix $R \in \mathcal{A}_n$
for $i \leftarrow 1$ to $n$ do
  $R_{1,i} \leftarrow 0$; $R_{i,1} \leftarrow 0$
  $R_{i,n} \leftarrow 0$; $R_{n,i} \leftarrow 0$
end for
for $i \leftarrow 2$ to $n$ do
  $j \leftarrow n-1$
  while $j \geq i$ do
    $1_{\mathrm{UR}}(i,j) \leftarrow 1_{\mathrm{UR}}(i-1,j) + 1_{\mathrm{UR}}(i,j+1) - 1_{\mathrm{UR}}(i-1,j+1) + A_{i-1,j+1}$
    if $1_{\mathrm{UR}}(i,j) < t$ then $R_{i,j} \leftarrow 0$; $R_{j,i} \leftarrow 0$;
    else $R_{i,j} \leftarrow 1$; $R_{j,i} \leftarrow 1$
    $j \leftarrow j-1$;
  end while
end for
return $R$
```

is black and $i \leq k < j$, then $\mathrm{UR}(i,j) \subset \mathrm{UR}(i,k)$, so $1_{\mathrm{UR}}(i,k) \geq 1_{\mathrm{UR}}(i,j) \geq t$, and thus (i,k) is also black. Similarly, (k,j) is also black. So, the region of black cells is convex around the diagonal, and R is indeed a Robinson matrix. We will now show in Theorem 3.1 that the distance between R and A is bounded by a function of t and n, if the parameter t satisfies Condition (3.1). We will then show in Corollary 3.3 that an appropriate t can be chosen as a function of $\Gamma_1(A)$, so that $\|A-R\|_1$ can be bounded in terms of $\Gamma_1(A)$.

THEOREM 3.1. *Let $A \in \mathcal{A}_n$ be a binary matrix, with the property that*

$$\text{for all } 1 \leq i \leq j \leq n,\ 1_{\mathrm{UR}}(i,j) < t \text{ or } 0_{\mathrm{LL}}(i,j) < t. \tag{3.1}$$

If R is the matrix produced as output of Algorithm 3.1 on input A with threshold t, then $\|A-R\|_1 \leq \frac{16\sqrt{t}+4}{n}$.

Proof. As explained earlier, black cells are exactly the cells (i,j) for which $R_{i,j}$ attains 1, *i.e.* the cells for which $1_{\mathrm{UR}}(i,j) \geq t$. Let $\mathcal{B}$ denote the collection of all black cells above the diagonal, that is

$$\mathcal{B} = \{(i,j) : 1 \leq i \leq j \leq n \text{ and } 1_{\mathrm{UR}}(i,j) \geq t\}.$$

Note that $\mathcal{B}$ is convex around the diagonal, in the sense that if a cell (i,j) belongs to $\mathcal{B}$ then $\mathrm{LL}(i,j) \subseteq \mathcal{B}$. However, the black region can be disconnected. Precisely, there can be diagonal cells not in $\mathcal{B}$, in which case $\mathcal{B}$ consists of a collection of connected regions which are convex around the diagonal. Note also that, by definition, $\mathcal{B}$ cannot contain any cells (i,j) so that $|\mathrm{UR}(i,j)| < t$, and specifically, $\mathcal{B}$ cannot contain any cells from the first row or the last column.

Let $\partial\mathcal{B}$ denote the set of boundary cells of $\mathcal{B}$, *i.e.*

$$\partial(\mathcal{B}) = \{(i,j) \in \mathcal{B} : \ (i-1,j) \notin \mathcal{B} \ \text{ or } \ (i,j+1) \notin \mathcal{B}\}.$$

The set $\partial\mathcal{B}$ precisely contains all black cells above the diagonal that are adjacent to cells outside $\mathcal{B}$. Thus,

$$\mathcal{B} = \bigcup_{(i,j)\in\partial\mathcal{B}} \mathrm{LL}(i,j). \tag{3.2}$$

Similarly, the white region $\mathcal{W}$ and its boundary cells are defined as

$$\mathcal{W} = \{(i,j) : 1 \le i \le j \le n \text{ and } 1_{\mathrm{UR}}(i,j) < t\},$$

and

$$\partial\mathcal{W} = \{(i,j) \in \mathcal{W} : \ (i+1,j) \notin \mathcal{W} \ \text{ or } \ (i,j-1) \notin \mathcal{W}\}.$$

Claim 1. Let $\mathcal{B}_0 = \{(i,j) \in \mathcal{B} : \ A_{i,j} = 0\}$. Then $|\mathcal{B}_0| \le 4n\sqrt{t}$

Proof of Claim 1. First, list elements of $\partial\mathcal{B}$ as $(i_1,j_1),(i_2,j_2),\ldots,(i_m,j_m)$ in such a way that $i_1 \le i_2 \le \ldots \le i_m$, and if $i_l = i_{l+1}$ then $j_l < j_{l+1}$. This ordering follows the "contour" of the black region, starting with the first black cell in the first row containing any black cells. Since both indices range from at least 1 to at most n, it is clear that the boundary contains at most $2n$ cells, and $m \le 2n$.

To control the amount of possible overlaps in the covering of $\mathcal{B}$ in (3.2), we now construct a subsequence $\mathcal{C}$ of $\partial\mathcal{B}$, so that the lower left regions of cells in $\mathcal{C}$ cover most of the black region. The subsequence $\mathcal{C} = \{(i_{n_k}, j_{n_k})\}$ is constructed inductively. In the first step, let (i_{n_1}, j_{n_1}) be the last cell in the first row of $\mathcal{B}$. At step $k \ge 1$, let n_k be the largest index so that $(i_{n_k}, j_{n_k}) \in \mathcal{C}$. Define n to be the first index in $\{1,\ldots,m\}$ which satisfies $j_{n_k} < j_n$, $(i_n, j_n+1) \notin \mathcal{B}$, and either $i_n - i_{n_k} > \lfloor\sqrt{t}\rfloor$ or $j_n - j_{n_k} > \lfloor\sqrt{t}\rfloor$. So cell (i_n,j_n) is the first cell in $\partial\mathcal{B}$ whose row or column index differs by at least $\lfloor\sqrt{t}\rfloor + 1$ from the last cell added to $\mathcal{C}$, and also is the last black cell in its row. Set $n_{k+1} = n$, and add $(i_{n_{k+1}}, j_{n_{k+1}})$ to $\mathcal{C}$. Since $\partial\mathcal{B}$ has at most $2n$ elements, and $\lfloor\sqrt{t}\rfloor + 1 \ge \sqrt{t}$, this inductive process ends in at most $\frac{2n}{\sqrt{t}}$ steps. So $|\mathcal{C}| \le \frac{2n}{\sqrt{t}}$.

We now claim that $\mathcal{B} \setminus \bigcup_{(i,j)\in\mathcal{C}} \mathrm{LL}(i,j)$ can be covered with $|\mathcal{C}|$ squares of dimensions $\lfloor\sqrt{t}\rfloor \times \lfloor\sqrt{t}\rfloor$. Consider two consecutive elements of $\mathcal{C}$, say (i_{n_k}, j_{n_k}) and $(i_{n_{k+1}}, j_{n_{k+1}})$. Let (i,j) be the first cell in $\partial\mathcal{B} \setminus \bigcup_{(i,j)\in\mathcal{C}} \mathrm{LL}(i,j)$ after (i_{n_k}, j_{n_k}). By construction, (i,j) is not in the same row or column as (i_{n_k}, j_{n_k}), and thus $i > i_{n_k}$ and $j > j_{n_k}$. Let (i',j') be the last cell before $(i_{n_{k+1}}, j_{n_{k+1}})$ in $\partial\mathcal{B} \setminus \bigcup_{(i,j)\in\mathcal{C}} \mathrm{LL}(i,j)$. (If no such n exists, then k is the last index in $\mathcal{C}$, and we let $(i',j') = (i_m,j_m)$, the last cell of $\partial\mathcal{B}$.)

By construction of $\mathcal{C}$, we know that $i' - i + 1 \le i' - i_{n_k} \le \lfloor\sqrt{t}\rfloor$ and $j' - j + 1 \le j' - j_{n_k} \le \lfloor\sqrt{t}\rfloor$. Moreover, since (i,j) and (i',j') belong to $\partial\mathcal{B}$, and $\mathcal{B}$ is convex around the diagonal, every cell $(a,b) \in \mathcal{B}$ with $i \le a \le i'$ must satisfy $j \le b \le j'$, so it must belong to the square whose top-left corner is (i,j), and its bottom-right corner is (i',j'). We denote this square by S_k, and note that $|S_k| \le t$. Thus,

$$\mathcal{B} \subseteq \bigcup_{k=1}^{|\mathcal{C}|} \mathrm{LL}(i_{n_k}, j_{n_k}) \cup \bigcup_{k=1}^{|\mathcal{C}|} S_k.$$

By Condition (3.1), every cell $(i,j) \in \mathcal{B}$ satisfies $0_{\rm LL}(i,j) \le t$. So,

$$|\mathcal{B}_0| \le \sum_{(i,j)\in\mathcal{C}} 0_{\rm LL}(i,j) + \sum_{(i,j)\in\mathcal{C}} |S_{i,j}| \le |\mathcal{C}|t + |\mathcal{C}|t = 4n\sqrt{t}.$$

□

Claim 2. Let $\mathcal{W}_1 = \{(i,j) \in \mathcal{W} :\ A_{i,j} = 1\}$. Then $|\mathcal{W}_1| \le 4n\sqrt{t} + 2n$.

Proof of Claim 2. This proof is similar to the proof of the previous claim. Cells in the boundary $\partial\mathcal{W}$ are enumerated similarly, and a subsequence $\mathcal{D} = \{(i_{n_k}, j_{n_k})\}$ of $\partial\mathcal{W}$ is defined analogous to $\mathcal{C}$: let (i_{n_1}, j_{n_1}) be the last cell in the first column of $\mathcal{W}$, and for $k \ge 1$, define n_{k+1} to be the first index in $\{1,\dots,m\}$ which satisfies $i_{n_{k+1}} > i_{n_k}$, $(i_{n_{k+1}}+1, j_{n_{k+1}}) \notin \mathcal{W}$, and either $i_{n_{k+1}} - i_{n_k} > \lfloor\sqrt{t}\rfloor$ or $j_{n_{k+1}} - j_{n_k} > \lfloor\sqrt{t}\rfloor$. Let S_k be the square of size at most $\lfloor\sqrt{t}\rfloor \times \lfloor\sqrt{t}\rfloor$ covering all white cells between consecutive cells of $\mathcal{D}$. Then we have

$$\mathcal{W} \subseteq \partial\mathcal{W} \cup \bigcup_{(i,j)\in\mathcal{D}} {\rm UR}(i,j) \cup \bigcup_{k=1}^{|\mathcal{D}|} S_k$$

Note that dealing with the white region is slightly different from the black region, in the sense that ${\rm UR}(i,j)$ does not include cell (i,j) and the cells in row i and column j (see Figure 2). So, we include the boundary explicitly to cover the white region.

By definition of $\mathcal{W}$, every white cell $(i,j) \in \mathcal{W}$ satisfies $1_{\rm UR}(i,j) < t$. This implies that

$$|\mathcal{W}_1| \le |\partial\mathcal{W}| + \sum_{(i,j)\in\mathcal{D}} 1_{\rm UR}(i,j) + \sum_{i=1}^{|\mathcal{D}|} |S_k| \le 2n + |\mathcal{D}|t + |\mathcal{D}|t \le 4n\sqrt{t} + 2n.$$

□

The above two claims show that R and A differ above the diagonal in at most $8n\sqrt{t} + 2n$ cells. Adding the region below the diagonal and normalizing, we conclude that $\|A - R\|_1 \le \frac{2(8\sqrt{t}+2)}{n}$. □

The following simple counting lemma provides a threshold, in terms of Γ_1, for which Condition (3.1) always holds.

LEMMA 3.2. *Let $A \in \mathcal{A}_n$ be a binary matrix whose diagonal entries are all 1. Then, for every cell (i,j), we have*

$$1_{\rm UR}(i,j)\ 0_{\rm LL}(i,j) \le 2n^4\Gamma_1(A).$$

Proof. Without loss of generality, assume that $i < j$.

$$\begin{aligned}
1_{\rm UR}(i,j)\ 0_{\rm LL}(i,j) = & \sum_{\substack{(s,t)\,\in\,{\rm UR}(i,j)\\ (s',t')\,\in\,{\rm LL}(i,j)}} [A_{s,t} - A_{s',t'}]_+ \\
& \le \frac{1}{2} \sum_{1\le s<i\le s'\le t'\le j<t\le n} [A_{s,t} - A_{s,t'}]_+ + [A_{s,t'} - A_{s',t'}]_+ + [A_{s,t} - A_{s',t}]_+ + [A_{s',t} - A_{s',t'}]_+ \\
& \le \frac{1}{2} \sum_{1\le s<s'\le t'<t\le n} [A_{s,t} - A_{s,t'}]_+ + [A_{s,t} - A_{s',t}]_+ \\
& + \frac{1}{2} \sum_{1\le s<s'\le t'<t\le n} [A_{s,t'} - A_{s',t'}]_+ + [A_{s',t} - A_{s',t'}]_+,
\end{aligned}$$

where the second inequality can be justified using the fact that if $[A_{s,t} - A_{s',t'}]_+ = 1$ then both $[A_{s,t} - A_{s,t'}]_+ + [A_{s,t'} - A_{s',t'}]_+ = 1$ and $[A_{s,t} - A_{s',t}]_+ + [A_{s',t} - A_{s',t'}]_+ = 1$ are satisfied. Now observe that

$$\sum_{1 \le s < s' \le t' < t \le n} [A_{s,t} - A_{s,t'}]_+ \le \sum_{s'=1}^{n} \sum_{1 \le s < t' < t \le n} [A_{s,t} - A_{s,t'}]_+ \le n^4 \Gamma_1(A),$$

and similarly $\sum_{1 \le s < s' \le t' < t \le n} [A_{s,t} - A_{s',t}]_+ \le n^4 \Gamma_1(A)$. Moreover, for every s and s', we have $[A_{s,s'} - A_{s',s'}]_+ = 0$, since $A_{s',s'} = 1$. Thus,

$$\begin{aligned}\sum_{1 \le s < s' \le t' < t \le n} [A_{s,t'} - A_{s',t'}]_+ &= \sum_{1 \le s < s' < t' < t \le n} [A_{s,t'} - A_{s',t'}]_+ \\ &\le \sum_{t=1}^{n} \sum_{1 \le s < s' < t' \le n} [A_{s,t'} - A_{s',t'}]_+ \le n^4 \Gamma_1(A),\end{aligned}$$

and similarly $\sum_{1 \le s < s' \le t' < t \le n} [A_{s',t} - A_{s',t'}]_+ \le n^4 \Gamma_1(A)$. This finishes the proof. ∎

Corollary 3.3. *For every binary matrix $A \in \mathcal{A}_n$, there exists a binary Robinson matrix $R \in \mathcal{A}_n$ such that*

$$\|A - R\|_1 \le \frac{5}{n} + 2^{9/2} \Gamma_1(A)^{1/4}.$$

Moreover, R can be computed in linear time.

Proof. From the binary matrix $A \in \mathcal{A}_n$, we first construct $\widetilde{A}$ by replacing every 0 entry on the diagonal of A with 1. Clearly, $\widetilde{A} \in \mathcal{A}_n$ and $\Gamma_1(\widetilde{A}) \le \Gamma_1(A)$. Moreover, $\|A - \widetilde{A}\|_1 \le \frac{1}{n}$. If $\Gamma_1(\widetilde{A}) = 0$, we take $R = \widetilde{A}$ and we are done. So assume that $\Gamma_1(\widetilde{A}) > 0$. Let $R \in \mathcal{A}_n$ be the matrix produced as output of Algorithm 3.1 on input $\widetilde{A}$ with threshold $t = n^2\sqrt{4\Gamma_1(\widetilde{A})}$. By Lemma 3.2 and the fact that $\Gamma_1(\widetilde{A}) > 0$, $\widetilde{A}$ and parameter $t = n^2\sqrt{4\Gamma_1(\widetilde{A})}$ satisfy Condition (3.1) of Theorem 3.1. Thus,

$$\|R - A\|_1 \le \|A - \widetilde{A}\|_1 + \|\widetilde{A} - R\|_1 \le \frac{5}{n} + 16(4\Gamma_1(\widetilde{A}))^{1/4} \le \frac{5}{n} + 2^{9/2}\Gamma_1(A)^{1/4}.$$

∎

4. Robinson similarity approximations of general matrices.

For general matrices, we first decompose the matrix into a convex combination of binary matrices, a standard technique widely used in the literature, and then apply Algorithm 3.1 to each summand. Given any matrix $A \in \mathcal{A}_n$, let range$(A) = \{A_{i,j} : 1 \le i \le j \le n\}$. Consider the linear ordering $0 = s_0 < s_1 < \cdots < s_m$ of range$(A) \cup \{0\}$, and define matrices $A^{(k)}$, $1 \le k \le m$ as follows:

$$A^{(k)}_{i,j} = \begin{cases} 1 & \text{if } A_{i,j} \ge s_k, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

Clearly, each matrix $A^{(k)}$ is binary, and

$$A = \sum_{k=1}^{m} (s_k - s_{k-1}) A^{(k)}. \tag{4.2}$$

We refer to the matrices $A^{(k)}$ as the *layers* of A. From the definition of the layers, it is easy to see that $A^{(k)} \leq A^{(l)}$ whenever $l < k$. Indeed for such l and k, if $A^{(k)}_{i,j} = 1$ then $A^{(l)}_{i,j} = 1$ as well, since $s_l < s_k$.

The advantage of writing A as a convex combination of its layers, as opposed to any other decomposition of A, lies in the fact that Γ_1 distributes over this particular decomposition of A, even though Γ_1 is not a linear map in general.

PROPOSITION 4.1. *Suppose A is "layered" as in Equation (4.2). Then we have*

$$\Gamma_1(A) = \sum_{l=1}^{m} (s_l - s_{l-1}) \Gamma_1(A^{(l)}). \tag{4.3}$$

Proof. Fix a triple i, j, k satisfying $1 \leq i < k < j \leq n$, and let $n_1, n_2, n_3 \in \{1, \ldots, m\}$ be so that $A_{i,j} = s_{n_1}$, $A_{i,k} = s_{n_2}$, and $A_{k,j} = s_{n_3}$. From the definition of the layers in (4.1), we have

(i) $A^{(l)}_{i,j} = 1$ if $l \leq n_1$, and $A^{(l)}_{i,j} = 0$ if $l > n_1$.
(ii) $A^{(l)}_{i,k} = 1$ if $l \leq n_2$, and $A^{(l)}_{i,k} = 0$ if $l > n_2$.
(iii) $A^{(l)}_{k,j} = 1$ if $l \leq n_3$, and $A^{(l)}_{k,j} = 0$ if $l > n_3$.

Note first that $[A_{i,j} - A_{i,k}]_+ = [s_{n_1} - s_{n_2}]_+ = 0$ precisely when $n_1 \leq n_2$. Using (i), (ii) and (iii) it is easy to see that, if $n_1 \leq n_2$, then $[A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ = 0$ for every $1 \leq l \leq m$. On the other hand, if $[A_{i,j} - A_{i,k}]_+ > 0$ and thus $n_1 > n_2$, then, if $l \leq n_2$ or $l > n_1$ then $[A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ = 0$, and if $n_2 < l \leq n_1$ then $[A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ = 1$. Hence, we can verify the following claim:

$$[A_{i,j} - A_{i,k}]_+ = \sum_{l=1}^{m} (s_l - s_{l-1}) [A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+.$$

Indeed, if $n_1 \leq n_2$ then both sides of the above equation are equal to 0. For the case where $n_1 > n_2$, we have

$$\sum_{l=1}^{m} (s_l - s_{l-1}) [A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ = \sum_{l=n_2+1}^{n_1} s_l - s_{l-1} = s_{n_1} - s_{n_2} = [A_{i,j} - A_{i,k}]_+.$$

Repeating the above argument, we obtain a similar claim for $[A^{(l)}_{i,j} - A^{(l)}_{k,j}]_+$, and consequently we get

$$[A_{i,j} - A_{i,k}]_+ + [A_{i,j} - A_{k,j}]_+ = \sum_{l=1}^{m} (s_l - s_{l-1}) \Big([A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ + [A^{(l)}_{i,j} - A^{(l)}_{k,j}]_+ \Big).$$

So we have

$$\begin{aligned}
\sum_{l=1}^{m} (s_l - s_{l-1}) \Gamma_1(A^{(l)}) &= \sum_{l=1}^{m} (s_l - s_{l-1}) \Big(\frac{1}{n^3} \sum_{1 \leq i < k < j \leq n} [A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ + [A^{(l)}_{i,j} - A^{(l)}_{k,j}]_+ \Big) \\
&= \frac{1}{n^3} \sum_{1 \leq i < k < j \leq n} \sum_{l=1}^{m} (s_l - s_{l-1}) \Big([A^{(l)}_{i,j} - A^{(l)}_{i,k}]_+ + [A^{(l)}_{i,j} - A^{(l)}_{k,j}]_+ \Big) \\
&= \frac{1}{n^3} \sum_{1 \leq i < k < j \leq n} [A_{i,j} - A_{i,k}]_+ + [A_{i,j} - A_{k,j}]_+ = \Gamma_1(A),
\end{aligned}$$

which finishes the proof. ∎

Algorithm 4.1 Robinson similarity approximation of a general matrix

```
input: Matrix $A \in \mathcal{A}_n$, positive thresholds $t_1, \ldots, t_m$
output: Robinson similarity matrix $R \in \mathcal{A}_n$
Compute range$(A) \cup \{0,1\}$, as an ordered list $s$
for $i \leftarrow 1$ to $n$ do
  for $k \leftarrow 1$ to $m$ do
    $R^{(k)}_{1,i} \leftarrow 0$; $R^{(k)}_{i,1} \leftarrow 0$
    $R^{(k)}_{i,n} \leftarrow 0$; $R^{(k)}_{n,i} \leftarrow 0$
    $A_{i,i} \leftarrow 1$
  end for
end for
for $i \leftarrow 2$ to $n$ do
  $j \leftarrow n-1$
  while $j \geq i$ do
    for $k \leftarrow 1$ to $m$ do
      if $A_{i,j} \geq s[k]$ then $temp \leftarrow 1$;
      else $temp \leftarrow 0$
      $1^k_{\mathrm{UR}}(i,j) \leftarrow 1^k_{\mathrm{UR}}(i-1,j) + 1^k_{\mathrm{UR}}(i,j+1) - 1^k_{\mathrm{UR}}(i-1,j+1) + temp$
      if $1^k_{\mathrm{UR}}(i,j) < t_k$ then $R^{(k)}_{i,j} \leftarrow 0$; $R^{(k)}_{j,i} \leftarrow 0$;
      else $R^{(k)}_{i,j} \leftarrow 1$; $R^{(k)}_{j,i} \leftarrow 1$
    end for
    $j \leftarrow j-1$;
  end while
end for
$R \leftarrow \mathbf{0}$
for $k \leftarrow 1$ to $m$ do $R \leftarrow R + (s[k] - s[k-1])R^{(k)}$
return $R$
```

Algorithm 4.1 is essentially a simultaneous execution of Algorithm 3.1 for each binary matrix $A^{(k)}$. The quantities $1^k_{\mathrm{UR}}(i,j)$ thus refer to the number of ones in the region $\mathrm{UR}(i,j)$ in $A^{(k)}$. The algorithm has complexity $O(n^2m)$, where m is the size of range(A), and thus $m \leq n^2$. Since every matrix $R^{(k)}$ is Robinson similarity, so is their linear combination R. In the following theorem, we will show that there exist thresholds $t_1, \ldots, t_m$ so that the difference between A and R is bounded by $C\Gamma_1(A)^{1/4}$ for some constant C. To avoid anomalous behavior, we assume that our input matrix is not Robinson similarity, *i.e.* $\Gamma_1(A) > 0$.

THEOREM 4.2. *Let $A \in \mathcal{A}_n$, and $\Gamma_1(A) > 0$. If R is the matrix produced as output of Algorithm 4.1 on input A with thresholds $t_k = \sqrt{4\Gamma_1(A^{(k)})}n^2$ for $k = 1, \ldots, m$, then*

$$\|A - R\|_1 \leq 2^{9/2}\Gamma_1(A)^{1/4}(1 + O(n^{-1/4})).$$

Proof. First suppose A is a binary matrix. Then by Corollary 3.3 applied to A with threshold $\sqrt{4\Gamma_1(A)}n^2$, we get $\|A - R\|_1 \leq \frac{5}{n} + 2^{9/2}\Gamma_1(A)^{1/4}$.

Next, assume that $A \in \mathcal{A}_n$ is a general matrix, and let $A = \sum_{k=1}^m (s_k - s_{k-1})A^{(k)}$ be the decomposition of A into layers of binary matrices as described in Equation (4.2). Then, we have $\Gamma_1(A) = \sum_{k=1}^m (s_k - s_{k-1})\Gamma_1(A^{(k)})$. To avoid clutter of notation, let $\epsilon = \Gamma_1(A)$ and $\epsilon_k := \Gamma_1(A^{(k)})$. For every $1 \leq k \leq m$, we apply Corollary 3.3 to $A^{(k)}$ with threshold $t_k = (4\epsilon_k)^{1/2}n^2$ to obtain a Robinson similarity matrix $R^{(k)}$ such that

$$\|A^{(k)} - R^{(k)}\|_1 \leq \frac{5}{n} + 2^{9/2}\epsilon_k^{1/4}.$$

So, Algorithm 4.1 computes $R = \sum_{k=1}^{m}(s_k - s_{k-1})R^{(k)}$, which is Robinson similarity as well. Moreover,

$$\begin{aligned}\|A - R\|_1 &\leq \sum_{k=1}^{m}(s_k - s_{k-1})\|A^{(k)} - R^{(k)}\|_1 \\ &\leq 2^{9/2}\sum_{k=1}^{m}(s_k - s_{k-1})\epsilon_k^{1/4} + \sum_{k=1}^{m}(s_k - s_{k-1})\frac{5}{n} \\ &\leq 2^{9/2}(\Gamma_1(A))^{1/4} + \frac{5}{n},\end{aligned}$$

where in the last inequality we used the fact that the function $f(x) = x^{1/4}$ is concave.

By definition, if $A \in \mathcal{A}_n$ is not Robinson similarity, then $\Gamma_1(A) \geq \frac{1}{n^3}$, and thus $n\Gamma_1(G)^{1/4} \geq n^{1/4}$. This completes the proof. ∎

5. Improvement through preprocessing. Finally, we give an algorithm that represents a preprocessing step for Algorithm 3.1. By Theorem 3.1, the Robinson similarity approximation produced by Algorithm 3.1 is at bounded distance from the input matrix A, provided that Condition (3.1) holds. By Lemma 3.2, the condition holds for every matrix, if we choose $t > \sqrt{2}n^2(\Gamma_1(A))^{1/2}$ (see proof of Corollary 3.3.) The preprocessing step is designed to transform the input matrix A, such that the condition holds for a smaller value of t. The modified matrix is then used as input to Algorithm 3.1 with the new value of t, which leads to an output matrix R that is closer to the input matrix.

The trade-off here is that the preprocessing step increases the complexity of the algorithm. This increase is tolerable, as the complexity still remains polynomial. We give an implementation which is quadratic for binary matrices, but we believe that a more sophisticated implementation would lead to an improvement in the complexity. In the following, the algorithm is given in detail for binary matrices only. It can easily be adapted to general matrices, in much the same way that Algorithm 4.1 is adapted from Algorithm 3.1.

First, we introduce some terminology. Suppose that matrix A is given, and a threshold value $t > 0$ is fixed. Let Δ denote the collection of all cells in A which are on or above the diagonal. We call a cell $(i, j) \in \Delta$ *inverted* if $1_{\text{UR}}(i, j) \geq t$ and $0_{\text{LL}}(i, j) \geq t$. Thus, there exist inverted cells if and only if Condition (3.1) is violated. To *toggle* a cell (i, j) is to set the value of all cells in $\text{UR}(i, j)$ equal to zero, and the values of all cells in $\text{LL}(i, j)$ equal to 1. It is easy, yet important, to observe that toggling a cell can only decrease $\Gamma_1(A)$.

LEMMA 5.1. *Let $A \in \mathcal{A}_n$ and $t > 0$ be given. Suppose that $(i, j) \in \Delta$ is an inverted cell, and let $\widetilde{A}$ denote the matrix obtained from A by toggling (i, j). Then for any fixed triple $1 \leq s < k < t \leq n$, we have*

$$[\widetilde{A}_{s,t} - \widetilde{A}_{s,k}]_+ \leq [A_{s,t} - A_{s,k}]_+ \quad \textit{and} \quad [\widetilde{A}_{s,t} - \widetilde{A}_{k,t}]_+ \leq [A_{s,t} - A_{k,t}]_+. \tag{5.1}$$

Consequently, we get $\Gamma_1(\widetilde{A}) \leq \Gamma_1(A)$.

Proof. We only prove the first inequality; the second one can be proved similarly. Fix a triple $1 \leq s < k < t \leq n$. First note that, the first inequality in (5.1) holds trivially whenever $(s,t), (s,k) \in \Delta \setminus (\mathrm{LL}(i,j) \cup \mathrm{UR}(i,j))$, as on these cells the matrices A and $\widetilde{A}$ are identical. For the cases where $(s,t) \in \mathrm{UR}(i,j)$ or $(s,k) \in \mathrm{LL}(i,j)$, we have $\widetilde{A}(s,t) = 0$ or $\widetilde{A}(s,k) = 1$. Thus, in both cases, we get $[\widetilde{A}_{s,t} - \widetilde{A}_{s,k}]_+ = 0$, and in particular $[\widetilde{A}_{s,t} - \widetilde{A}_{s,k}]_+ \leq [A_{s,t} - A_{s,k}]_+$. Moreover, it is clear from the definition of the region $\mathrm{LL}(i,j)$ that if $(s,t) \in \mathrm{LL}(i,j)$ then $(s,k) \in \mathrm{LL}(i,j)$. Similarly, if $(s,k) \in \mathrm{UR}(i,j)$ then $(s,t) \in \mathrm{UR}(i,j)$ as well. Putting all these together, we conclude that the desired inequality holds in all cases. Therefore,

$$\begin{aligned}\Gamma_1(\widetilde{A}) &= \frac{1}{n^3} \sum_{1 \leq s < k < t \leq n} [\widetilde{A}_{s,t} - \widetilde{A}_{s,k}]_+ + [\widetilde{A}_{s,t} - \widetilde{A}_{k,t}]_+ \\ &\leq \frac{1}{n^3} \sum_{1 \leq s < k < t \leq n} [A_{s,t} - A_{s,k}]_+ + [A_{s,t} - A_{k,t}]_+ \\ &= \Gamma_1(A),\end{aligned}$$

which finishes the proof. □

Algorithm 5.1 can then be described as follows: each inverted cell is toggled, and after each toggling step, the values of 1_{UR} and 0_{LL} are recalculated for each cell.

Algorithm 5.1 Preprocessing step

```
input: Matrix $A \in \mathcal{A}_n$, threshold $t$
output: Updated matrix $A$
for $i \leftarrow 1$ to $n$ do
  for $j \leftarrow i$ to $n$ do
    Compute $1_{\mathrm{UR}}(i,j)$ and $0_{\mathrm{LL}}(i,j)$
    if $1_{\mathrm{UR}}(i,j) \geq t$ and $0_{\mathrm{LL}}(i,j) \geq t$ then
    for all cells $(r,s) \in \mathrm{UR}(i,j)$ do  $A_{r,s} \leftarrow 0$; $A_{s,r} \leftarrow 0$
    for all cells $(r,s) \in \mathrm{LL}(i,j)$ do  $A_{r,s} \leftarrow 1$; $A_{s,r} \leftarrow 1$
  end for
end for
```

Note that Algorithm 5.1 involves computing $1_{\mathrm{UR}}(a,b)$ for all $1 \leq a \leq i$ and $j \leq b \leq n$, and computing $0_{\mathrm{LL}}(a,b)$ for all $i \leq a \leq j$ and $i \leq b \leq j$. These values must be recomputed in each iteration, since A is being changed. Any cell (i,j) is tested exactly once, to see whether it is inverted, and if it is, it is toggled. This naive implementation of the algorithm has complexity $O(n^4)$ and is thus quadratic in the size of the input. When adapted to general matrices, the complexity becomes $O(mn^4)$, where m is the number of different values taken by entries of A. Clearly $m \leq n^2$. Also, if all entries of A are rounded to the nearest multiple of $\epsilon^A 1/3$ (recall that $\epsilon = \Gamma_1(A)$), then $m = \epsilon^{-1/3}$, while the error between A and R is still of the same order.

One may be concerned that toggling some cells may create new inverted cells, in which case, just considering each cell once would not be sufficient, and the complexity could increase. The following lemma shows this cannot be the case.

Lemma 5.2. *Suppose Algorithm 5.1 is applied to a binary matrix A, with threshold t. Then the output of the algorithm, which we call the modified matrix, satisfies the condition that, for each cell (i,j), $1_{\mathrm{UR}}(i,j) < t$ or $0_{\mathrm{LL}}(i,j) < t$.*

Proof. It suffices to show that, if an inverted cell (i,j) is toggled, no cell that was not inverted before the toggle can become inverted afterwards. Suppose to the contrary that there exists a cell (k,ℓ) which becomes inverted after the toggle. That is, either $1_{\mathrm{UR}}(k,\ell)$ is increased by the toggle, or $0_{\mathrm{LL}}(k,\ell)$ is increased. Assume without loss of generality that $1_{\mathrm{UR}}(k,\ell)$ increases after the toggle. The toggle sets all entries in $\mathrm{UR}(i,j)$ to zero, and all entries to $\mathrm{LL}(i,j)$ to one. So for $1_{\mathrm{UR}}(k,\ell)$ to increase, $\mathrm{UR}(k,\ell)$ and $\mathrm{LL}(i,j)$ must intersect. Thus, $(k,\ell)\in \mathrm{LL}(i,j)$. Therefore, after the toggle, all entries in $\mathrm{LL}(k,\ell)$ have been set to one, so $0_{\mathrm{LL}}(k,\ell)=0<t$. Therefore, (k,ℓ) is not an inverted cell after the toggle, which contradicts our assumption. ◻

The above lemma shows that the modified matrix $\widehat{A}$ returned by Algorithm 5.1 satisfies Condition 3.1 of Theorem 3.1 for the chosen value of t. Thus, this matrix can be used as input for Algorithm 3.1 to obtain a Robinson similarity approximation R, and Theorem 3.1 can be applied to bound $\|\widehat{A}-R\|_1$. However, to obtain a good approximation we need to make sure we can also bound $\|A-\widehat{A}\|_1$, the distance between the input and output matrices of Algorithm 5.1. The following lemma gives such a bound.

LEMMA 5.3. *Let $A\in\mathcal{A}_n$ be a binary matrix. Let $\widehat{A}$ denote the output of Algorithm 5.1 with threshold t. Then $\|A-\widehat{A}\|_1\le \frac{4n^2}{t}(\Gamma_1(A)-\Gamma_1(\widehat{A}))$.*

Proof. The output matrix $\widehat{A}$ is obtained from A by consecutive toggling of inverted cells, occurring in lines 5 and 6 of Algorithm 5.1. The condition in line 4 checks whether a cell is inverted. Let $A=A^0,A^1,A^2,\dots,A^m=\widehat{A}$ denote the matrices in the intermediate steps. We will bound the distance between two consecutive matrices.

Fix s, $0\le s<m$, and assume that A^s is modified because cell (i,j) is found to be inverted, and is then toggled. So $1_{\mathrm{UR}}(i,j)\ge t$ and $0_{\mathrm{LL}}(i,j)\ge t$, where 1_{UR} and 0_{LL} are computed from A^s. So A^{s+1} is formed from A^s by adjusting every cell in $\mathrm{UR}(i,j)$ and its counterpart below the diagonal to be 0, and every cell in $\mathrm{LL}(i,j)$ and its counterpart below the diagonal to be 1.

We can intuitively observe that $\Gamma_1(A^s)$ drops by at least $\frac{0_{LL}(i,j)1_{UR}(i,j)}{n^4}$ when (i,j) is toggled. Note that, normalizing as in the definition of Γ_1 and applying the same reasoning as in Lemma 3.2, one may be tricked into thinking that $\Gamma_1(A^s)$ drops at least by $\frac{2}{n^3}0_{LL}(i,j)1_{UR}(i,j)$ when (i,j) is toggled, since a contribution of $\frac{2}{n^3}$ is removed for each pair of "bad" cells from $\mathrm{UR}(i,j)$ and $\mathrm{LL}(i,j)$. However, this reasoning does not take overcounting into consideration. We dedicate the following claim to a rigorous proof of this intuitive observation.

CLAIM 5.4. *For A, s, and (i,j) as above, we have*

$$\Gamma_1(A^s)-\Gamma_1(A^{s+1})\ge \frac{0_{\mathrm{LL}}(i,j)1_{\mathrm{UR}}(i,j)}{n^4}.$$

Proof of claim. First, consider a cell (k,ℓ) from $\mathrm{UR}(i,j)$ containing 1, and a cell (k',ℓ') from $\mathrm{LL}(i,j)$, containing 0. Similar to the proof of Lemma 3.2, we have that

$$\begin{aligned} 2=2[A^s_{k,\ell}-A^s_{k',\ell'}]_+ &= [A^s_{k,\ell}-A^s_{k',\ell}]_+ + [A^s_{k',\ell}-A^s_{k',\ell'}]_+ \\ &\quad + [A^s_{k,\ell}-A^s_{k,\ell'}]_+ + [A^s_{k,\ell'}-A^s_{k',\ell'}]_+ . \end{aligned} \tag{5.2}$$

By Lemma 5.1, for every triple $1\le k<l'<l\le n$ we have

$$[{A^{s+1}}_{k,l}-{A^{s+1}}_{k,l'}]_+\le [A^s_{k,l}-A^s_{k,l'}]_+ \quad\text{and}\quad [{A^{s+1}}_{k,l}-{A^{s+1}}_{l',l}]_+\le [{A^s}_{k,l}-{A^s}_{l',l}]_+ .$$

This, together with the fact that $A^{s+1}_{k,l}=0$ whenever $(k,l)\in \mathrm{UR}(i,j)$, implies that

$$\sum_{\substack{1\le k<l'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le l'\le j}} [A^s_{k,l}-A^s_{k,l'}]_+ = \sum_{\substack{1\le k<l'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le l'\le j}} [A^s_{k,l}-A^s_{k,l'}]_+ - [A^{s+1}{}_{k,l}-A^{s+1}{}_{k,l'}]_+$$
$$\le \sum_{1\le k<l'<l\le n} [A^s_{k,l}-A^s_{k,l'}]_+ - [A^{s+1}{}_{k,l}-A^{s+1}{}_{k,l'}]_+.$$

Similarly, we have

$$\sum_{\substack{1\le k<k'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le k'\le j}} [A^s_{k,l}-A^s_{k',l}]_+ \le \sum_{1\le k<k'<l\le n} [A^s_{k,l}-A^s_{k',l}]_+ - [A^{s+1}{}_{k,l}-A^{s+1}{}_{k',l}]_+.$$

Adding up the above two inequalities, we get,

$$\sum_{\substack{1\le k<l'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le l'\le j}} [A^s_{k,l}-A^s_{k,l'}]_+ + \sum_{\substack{1\le k<k'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le k'\le j}} [A^s_{k,l}-A^s_{k',l}]_+ \;\le\; n^3(\Gamma(A^s)-\Gamma(A^{s+1})). \tag{5.3}$$

Repeating the above argument for elements of $\mathrm{LL}(i,j)$, the following inequality can be derived in a similar fashion.

$$\sum_{\substack{1\le k<k'<l'\le n\\ (k',l')\in \mathrm{LL}(i,j)\\ 1\le k\le i}} [A^s_{k,l'}-A^s_{k',l'}]_+ + \sum_{\substack{1\le k'<l'<l\le n\\ (k',l')\in \mathrm{LL}(i,j)\\ j\le l\le n}} [A^s_{k',l}-A^s_{k',l'}]_+ \;\le\; n^3(\Gamma(A^s)-\Gamma(A^{s+1})). \tag{5.4}$$

Combining (5.2), (5.3), and (5.4), we conclude the following.

$$\begin{aligned}
1_{\mathrm{UR}}(i,j)0_{\mathrm{LL}}(i,j) &= \sum_{\substack{(k,l)\in \mathrm{UR}(i,j)\\ (k',l')\in \mathrm{LL}(i,j)}} [A^s_{k,\ell}-A^s_{k',\ell'}]_+\\
&= \frac{1}{2}\sum_{\substack{(k,l)\in \mathrm{UR}(i,j)\\ (k',l')\in \mathrm{LL}(i,j)}} ([A^s_{k,\ell}-A^s_{k',\ell}]_+ + [A^s_{k,\ell}-A^s_{k,\ell'}]_+)\\
&+ \frac{1}{2}\sum_{\substack{(k,l)\in \mathrm{UR}(i,j)\\ (k',l')\in \mathrm{LL}(i,j)}} ([A^s_{k',\ell}-A^s_{k',\ell'}]_+ + [A^s_{k,\ell'}-A^s_{k',\ell'}]_+)\\
&\le \frac{n}{2}\sum_{\substack{1\le k<l'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le l'\le j}} [A^s_{k,l}-A^s_{k,l'}]_+ + \frac{n}{2}\sum_{\substack{1\le k<k'<l\le n\\ (k,l)\in \mathrm{UR}(i,j)\\ i\le k'\le j}} [A^s_{k,l}-A^s_{k',l}]_+\\
&+ \frac{n}{2}\sum_{\substack{1\le k<k'<l'\le n\\ (k',l')\in \mathrm{LL}(i,j)\\ 1\le k\le i}} [A^s_{k,l'}-A^s_{k',l'}]_+ + \frac{n}{2}\sum_{\substack{1\le k'<l'<l\le n\\ (k',l')\in \mathrm{LL}(i,j)\\ j\le l\le n}} [A^s_{k',l}-A^s_{k',l'}]_+\\
&\le n^4(\Gamma_1(A^s)-\Gamma_1(A^{s+1})).
\end{aligned}$$

Note that the factor $\frac{n}{2}$ in the above inequalities appear, because every fixed row or column of A^s can have at most n cells from $\mathrm{LL}(i,j)$ or $\mathrm{UR}(i,j)$. This completes the proof of the claim. ◻

Finally, it is easy to observe that $\|A^s - A^{s+1}\|_1 \leq \frac{2(0_{\mathrm{LL}}(i,j)+1_{UR}(i,j))}{n^2}$. Without loss of generality, assume that $1_{\mathrm{UR}} \geq 0_{\mathrm{LL}}$. Then

$$\Gamma_1(A^s)-\Gamma_1(A^{s+1}) \geq \frac{0_{LL}(i,j)1_{UR}(i,j)}{n^4} = \left(\frac{0_{\mathrm{LL}}(i,j)}{4n^2}\right)\left(4\frac{1_{UR}(i,j)}{n^2}\right) \geq \frac{t}{4n^2}\|A^s - A^{s+1}\|_1.$$

Applying the above result, we get

$$\begin{aligned}\Gamma_1(A) - \Gamma_1(\widehat{A}) &= \sum_{s=0}^{m}(\Gamma_1(A^s) - \Gamma_1(A^{s+1})) \geq \sum_{s=0}^{m} \frac{t}{4n^2}\|A^s - A^{s+1}\|_1 \\ &\geq \frac{t}{4n^2}\|\sum_{s=0}^{m} A^s - A^{s+1}\|_1 = \frac{t}{4n^2}\|A - \widehat{A}\|_1,\end{aligned}$$

which finishes the proof. ◻

We now have all the ingredients to prove Theorem 2.2, by combining the above lemma with Theorem 3.1. For this, parameter t must be tuned so that the bounds from the preprocessing step and from Algorithm 5.1 give the best possible result; it appears that the best choice is $t = 4^{-2/3}\Gamma_1(A)^{2/3}n^2$. (Note that this value is smaller than the value used in Corollary 3.3.) For this choice of t, the distance between the input matrix A and the Robinson similarity matrix returned by Algorithm 3.1 when applied to the updated version of matrix A (after being processed by Algorithm 5.1) is bounded by $26\Gamma_1(A)^{1/3}$. As the exponent on $\Gamma_1(A)$ has been decreased from 1/4 to 1/3, the preprocessing step leads to a substantially better Robinson approximation.

Proof of Theorem 2.2. First assume that $A \in \mathcal{A}_n$ is a binary matrix with $\Gamma_1(A) = \epsilon > 0$. Let $\widehat{A}$ be the output of Algorithm 5.1, with threshold $t = 4^{-2/3}\epsilon^{2/3}n^2$. From Lemma 5.3,

$$\|A - \widehat{A}\|_1 \leq 4^{5/3}\epsilon^{-2/3}(\Gamma_1(A) - \Gamma_1(\widehat{A})) \leq 4^{5/3}\epsilon^{1/3}.$$

From Lemma 5.2, we have that $\widehat{A}$ satisfies Condition (3.1) of Theorem 3.1 for our choice of t. Let R be the output of Algorithm 3.1 applied to $\widehat{A}$ with parameter $t = 4^{-2/3}\epsilon^{2/3}n^2$. Then, by Theorem 3.1, we have

$$\|\widehat{A} - R\|_1 \leq \frac{16\sqrt{t}+4}{n} = 4^{5/3}\epsilon^{1/3} + \frac{4}{n}.$$

Combining these inequalities, we get $\|A - R\|_1 \leq 2 \cdot 4^{5/3}\epsilon^{1/3} + \frac{4}{n} \leq 26\epsilon^{1/3}$, where we used the fact that $\Gamma_1(A) = \epsilon \geq \frac{1}{n^3}$ in the last inequality.

Next, assume that $A \in \mathcal{A}_n$ is a general matrix, and let $A = \sum_{k=1}^{m}(s_k - s_{k-1})A^{(k)}$ be the decomposition of A into layers of binary matrices as described in Equation (4.2). Let $\epsilon_k := \Gamma_1(A^{(k)})$, and recall that $\epsilon = \sum_{k=1}^{m}(s_k - s_{k-1})\epsilon_k$. For every $1 \leq k \leq m$, we apply the process described in the above paragraph to $A^{(k)}$ with parameter $t_k = 4^{-2/3}\epsilon_k^{2/3}n^2$, and we obtain Robinson matrices $R^{(k)}$ such that

$$\|A^{(k)} - R^{(k)}\|_1 \leq 26\epsilon_k^{1/3}.$$

Letting $R = \sum_{k=1}^{m}(s_k - s_{k-1})R^{(k)}$, we have

$$\|A - R\|_1 \leq \sum_{k=1}^{m}(s_k - s_{k-1})\|A^{(k)} - R^{(k)}\|_1 \leq 26\sum_{k=1}^{m}(s_k - s_{k-1})\epsilon_k^{1/3} \leq 26\epsilon^{1/3},$$

where in the last inequality we used the fact that the function $f(x) = x^{1/3}$ is concave.

Finally, we observe that the outputs of Algorithm 5.1 and Algorithm 3.1, when applied to a binary matrix A, are again binary matrices. Therefore, the Robinson approximation R of Theorem 2.2 is binary, when A is a binary matrix. $\square$

6. Conclusions and further work.

We defined a parameter Γ_1 which measures how much a matrix resembles a Robinson similarity matrix. We gave a polynomial time algorithm which takes as input a symmetric matrix A, and finds a Robinson matrix R so that the normalized ℓ^1-distance between A and R is bounded by $26\Gamma_1(A)^{1/3}$. The motivation of our work is the application to the problem of seriation of noisy data. This problem can now be approached by solving instead the optimization problem: given a matrix A, find a permutation π of the rows and columns of A so that $\Gamma_1(A^\pi)$ is minimized.

Our construction method is based on a combinatorial algorithm that runs in polynomial time. However, this may not give the best possible such Robinson approximation. As remarked in the introduction, the problem of finding the best possible Robinson approximation, with error measured in ℓ_1 norm, can be formulated as a linear program. An open problem is whether there exists a combinatorial algorithm for this task (as there exist for the ℓ_∞ norm.)

In future work, we propose to study this optimization problem, to attempt to find algorithms which solve or approximate the problem, and to determine their complexity. It is well-known that the second eigenvector of the Laplacian of the matrix, also known as the Fiedler vector, is effective in finding the correct permutation in seriation without error. We propose to study the relationship between the Fiedler vector and the parameter Γ_1 .

An immediate next step is to test our algorithm on real data, and see whether, in practice, the algorithm outperforms the theoretical bound. As well, a clever implementation of Algorithm 5.1 will likely lead to improved efficiency.

Acknowledgments. The authors thank the anonymous referees, whose suggestions greatly improved the paper.

REFERENCES

[1] J. E. Atkins, E. G. Boman, and B. Hendrickson, *A spectral algorithm for seriation and the consecutive ones problem*, SIAM J. Comput., 28 (1999), pp. 297–310, https://doi.org/10.1137/S0097539795285771, http://dx.doi.org/10.1137/S0097539795285771.

[2] J.-P. Barthélemy and F. Brucker, *NP-hard approximation problems in overlapping clustering*, J. Classif., 18 (2001), pp. 159–183, https://doi.org/10.1007/s00357-001-0014-1, https://doi.org/10.1007/s00357-001-0014-1.

[3] C.-H. Chen, *Generalized association plots: information visualization via iteratively correlated matrices*, Statistica Sinica, 12 (2002), pp. 7–29.

[4] V. Chepoi, B. Fichet, and M. Seston, *Seriation in the presence of errors: NP-hardness of ℓ_∞-fitting robinson structures to dissimilarity matrices*, J. Classification, 26 (2009), pp. 279–296.

[5] V. Chepoi and M. Seston, *Seriation in the presence of errors: A factor 16 approximation algorithm for ℓ_∞-fitting robinson structures to distances*, Algorithmica, 59 (2011), pp. 521–568.

[6] H. Chuangpishit, M. Ghandehari, M. Hurshman, J. Janssen, and N. Kalyaniwalla, *Linear embeddings of graphs and graph limits*, Journal of Combinatorial Theory, Series B, 113 (2015), pp. 162 – 184, https://doi.org/https://doi.org/10.1016/j.jctb.2015.02.002, http://www.sciencedirect.com/science/article/pii/S0095895615000179.

[7] H. Chuangpishit, M. Ghandehari, and J. Janssen, *Uniform linear embeddings of graphons*, European J. Combin., 61 (2017), pp. 47–68, https://doi.org/10.1016/j.ejc.2016.09.004, http://dx.doi.org/10.1016/j.ejc.2016.09.004.

[8] D. G. Corneil, *A simple 3-sweep LBFS algorithm for the recognition of unit interval graphs*, Discrete Applied Mathematics, 138 (2004), pp. 371 – 379.

[9] D. G. Corneil, H. Kim, S. Natarajan, S. Olariu, and A. P. Sprague, *Simple linear time recognition of unit interval graphs*, Information Processing Letters, 55 (1995), pp. 99 – 104.

[10] N. Flammarion, C. Mao, and P. Rigollet, *Optimal rates of statistical seriation*. arXiv:1607.02435 [math.ST], 2016.

[11] F. Fogel, A. d'Aspremont, and M. Vojnovic, *Spectral ranking using seriation*, J. Mach. Learn. Res., 17 (2016), pp. Paper No. 88, 45.

[12] M. Ghandehari and J. Janssen, *A stable parameter to quantify geometric structure of graphons*. Unpublished Manuscript, 2018.

[13] M. Hahsler, K. Hornik, and C. Buchta, *Getting things in order: An introduction to the R package seriation*, J. Statistical Software, 25 (3) (2008).

[14] M. Laurent and M. Seminaroti, *A lex-bfs-based recognition algorithm for robinsonian matrices*, Discrete Applied Mathematics, 222 (2017), pp. 151 – 165, https://doi.org/https://doi.org/10.1016/j.dam.2017.01.027, http://www.sciencedirect.com/science/article/pii/S0166218X17300641.

[15] M. Laurent and M. Seminaroti, *Similarity-first search: a new algorithm with application to robinsonian matrix recognition*, SIAM J. Discrete Math., 31 (2017), pp. 1765–1800, https://doi.org/https://doi.org/10.1137/16M1056791.

[16] I. Liiv, *Seriation and matrix reordering methods: An historical overview*, Stat. Anal. Data Min., 3 (2010), pp. 70–91.

[17] P. J. Looges and S. Olariu, *Optimal greedy algorithms for indifference graphs*, Computers & Mathematics with Applications, 25 (1993), pp. 15 – 25.

[18] L. Lovász and B. Szegedy, *Limits of dense graph sequences*, J. Combin. Theory Ser. B, 96 (2006), pp. 933–957, https://doi.org/10.1016/j.jctb.2006.05.002.

[19] B. G. Mirkin and S. N. Rodin, *Graphs and genes*, vol. 11 of Biomathematics, Springer-Verlag, Berlin, 1984. Translated from the Russian by H. Lynn Beus.

[20] P. Préa and D. Fortin, *An optimal algorithm to recognize Robinsonian dissimilarities*, J. Classification, 31 (2014), pp. 351–385, https://doi.org/10.1007/s00357-014-9150-2, http://dx.doi.org/10.1007/s00357-014-9150-2.

[21] F. Roberts, *Indifference graphs*, in Proof Techniques in Graph Theory, F. Harary, ed., Academic Publishers, 1969, pp. 139–146.

[22] W. S. Robinson, *A method for chronologically ordering archeological deposits*, American Antiquity, 16 (1951), pp. 293–301.

[23] M. Seminaroti, *Combinatorial Algorithms for the Seriation Problem*, PhD thesis, Un. Tilburg, 2016.

[24] M. Seston, *A simple algorithm for recognize Robinsonian dissimilarities*, in COMPSTAT 2008—Proceedings in Computational Statistics, Physica-Verlag/Springer, Heidelberg, 2008, pp. 241–248, CD–ROM.