

Mining Maximally Banded Matrices in Binary Data

Faris Alqadah *

Raj Bhatnagar †

Anil Jegga ‡

Abstract

Binary data occurs often in several real world applications ranging from social networks to bioinformatics. Extracting patterns from such data has been a focus of fundamental data mining tasks including association rule analysis, sequence mining and bi-clustering. Recently, the utility of banded structures in binary matrices has been pointed out with applications in paleontology, bioinformatics and social networking. A binary matrix has a banded structure if both the rows and columns can be permuted so that the 1's exhibit a staircase pattern down the rows, along the leading diagonal. In this paper we show the correspondence between bi-clustering and banded structures in matrices; and the MMBS (Mine Maximally Banded Sub-matrices) algorithm is presented as a direct result of this correspondence. The current state of the art algorithm, MBS, only allows for the discovery of a single band and assumes a fixed column permutation. On the other hand MMBS facilitates the discovery of multiple bands that may possibly be overlapping or segmented. Our experimental results, presented here, clearly indicate the advantage of MMBS over MBS with both, synthetic and real data sets.

1 Introduction

Binary matrices occur frequently in many real-world applications such as market-basket data [1], bioinformatics [23], paleontology, ecology [15] and information retrieval [6]. As a result, extracting patterns and clusters in 0-1 data is an important task and has been an active field of study in data mining, resulting in the development of association rule mining [1], sequence mining, and bi-clustering [2] algorithms. In this paper we study the banded structure of binary matrices; a binary matrix is said to be fully banded if both the rows and columns can be permuted such that the 1's exhibit a staircase pattern of overlapping rows along the leading diagonal (figure 1).

While the concept of banded matrices has its origins in numerical analysis [21], the concept has been studied recently in the data mining community [19, 12]. From the data mining perspective, banded structures have myriad applications and natural interpretations. Consider for example a bi-

	A	B	C	D	E
1	1	1	1	0	0
2	0	1	1	0	0
3	0	0	1	0	0
4	0	0	1	1	0
5	0	0	0	1	1

Figure 1: Sample banded matrix

nary matrix containing documents as the rows and keywords as the columns, where the set of documents revolve around the single theme of “clustering”. Early documents on clustering, from around the 60's, may contain terms like “k-means”, while documents in the 70's may contain both “k-means” and “expectation maximization”, and eventually documents in recent years will contain terms like “subspace clustering”, “bi-clustering”, and “curse of dimensionality”. While recent documents may contain the terms “k-means” and “expectation maximization”, we do not expect documents from the 60's or 70's to contain the terms “bi-cluster” or “subspace cluster”. Thus, an evolution of concepts can be seen in the documents via the terms that occur in the documents, and this pattern will resemble a band in the data matrix. Other natural interpretations of a banded structure include overlapping communities in social networks, overlapping roles of genes in various diseases, and patterns of species occurring in spatially correlated sites [19, 12, 15].

This paper presents the novel MMBS algorithm for detecting banded structures and/or approximate banded structures in subspaces of binary datasets. Our algorithm allows for the discovery of multiple, possibly overlapping or segmented, maximally banded sub-matrices from the original data matrix. This differs from the previously proposed MBS [12] algorithm which only allows for the discovery of a single band and fixes the column permutations of the data matrix before executing the algorithm. In addition, we formally illustrate the correspondence between the well studied problem of bi-clustering and the problem of discovering banded structures. As far as we know, no other work has pointed out and studied the relation between these two seemingly disjoint problems. Thus the main contributions of this paper are:

1. Establishing correspondence between banded structures

*Computer Science Dept. University of Cincinnati

†Computer Science Dept. University of Cincinnati

‡Cincinnati Children's Hospital Medical Center

and bi-clustering in binary datasets.

2. Introducing the novel MMBS algorithm to uncover multiple, possibly overlapping, banded sub-matrices.
3. Empirical results verifying the advantage of MMBS over previous approaches.

The rest of the paper is organized as follows: Section 2 formally defines the banded matrix problem, section 3 conveys the relationship between bi-clustering and banded structures and the next two sections present the MMBS algorithm followed by experimental results on synthetic and real data. Finally, section 6 discusses related work, followed by concluding remarks.

2 Problem Definition

Consider a binary matrix K , with row labels in the set G and column labels in the set M . In the documents-keywords example given earlier the elements of G would be documents and those of M would be the keywords. Without loss of generality the elements of G and M can be mapped to consecutive integers $\{1, 2, \dots, |G|\}$ and $\{1, 2, \dots, |M|\}$ and K may be represented by a **context** (as per the theory of Formal Concept Analysis [11]) $\mathbb{K} = (G, M, I)$, where I is a relation such that if gIm then $M(g, m) = 1$ and zero otherwise. Throughout the rest of the paper we will use the symbols K and \mathbb{K} interchangeably. We denote the i -th row of \mathbb{K} by \mathbb{K}^i and the j -th column by \mathbb{K}_j . Given a permutation π of G , and permutation τ of M , then \mathbb{K}_τ^π is the permutation of rows and columns according to π and τ . We will use $g^{\pi i}$ to denote the i -th row and $m_{\tau j}$ to denote the i -th row and j -th column respectively under permutations π and τ .

DEFINITION 1. A binary matrix $\mathbb{K} = (G, M, I)$ is **fully banded** if there exists a permutation π of G and permutation τ of M such that (1) for every row i in \mathbb{K}_τ^π the entries with 1s occur in consecutive column indices $\{m_i, m_i + 1, \dots, m_i^*\}$ and (2) the values of starting indices for 1s in successive rows (i and $i + 1$) satisfy the conditions $m_i \leq m_{i+1}$ and $m_i^* \leq m_{i+1}^*$. An illustration of this is given in figure 1.

Testing a given matrix to find whether it is banded can be accomplished in polynomial time [12]. In real datasets a fully banded structure may not exist due to noise or irrelevant dimensions, however, we are still interested in discovering 1) approximately banded matrices and 2) maximally banded sub-matrices of the dataset. A maximally banded sub-matrix of a matrix, intuitively, is one in which no more rows from the original matrix can be added while still preserving the bandedness of the selected sub-matrix. The quality of a banded structure can be measured in terms of noise, where noise is the minimum number of 0s or 1s that must be flipped in order to achieve a fully banded matrix. Given an approximately banded matrix \mathbb{K}_τ^π , let $e(\mathbb{K}_\tau^\pi)$ denote the noise

or error of the banded structure in \mathbb{K}_τ^π . We say that \mathbb{K}_τ^π is ϵ -banded if $e(\mathbb{K}_\tau^\pi) \leq \epsilon$.

PROBLEM 1. Given binary matrix \mathbb{K} and noise threshold ϵ , find all sub-matrices \mathbb{K} of \mathbb{K} that are ϵ -banded and maximal.

Unveiling all maximal ϵ -banded sub-matrices is useful in datasets that contain banded structures in certain subsets of dimensions along with possibly independent or segmented bands; *segmented bands* are the situations in which cells around multiple, non-principal diagonals are populated by 1s. Moreover, relaxing the requirements from full bandedness to ϵ -bandedness allows for clear band structures to be identified despite the presence of noise. The algorithmic complexity of this problem is expected to be hard since it is a generalization of both the MBA and MBS problems given in [12], which were both shown to be hard.

3 Bandedness and Bi-clustering

Bi-clustering (subspace clustering, co-clustering, closed itemsets, maximal bicliques) in binary data has been extensively studied [8, 10, 13, 18, 3, 11]. In these works the model for the subspace cluster is derived either from graph theory or Formal Concept Analysis (FCA) [11] (both models are equivalent but have different roots). Under both models, clusters are viewed as maximal rectangles of 1s in the matrix under a suitable permutation of the rows and columns. Such maximal rectangles encode tight correlations between the sets of rows and columns that they encompass; however, clearly banded patterns cannot be detected under this model and thus these types of patterns may represent a limitation of the traditional bi-clustering scheme. On the other hand, we show in this section that FCA theory can in-fact be utilized as a building block to discover maximally ϵ -banded sub-matrices.

3.1 Formal Concept Analysis The row and column vectors of \mathbb{K} can be interpreted as sets of indices, that is, a row \mathbb{K}^i is a set of column indices that appear in the row. Given this interpretation, FCA [11] defines operators over sets of row and column vectors. Given, $\mathbb{K} = (G, M, I)$ and $A \subseteq G$, then we define $A' = \{m \in M | gIm \text{ for all } g \in A\}$. Also for $B \subseteq M$, we have $B' = \{g \in G | gIm \text{ for all } m \in B\}$. A **formal concept** or **bi-cluster** of \mathbb{K} is then a pair (A, B) such that $A' = B$ and $B' = A$. In FCA elements of G and A are referred to as **objects** and **attributes** respectively. Noticeably, the bi-clusters of \mathbb{K} can be represented by a maximal rectangle of 1s under suitable permutations of the rows and columns of \mathbb{K} (figure 2(b)). Given bi-clusters, (A_1, B_1) (A_2, B_2) , we may order them as $(A_1, B_1) \leq (A_2, B_2)$ provided that $A_1 \subseteq A_2$ (equivalently $B_2 \subseteq B_1$). Bi-cluster C_1 is an upper neighbor of C_2 if $C_1 \leq C_2$ and there does not exist a bi-cluster C_3 s.t. $C_1 \leq C_3 \leq C_2$. We denote this by $C_1 \prec C_2$. The set of all bi-clusters ordered in this way

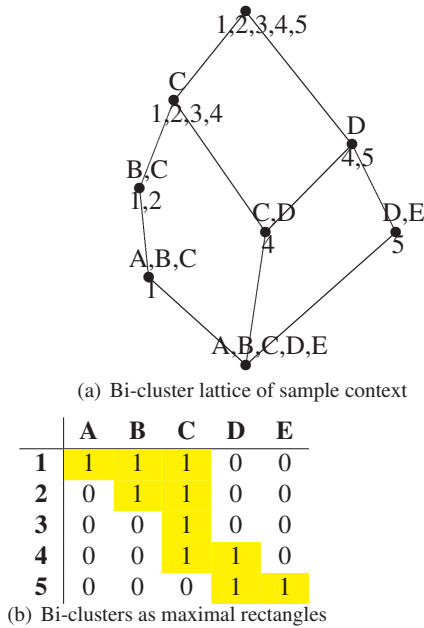


Figure 2: Bi-clusters and maximal rectangles

is denoted by $\mathfrak{B}(G, M, I)$ and is called the **bi-cluster lattice** of \mathbb{K} ; it contains all the bi-clusters of \mathbb{K} arranged in a lattice structure. The basic theorem of FCA states that the bi-cluster lattice $\mathfrak{B}(G, M, I)$ is a complete lattice.

Given any bi-cluster $C = (A, B)$ of a context \mathbb{K} the set of rows (columns) specified by A (B) clearly satisfy both conditions of a banded matrix (every element of a bi-cluster is a 1). Thus C is a banded sub-matrix of \mathbb{K} , and can be utilized as a building block to construct larger bands (in larger submatrices). Intuitively, any fully banded matrix can be splintered exactly into maximal rectangles of 1s, as shown in figure 2(b). Each of these rectangles corresponds exactly to a bi-cluster. Thus, a fully banded sub-matrix may be constructed by combining a suitably selected sequence of bi-clusters C_1, \dots, C_n . Formally, given a fully banded matrix, \mathbb{K}_τ^π , for any row $g \in \mathbb{K}_\tau^\pi$, let $\Gamma(g)$ be a mapping from g to the set of bi-clusters that contain g as a row, other than the null-bi-cluster element.

$$\Gamma(g) = \{(A, B) | \{g\} \subseteq A \wedge B \neq \emptyset \wedge A' = B \wedge B' = A\}$$

Moreover the objects and attributes of any bi-cluster $C \in \Gamma(g)$ can always be ordered according to π and τ due to the fact that a bi-cluster only contains 1s. Let $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ be the union of all $\Gamma(g)$ for any $g \in G$, that is

$$\mathfrak{F}(\mathbb{K}_\tau^\pi) = \bigcup_{g \in G} \Gamma(g)$$

Then $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ can be ordered to make an n -tuple of bi-clusters $\{C_1, \dots, C_n\}$ having a total ordering $\{<_{\pi_1, \tau_1}, \dots, <_{\pi_n, \tau_n}\}$

as determined from the lattice structure. Thus we may define a lexicographical order $<^{\pi, \tau}$ on $C_1 \times C_2 \times \dots \times C_n$.

Therefore, considering the bi-clusters in $\mathfrak{F}(\mathbb{K}_\tau^\pi)$, in order, we may completely specify the permutations π and τ ; hence $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ constitutes a sequence of bi-clusters that completely determines the banded structure of \mathbb{K} .

PROPOSITION 3.1. *Given a context \mathbb{K} , if permutations π and τ exist such that \mathbb{K}_τ^π is fully banded then there exists a sequence of bi-clusters $C_1 = (A_1, B_1), \dots, C_n = (A_n, B_n)$ s.t.*

$$\begin{aligned} \pi &= \{A_1, A_2 \setminus A_1, \dots, A_n \setminus A_{n-1}\} \\ \tau &= \{B_1 \setminus B_2, \dots, B_{n-1} \setminus B_n, B_n\} \end{aligned}$$

where $A_2 \setminus A_1$ is the set difference.

The proof of proposition 3.1 is straightforward: C_1, \dots, C_n can always be constructed by considering the bi-clusters of $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ in order, while set differences are taken to eliminate duplicate rows and columns. Proposition 3.1 establishes a clear correspondence between fully banded matrices and FCA; **specifically, if a matrix has a fully banded structure then there always exists a sequence of bi-clusters that stipulates its π and τ .**

EXAMPLE 1. *Consider the sample context in figure 1. The permutations are $\pi = \{1, 2, 3, 4, 5\}$ and $\tau = \{A, B, C, D, E\}$, therefore, lexicographically, $1 < 2 < 3 < 4 < 5$ and $A < B < C < D < E$. The table below illustrates $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ and the resulting lexicographical ordering.*

g	$\Gamma(g)$
1	$\{(1, ABC), (12, BC), (1234, C)\}$
2	$\{(12, BC), (1234, C)\}$
3	$\{(1234, C)\}$
4	$\{(4, CD), (45, D)\}$
5	$\{(5, DE), (45, D)\}$
$\mathfrak{F}(\mathbb{K}_\tau^\pi)$	
$\{(1, ABC) < (12, BC) < (1234, C) < (4, CD) < (45, D) < (5, DE)\}$	

π and τ can be constructed from $\mathfrak{F}(\mathbb{K}_\tau^\pi)$ as

$$\begin{aligned} \pi &= \{1, 12 \setminus 1, \dots, 5 \setminus 45\} \\ &= \{1, 2, 3, 4, 5\} \\ \tau &= \{ABC \setminus BC, \dots, D \setminus DE, DE\} \\ &= \{A, B, C, D, E\} \end{aligned}$$

In the next section we show that a banded structure can be grown as a path of bi-clusters in the bi-cluster lattice. In addition, we derive an expression for the upper bound of the error when constructing such banded sub-matrices.

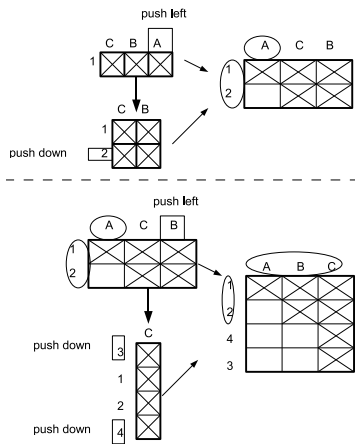


Figure 3: Constructing banded matrices from paths in \mathbb{G}

3.2 Banded sub-matrices and paths in the lattice The bi-cluster lattice may be viewed as an undirected graph $\mathbb{G} = (V, E)$. The set of bi-clusters corresponds to the set of vertices V , and the set of edges E consists of edges that connect pairs of bi-clusters - upper neighbors to lower neighbors. For a pair of neighbor bi-clusters we can say: $C_1, C_2 \in E \leftrightarrow C_1 \prec C_2 \vee C_2 \prec C_1$. Let $\bar{P} = C_1, C_2, \dots, C_n$ be any path in the lattice \mathbb{G} , then by definition, for every edge $(C_i, C_{i+1}) \in \bar{P}$, $A_{i+1} \subseteq A_i$ and $B_i \subseteq B_{i+1}$ if $C_i \prec C_{i+1}$ (dually $A_i \subseteq A_{i+1}$ and $B_{i+1} \subseteq B_i$ if $C_i \succ C_{i+1}$). Due to the duality of upper and lower neighbors, we restrict the discussion to the case of upper neighbors only without any loss of generality. A banded sub-matrix can be constructed from any edge of the lattice by initiating the matrix to be exactly C_i , by definition every position in this matrix is filled with a 1. Next we place the rows of $A_{i+1} \setminus A_i$ beneath the rows of A_i and shift the columns where A_i and $A_{i+1} \setminus A_i$ differ, to the left. By the definitions of the hierarchal order and bi-clusters, the 0s are located precisely at rows $A_{i+1} \setminus A_i$ and columns $B_i \setminus B_{i+1}$. Consequently, any edge in \mathbb{G} can be converted into a fully banded sub-matrix.

For example, consider the edge $((1, ABC), (12, BC))$ (figure 3), the two bi-clusters differ by column $\{A\}$ and row $\{2\}$, thus $\{A\}$ is placed at the leftmost position and $\{2\}$ at the bottom-most position. At this point the positions of rows $\{1\}, \{2\}$ and column $\{A\}$ are fixed and cannot be altered, while the position of columns $\{B\}$ and $\{C\}$ are interchangeable. Thus we obtain the banded sub-matrix with permutations $\pi = \{1, 2\}$ and $\tau = \{A, C, B\}$ or $\tau = \{A, B, C\}$. Next, the path is further expanded by adding the edge $((12, BC), (1234, C))$, resulting in shifting column $\{C\}$ to the right and rows $\{3\}$ and $\{4\}$ to the bottom. This fixes the positions of all the columns but

leaves the positions of rows $\{3\}$ and $\{4\}$ interchangeable. In general, the procedure for augmenting a path \bar{P} with an edge (C_n, C_{n+1}) and maintaining the banded structure is described by the `AddToPath` procedure displayed below. Notice that the symmetric difference set operator is used, as opposed to regular set difference, due to the fact that upper or lower neighbors may be added to the path. The procedure determines if the new bi-cluster C_{n+1} is an upper or lower neighbor of C_n and if the new rows or columns of C_{n+1} are not already contained in \bar{P} (lines 4 and 10). If this is the case then the new rows \hat{A} are added to the bottom of the band, as signified by the \uplus symbol (line 5), and the differing columns are moved to the leftmost positions within the interchangeable block that they occur in (lines 6-9). This can be easily accomplished if every path maintains integers x and y indicating the position of fixed blocks of columns or rows. Lines 10-15 implement the exact same procedure, however the rows and columns are reversed because a lower neighbor is added to the path as opposed to an upper neighbor.

```

Input:  $\bar{P} = C_1, C_2, \dots, C_n$ 
Data:  $\mathbb{A} = \bigcup_{j=1}^n A_j, \mathbb{B} = \bigcup_{j=1}^n B_j$ 
Data:  $x, y$  : position of fixed column and row indices
Input: new bi-cluster  $C_{n+1}$ 
1 begin
2    $\hat{A} \leftarrow A_{n+1} \ominus A_n$ ;
3    $\hat{B} \leftarrow B_{n+1} \ominus B_n$ ;
4   if  $A_{n+1} \supset \mathbb{A}$  then
5      $\pi \leftarrow \pi \uplus \hat{A}$ ;
6     for  $b \in \hat{B}$  do
7       if  $pos(b) > x$  then
8         swap element at pos  $x$  in  $\tau$  with  $b$ ;
9          $x \leftarrow x + 1$ ;
10    else if  $B_{n+1} \supset \mathbb{B}$  then
11       $\tau \leftarrow \tau \uplus \hat{B}$ ;
12      for  $a \in \hat{A}$  do
13        if  $pos(a) > y$  then
14          swap element at pos  $y$  in  $\pi$  with  $a$ ;
15           $x \leftarrow x + 1$ ;
16 end

```

Procedure AddToPath

The `AddToPath` procedure illustrates how a path in \mathbb{G} can be mapped to a sub-matrix of \mathbb{K} that is at least partially banded; hence we refer to a sub-matrix $\widehat{\mathbb{K}}_{\pi}^{\tau}$ and a path \bar{P} interchangeably. Due to the fact that each individual edge in \bar{P} is guaranteed to produce a banded structure, the only

possible errors when adding a concept $C_{n+1} \succ C_n$ to \bar{P} occurs if a newly added row $a \in \hat{A}$ contains a column index b such that $pos(b) < x$, where x indicates the position of the fixed block of columns.

PROPOSITION 3.2. *Let $\bar{P}_{n-1} = \mathbf{C}_1, \dots, \mathbf{C}_n$ be a path constructed by repeated calls to procedure `AddToPath` with associated permutations π, τ and integers x and y . Given bi-cluster C_{n+1} s.t. C_{n+1} is augmented to \bar{P} via `AddToPath` then*

$$e(\bar{P}_n) \leq \begin{cases} 0 & \text{if } n \leq 1 \\ e(\bar{P}_{n-1}) + \sum_{a \in \hat{A}} |a' \cap \mathbb{B}| & \text{if } \mathbf{C}_{n+1} \succ \mathbf{C}_n \\ e(\bar{P}_{n-1}) + \sum_{b \in \hat{B}} |b' \cap \mathbb{A}| & \text{if } \mathbf{C}_{n+1} \prec \mathbf{C}_n \end{cases}$$

where $\hat{A} = A_{n+1} \ominus A_n$, $\hat{B} = B_{n+1} \ominus B_n$, $\mathbb{A} = \bigcup_{j=1}^y a^{\pi_j}$ and

$$\mathbb{B} = \bigcup_{j=1}^x b_{\tau_j}$$

Proof. We prove this by induction on n .

Base case, $n \leq 1$: Only one edge $\mathbf{C}_1, \mathbf{C}_2$ exists in the path, and without loss of generality we assume $\mathbf{C}_1 \prec \mathbf{C}_2$. In this case

$$\begin{aligned} \pi &= \{A_1, A_2 \ominus A_1\} \\ &= \{a^{\pi_1}, \dots, a^{\pi_{|A_1|}}, a^{\pi_{|A_1|+1}}, \dots, a^{\pi_{|A_1|+|A_2 \ominus A_1|}}\} \end{aligned}$$

and

$$\begin{aligned} \tau &= \{B_1 \ominus B_2, B_2\} \\ &= \{b_{\tau_1}, \dots, b_{\tau_{|B_1 \setminus B_2|}}, b_{\tau_{|B_1 \setminus B_2|+1}}, \dots, b_{\tau_{|B_1 \setminus B_2|+|B_2|}}\} \end{aligned}$$

By definition of a concept, rows $a^{\pi_1}, \dots, a^{\pi_{|A_1|}}$ contain 1s in all the columns $b_{\tau_1}, \dots, b_{\tau_{|B_1|}}$. By definition of the hierarchical order and set difference, the remaining rows $a^{\pi_{|A_1|+1}}, \dots, a^{\pi_{|A_1|+|A_2 \ominus A_1|}}$ contain zeros at precisely $b_{\tau_1}, \dots, b_{\tau_{|B_1 \setminus B_2|}}$ and ones at $b_{\tau_{|B_1 \setminus B_2|+1}}, \dots, b_{\tau_{|B_1 \setminus B_2|+|B_2|}}$. Thus by definition of banded matrices \bar{P} is fully banded and $e(\bar{P}_1) = 0$.

Inductive step: Assume true for paths up to length n
After adding concept C_{n+1} the procedure only re-arranges column indices that are non-fixed and thus inter-changeable (lines 7-9) therefore not effecting the error. Moreover, every newly added row $a \in A_{n+1} \ominus A_n$ contains 1s in exactly columns B_{n+1} which is a proper subset of B_n by the definition of the hierarchical order. Thus $\hat{B} = B_n \setminus B_{n+1}$ and all columns $b_{\tau_{x+1}}, \dots, b_{\tau_{|\tau|}}$ contain 1s which cause no errors. Hence by definition of banded matrices, only 1s occurring in a newly added row $a \in A_{n+1} \setminus A_n$ and a column $b_{\tau_1}, \dots, b_{\tau_x}$ can cause additional error as a result of adding

\mathbf{C}_{n+1} . These 1s can be precisely counted as $\sum_{a \in \hat{A}} |a' \cap \mathbb{B}|$, so by the inductive theorem

$$(3.1) \quad e(\bar{P}_{n+1}) \leq e(\bar{P}_n) + \sum_{a \in \hat{A}} |a' \cap \mathbb{B}|$$

4 MMBS Algorithm

Aggregating propositions 3.1, 3.2 and the `AddToPath` procedure we have developed the MMBS (Mine Maximally Banded Sub-matrix) algorithm. Proposition 3.1 stipulates that any banded sub-matrix of \mathbb{K} can be enumerated as a sequence of bi-clusters of \mathbb{K} . Moreover, invoking `AddToPath` repeatedly provides a mechanism to construct a band from a path in the lattice of bi-clusters, and proposition 3.2 is utilized to compute an upper bound on the error. Consider the graph \mathbb{G} of a bi-cluster lattice $\mathfrak{B}(G, M, I)$, then any edge $(\mathbf{C}_i, \mathbf{C}_j)$ can be weighted by the amount of error that would be introduced if \mathbf{C}_j is added to the path $(\bar{P}) = \mathbf{C}_1, \dots, \mathbf{C}_i$. **Thus, identifying maximally ϵ -banded sub-matrices is equivalent to identifying all maximal paths in \mathbb{G} with total weight less than ϵ .** This problem differs from the all-pairs shortest path problem due to the fact that the edge weights are clearly non-constant; to the contrary, the edge weights are a function of all previously added edges along the current path \bar{P} . The MMBS algorithm consists of three major steps: 1) computing $\mathfrak{B}(G, M, I)$ and \mathbb{G} , 2) searching the paths of $\mathfrak{B}(G, M, I)$, and 3) determining the top banded sub-matrices to output to the user.

4.1 Compute $\mathfrak{B}(G, M, I)$ Computing the bi-cluster lattice of a context has been widely studied and numerous algorithms exist to accomplish this task [11, 17, 13, 16, 20]. These algorithms can be either incremental or non-incremental. The incremental algorithms ([17, 7, 16]) compute the lattices one bi-cluster at a time, by determining the upper and lower neighbors of any given bi-cluster. Thus, with these algorithms the task of computing $\mathfrak{B}(G, M, I)$ can be embedded directly into the mining process. On the other hand, non-incremental algorithms such as those described in [20, 16] do not attain the full and correct lattice structure until the termination of the algorithm implying that these algorithms cannot be directly embedded into MMBS; however, the computation time of the CHARM-L algorithm [20] is significantly lower than all other approaches. Thus in our implementation of MMBS we utilized CHARM-L to compute $\mathfrak{B}(G, M, I)$. Note, however, that any of the algorithms mentioned above will suffice and in all subsequent descriptions of the MMBS we assume that $\mathfrak{B}(G, M, I)$ and therefore \mathbb{G} is readily available.

4.2 Search Space MMBS conducts a depth first search with backtracking in order to identify maximal ϵ -banded paths. Undoubtedly, in the worst case the search space is

exponential in the size of the lattice, due to the number of potential paths. Fortunately, the error associated with a path in the lattice grows monotonically allowing the search to prune entire branches of the space whenever an edge that results in error greater than ϵ is encountered. Despite such pruning, the task of searching all paths is still indeed intractable. We present two heuristic arguments to make the problem approachable. First, let \mathbb{U} be the set of upper neighbors of the bottom element in $\mathfrak{B}(G, M, I)$, then each path is rooted at a bi-cluster $\mathbf{C} \in \mathbb{U}$. In essence, this imposes a slight restriction on the possible row orderings since every band will begin with the rows contained in \mathbb{U} . However, it is these rows precisely that contain the largest sets of column indices, therefore allowing the greatest freedom to swap these indices as more bi-clusters are added to the path. Next, we set each vertex to remember the minimum weight edge encountered throughout the search. Due to the monotonicity of the error measure, nodes that have been previously visited are only added to a new path if and only if the newly computed error is less than or equal to the minimum weight edge encountered previously in the search.

4.3 Determining top bands Identifying all maximal ϵ -banded paths in the lattice is not very useful to the user, as there may exist an explosive number of such paths. In order to determine the most interesting banded sub-matrices we allow several parameters to be set in addition to ϵ . First, *minRows* and *minCols* parameters determine the minimum number of rows and columns any band should contain, and we use the parameter *w* to specify the relative weight that should be assigned to the error in a banded structure. Next *maxOvlp* specifies the maximum overlap allowed between any two bands. Computing the overlap between any two bands consists of computing the Jaccard coefficient of both the rows and columns and weighing each. Given path $\bar{P} = \mathbf{C}_1, \dots, \mathbf{C}_n$. Let $r(\bar{P})$ denote the rows of \bar{P} and $c(\bar{P})$ the columns, then $ovlp(\bar{P}_1, \bar{P}_2)$ is:

$$c * J(r(\bar{P}_1), r(\bar{P}_2)) + (1 - c) * J(c(\bar{P}_1), c(\bar{P}_2))$$

where $0 \leq c \leq 1$ and $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$. Finally, a quality measure q is needed to rank candidate ϵ -banded paths that satisfy all other user parameters. At an intuitive level, the quality of a band should be monotonic in the size of the band and penalized for errors. Following this intuition we utilized the simple quality measure

$$(4.2) \quad q(\bar{P}) = |r(\bar{P})| * |c(\bar{P})| - w * e(\bar{P})$$

4.4 Pseudocode and complexity Pseudo code of MMBS and the search procedure appears as algorithm 2 and procedure *Search*. We focus on the *Search* procedure as most of the work is conducted there. The first *for* loop on line 4 iterates through all possible concepts that the current

```

Input:  $\mathfrak{B}(G, M, I), \mathbb{G}$ 
Input:  $\epsilon, w, minRows, minCols, maxOvlp$ 
1 begin
2    $\mathcal{W}[] \leftarrow \infty;$ 
   /* keep track of min error */
3    $\mathcal{R}[] \leftarrow \emptyset;$ 
   /* store bands */
4   foreach  $\mathbf{C} \in \mathbb{U}$  do
5     initiate path  $\bar{P}$  to  $\mathbf{C}$ ;
6     Search( $\bar{P}$ );
7 end

```

Algorithm 2: MMBS

```

1 begin
2    $expand \leftarrow false;$ 
3    $\bar{P}_{copy} \leftarrow \bar{P};$ 
4   foreach  $\hat{\mathbf{C}} \succ \mathbf{C}_n \vee \hat{\mathbf{C}} \prec \mathbf{C}_n$  do
5     if  $\hat{\mathbf{C}} \notin \bar{P} \wedge (\hat{A} \supset r(\bar{P}) \vee \hat{B} \supset c(\bar{P}))$  then
6        $x \leftarrow e(\bar{P}) - \epsilon(\bar{P} \cup \hat{\mathbf{C}});$ 
7       if  $e(\bar{P} \cup \hat{\mathbf{C}}) \leq \epsilon \wedge \mathcal{W}[\hat{\mathbf{C}}] \geq x$  then
8          $\mathcal{W}[\hat{\mathbf{C}}] \leftarrow x;$ 
9         AddToPath( $\bar{P}$ );
10        Search( $\bar{P}$ );
11         $\bar{P} \leftarrow \bar{P}_{copy};$ 
12         $expand \leftarrow true;$ 
13  if  $!expand$  then
14    if
15       $|r(\bar{P})| \geq minRows \wedge |c(\bar{P})| \geq minCols$ 
16      then
17        if  $max\ ovlp(\bar{P}, \bar{P}_i \in \mathcal{R}) < maxOvlp$ 
18        then
19           $\mathcal{R} \leftarrow \mathcal{R} \cup \bar{P};$ 
20        else
21          Let  $\bar{P}_x$  be path s.t.
22             $ovlp(\bar{P}_x, \bar{P}) \geq maxOvlp;$ 
23            if  $q(\bar{P}) > q(\bar{P}_x)$  then
24               $\mathcal{R} \leftarrow \mathcal{R} \cup \bar{P};$ 
25  end

```

Procedure Search(\bar{P})

path \bar{P} can expand to. If the error of augmenting the path with candidate concept $\hat{\mathbf{C}}$ is less than ϵ and at no more than any previous edge weight, then the best edge weight of $\hat{\mathbf{C}}$ is updated and the path is expanded to include $\hat{\mathbf{C}}$ (utilizing *AddToPath*). The search delves deeper in order to attempt to maximize the path (line 10), while the copy on line 11

implements backtracking. If the *expand* flag is never set to true then the current path \bar{P} cannot be further expanded and the algorithm must evaluate if \bar{P} meets all the user defined parameters. If \bar{P} meets the *minRows* and *minCols* parameters, then it is compared to all previously mined bands to determine its degree of overlap (line 15). If the overlap between \bar{P} and all other bands \bar{P}_i does not exceed *maxOvlp* then it is added to \mathcal{R} . On the other hand, if the overlap of \bar{P} and \bar{P}_i does exceed the threshold, then the higher quality band is kept.

The three basic operations of MMBS are set difference, set intersection and swap operations. Let $X = \max |A_i \ominus A_j|$ for all $A_i \in \mathbf{C}_i, A_j \in \mathbf{C}_j$ s.t. $\mathbf{C}_j \succ \mathbf{C}_i$. Also let $Y = \max |B_i \ominus B_j|$ for all $B_i \in \mathbf{C}_i, B_j \in \mathbf{C}_j$ s.t. $\mathbf{C}_j \prec \mathbf{C}_i$. Augmenting a concept to a path utilizing the `AddToPath` procedure involves two set differences and at most $O(X)$ or $O(Y)$ swaps, while computing the error on line 6 involves at most $O(X)$ or $O(Y)$ set intersections. Clearly, `AddToPath` is invoked at least as many times as the error is computed; however the number of times this occurs is difficult to analyze, as it depends on several factors such as ϵ , and the actual structure of the lattice. Nevertheless, the error will never be computed more than $O(|E|)$ times due to the fact that an edge is never visited more than once. Lastly, the search procedure invoked $|\mathbb{U}|$ times, thus the total cost of MMBS is

$$(4.3) \quad O(|\mathbb{U}| \times |E| \times \max\{X, Y\})$$

The memory footprint of MMBS is minimal, only a single path needs to be maintained in memory throughout the search, while \mathbb{G} can be maintained on disk or in main memory. Moreover, our experimental results indicate that it is very plausible to maintain \mathbb{G} in the main memory comfortably for even moderately large matrices (4000×4000) as 0-1 datasets tend to be sparse.

4.4.1 Speeding up the algorithm The dominant term in equation 4.3 is clearly $|E|$, while if $|\mathbb{U}|$ is large then the computational cost is quite expensive. Unfortunately, many of the bi-clusters in $|\mathbb{U}|$ tend to be very similar, and the `Search` procedure often does not yield any new bands, but consumes computation time. Thus, in order to speed up the algorithm, we introduce a pre processing step to eliminate similar concepts and reduce the number of times the `Search` procedure is invoked. Specifically, $ovlp(\mathbf{C}_1, \mathbf{C}_2)$ is computed for all bi-clusters $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{U}, \mathbf{C}_1 \neq \mathbf{C}_2$, if $ovlp(\mathbf{C}_1, \mathbf{C}_2) > maxOvlp$ then the larger concept is kept. In all performance tests, this pre-processing step accelerated the computation time dramatically (see next section) while producing very comparable results to our original MMBS algorithm.

5 Experimental Results

In this section we illustrate the efficacy of MMBS experimentally. The performance of MMBS is compared with the MBS algorithm proposed in [12]. We show that MMBS consistently outperforms MBS in three different ways:

1. MMBS uncovers several banded structures as opposed to a single band mined by MBS.
2. MMBS consistently uncovers higher quality bands.
3. More scalable computation time.

These results are exhibited with both synthetic and real world datasets. MMBS was implemented in C++ utilizing the STL data structures and is available at <http://homepages.uc.edu/~alqadaf>. All experiments were conducted on a 2.7 GhZ AMD Athlon 64 x2 CPU with 5.8 Gb of RAM, running Ubuntu Linux. Additionally, `MMBS_Fast` was implemented, in which the pre-processing step described above was applied. Source code for MBS was kindly provided by the authors of [12].

5.1 Synthetic datasets Several synthetic datasets were created to test the capability of MMBS. Three different methods were utilized to create three different classes of datasets. First, single band datasets were created utilizing the method described in [12]. Briefly, for a given number of rows ($|G|$), columns ($|M|$) and width parameter w_i a fully banded matrix is generated by means of a random walk starting at the (0, 0) coordinate. Initially all cells in the matrix are set to 0, and the walk chooses to either step down or to the right with equal probability. On a step to the right the w_i cells above and below the current position are all set to 1s. Noise can be added to the band by flipping the original values to 0 or 1 with probabilities p and q . Next, multiple band matrices were created by first producing several single band matrices K_1, \dots, K_n , followed by "concatenating" the single bands into a single "switch matrix" as:

$$\begin{pmatrix} \dots & \dots & 0 & K_1 \\ \dots & \dots & K_2 & 0 \\ 0 & \dots & 0 & \vdots \\ K_n & 0 & \dots & \vdots \end{pmatrix}$$

Finally, random binary matrices with a set sparsity level were also created.

Three sets of experiments were conducted on each class of synthetic data. All experiments were conducted with $w = 1, maxOvlp = 0.1, minRows = minCols = 5$, and $\epsilon = 99$ (the maximum allowed misplaced 1s or 0s was 99). The `MBS_BD` algorithm allows for bi-directional flipping of both 0s to 1s and 1s to zeros, while `MBS_SD` only allows for flipping 1s to 0s. Several possible initial

Dataset name	Dataset Size	p	Num. Planted bands	Algorithm	Quality top ranked	Num. bands mined
SynBand100_001	100 × 100	0.01	1	MMBS	3590	6
				MMBS_Fast	3406	4
				MBS_BD	2507	1
				MBS_SD	438	1
SynBand100_005	100 × 100	0.05	1	MMBS	2278	9
				MMBS_Fast	1503	8
				MBS_BD	1050	1
				MBS_SD	1201	1
SynBand500_001	500 × 500	0.01	1	MMBS	8918	7
				MMBS_Fast	8261	6
				MBS_BD	2822	1
				MBS_SD	2145	1
SynMultiBand100_001	100 × 100	0.01	2	MMBS	3367	2
				MMBS_Fast	3367	2
				MBS	4101	1
				MBS_SD	4045	1
SynMultiBand100_001	100 × 100	0.05	2	MMBS	4054	2
				MMBS_Fast	3933	2
				MBS_BD	3910	1
				MBS_SD	3736	1
SynMultiBand500_001	500 × 500	0.01	2	MMBS	28242	8
				MMBS_Fast	21346	5
				MBS_BD	17498	1
				MBS_SD	430	1
SynRandom100_005	100 × 100	0.05	unknown	MMBS	3311	17
				MMBS_Fast	3220	14
				MBS_BD	2801	1
				MBS_SD	1949	1
SynRandom500_001	500 × 500	0.01	unknown	MMBS	18635	73
				MMBS_Fast	16163	64
				MBS_BD	16771	1
				MBS_SD	5229	1

(a) Experimental results on synthetic data

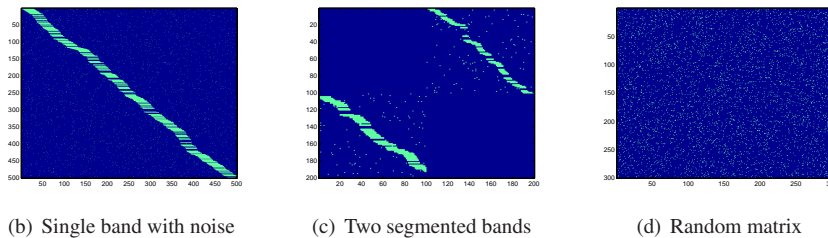


Figure 4: Synthetic data and experimental results

orderings are possible with the MBS algorithms, including Hamiltonian ordering with distance measures and spectral ordering with similarity measures; all orderings were utilized in the experiments and the best results are reported here for comparison.

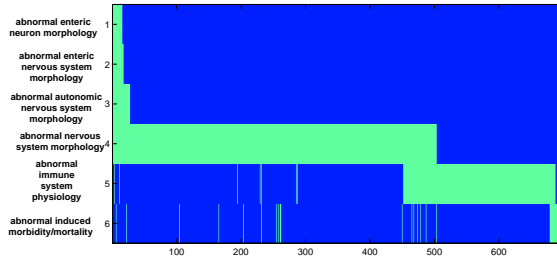
Figure 4(a) shows the results, and as can be seen MMBS and MMBS_Fast consistently discover higher quality bands. In the single band case, we see that MMBS is more resilient to noise and is able to discover a larger portion of the hidden band; this is a direct consequence of not fixing the column permutations. MBS_BD and MBS_SD discover bands of higher quality on two of the multi-band class datasets; however upon closer inspection we see that this is due to a failure of these algorithms to recognize that two segmented bands exist in the data, (as can clearly be seen from figure 4(c)). The MBS algorithms attempt to lump all the rows into one band, resulting in a larger band than any of the individual segmented bands, but does not complete the job and permute the matrix into a single band. In other words the band produced by MBS loses a very natural real world interpretation of two segmented bands. On the other hand, the MMBS algorithms were able to discover the two segments almost completely, both of which had very similar quality.

Moreover, in the largest of the multiple banded datasets the MMBS algorithm discovered both the segmented bands even though the quality of only one of them exceeds the quality of the single band mined by MBS. Finally, the random binary matrices represent the real world scenario of not knowing if a banded structure exists in the data. Once again MMBS algorithms outperform MBS and discovers larger bands, with the same limit of error allowed.

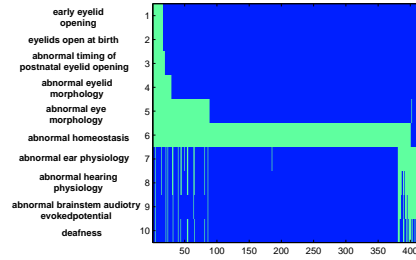
5.2 Real world datasets Four real world datasets from differing domains were utilized to examine the true utility of MMBS. The first two datasets, **Gene_Phenotypes** and **Genes_Drugs** came from the bioinformatics domain and are available publicly at <http://homepages.uc.edu/~alqadaf>. The rows of each matrix correspond to genes, while the columns to phenotypes and drugs respectively. The final two datasets were obtained from the large **News-Groups** dataset [4], with rows corresponding to usenet documents and columns to words extracted from the subject lines. **Mideast_Religion** contains documents found in both the Mideast and Religion groups, while **All_PC** contains usenet documents from all of the PC groups. All algorithms were executed with the same parameters discussed above, and the results appear in figure 5(a). Once again, we notice

Dataset	Size	Sparsity	Algorithm	Quality top ranked	Num. bands mined
Genes_Phenotypes	1910 × 3965	0.008	MMBS	6665	56
			MMBS_Fast	6665	43
			MBS_BD	5204	1
			MBS_SD	3578	1
Genes_Drugs	1608 × 49	0.042	MMBS	6423	18
			MMBS_Fast	6423	13
			MBS_BD	5346	1
			MBS_SD	3047	1
NewsGroups_Mideast_Religion	2000 × 890	0.003	MMBS	72906	42
			MMBS_Fast	61410	31
			MBS_BD	59781	1
			MBS_SD	58713	1
NewsGroups_AllPC	5000 × 2805	0.0001	MMBS	93368	5
			MMBS_Fast	93368	5
			MBS_BD	89106	1
			MBS_SD	74125	1

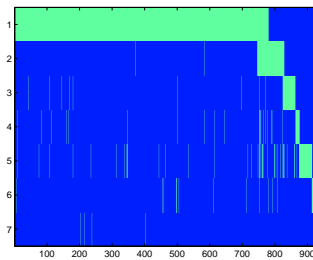
(a) Experimental results on real-world data



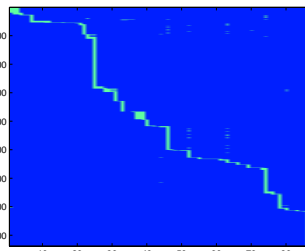
(b) Genes_Phenotypes



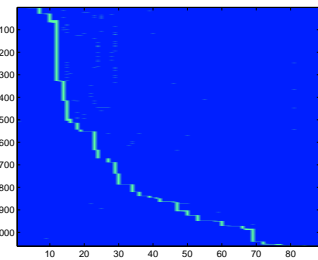
(c) Genes_Phenotypes



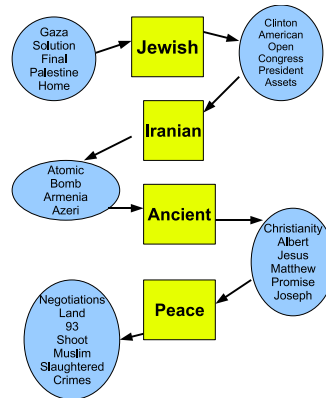
(d) Gene_Drugs



(e) NewsGroups_Mideast_Religion



(f) NewsGroups_AllPC



(g) Natural interpretation of band as overlapping communities

Figure 5: Real world datasets and mined sub-matrix bands

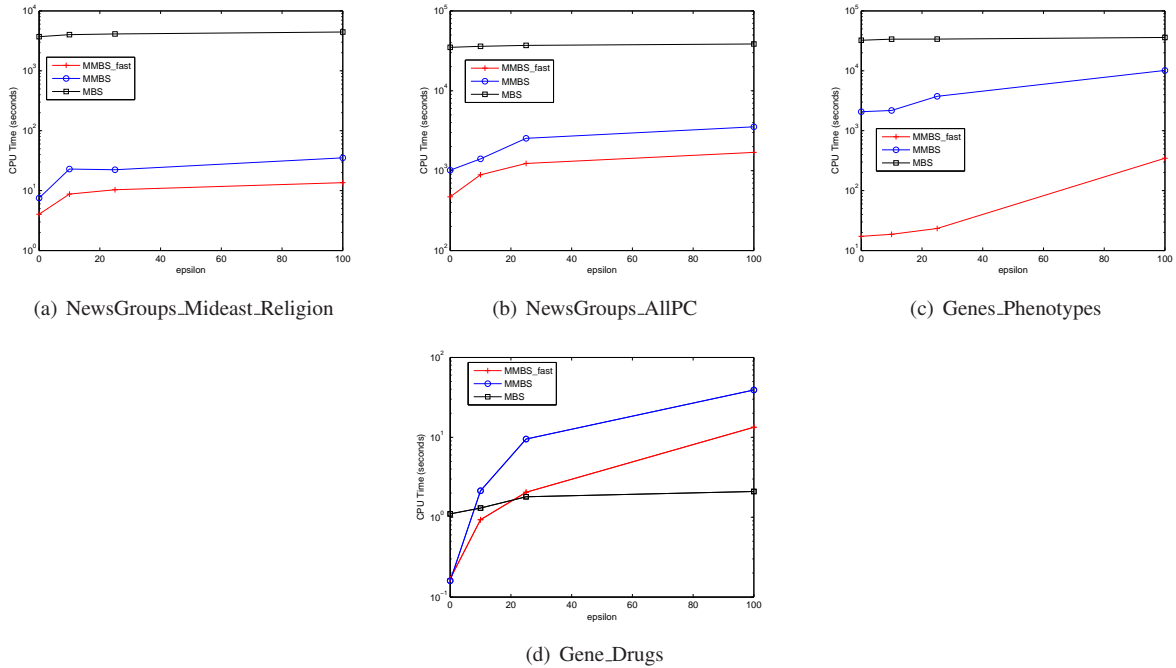


Figure 6: Performance Study

that MMBS algorithms not only consistently produce better quality bands, but also discover several banded structures in the data. Figure 5 illustrates the data matrices and the resulting bands discovered by MMBS. Investigating figures 5(b) and 5(c) reveals the advantage of mining multiple banded sub-matrices as opposed to a single band. In figure 5(b) the phenotypes all correspond to abnormal nervous system phenomenon; on the other hand, in figure 5(c) the first five phenotypes are abnormalities in the eyes and eyelids, while the last phenotypes indicate abnormalities in the ears. In both cases, the banded structure uncovers the overlapping roles of genes in different phenotypes, but the fact that they are two separate bands emphasizes the correlation between the phenotypes within each submatrix.

Utilizing the mined band from the **Mideast_Religion** we were able to construct a graph that successfully illustrated different themes discussed among usenet users (figure 5(g)). Specifically, collections of keywords and documents were constructed by following the permutations of the band until a row(s) or column(s) was encountered that was longer than average. All documents up to this point were placed into a collection (blue circles), while the longer than average row(s) or column(s) were placed in the yellow boxes with the bold font. For ease of interpretation we have only included keywords in the graph. At an intuitive level we can see that within each collection (circles) the keywords are highly correlated around a similar theme. At the same time, keywords

across collections are quite distinct, however, keywords in the boxes represent overlapping themes that links the two distinct collections of documents and keywords. Thus the banded structure carries with it a natural interpretation of the distinct discussion threads in the newsgroup, along with possible links between these distinct threads.

5.3 Performance Tests Performance tests to measure the practical computational cost of MMBS, MMBS_Fast, and MBS were conducted on the real world datasets. Each algorithm was executed ten times, while the ϵ -parameter was varied, and the average CPU times are reported in figure 6. We made use of the CHARM-L algorithm [20] to compute the bi-cluster lattice of each dataset and the cost of this computation is included in the results. Noticeably, the cost of MMBS_Fast is significantly lower than the other algorithms in three of the four datasets. In all three of these cases MMBS_Fast outperforms MMBS by an order of magnitude, and MBS by two orders of magnitude. The MBS algorithm proved not to be very sensitive to the ϵ parameter resulting in more efficient performances in smaller datasets such as the Gene_Drugs dataset. On the other hand, both MMBS and MMBS_Fast are very sensitive to ϵ . As ϵ increases, less pruning steps occur and the search procedure tends to the worst case cost of exploring all edges of the lattice. Despite this fact, both the MMBS algorithms scaled much better to the three larger datasets than MBS, as even the original MMBS

outperformed MBS by at least an order of magnitude in each case.

6 Related Work

The properties of banded matrices and how they relate to data analysis were first studied in [12]. The authors addressed the minimum banded augmentation (MBA) problem and the maximum banded submatrix (MBS) problem. The MBA problem is given a binary matrix K , find the minimum number of 0s that need to be modified into 1s so that K becomes fully banded. The MBS problem is given K and integer n find the maximum submatrix K' of K s.t. it is banded after n flips. The authors assume a fixed column permutation in the proposed solutions for both problems. While this is not a very realistic assumption in many real world scenarios, heuristic methods are proposed to determine a suitable fixed column permutation. The solution for the MBS problem builds upon the fixed-column algorithm utilized for MBA, and therefore also assumes fixed column permutations; moreover, only a single maximum submatrix is produced. Our work does not make any a-priori assumptions about the permutations, and moreover discovers multiple banded sub-matrices in the data.

In [19] the ecological concept of nestedness in binary data was introduced. A dataset is nested if for all pairs of rows one row is either a superset or subset of the other. It was shown in [12], however, that bandedness is a generalization of this ecological concept. In [14, 22] the authors establish a hierarchy between different classes of binary matrices. They consider banded matrices, zero partitionable matrices and nested matrices. It was shown that every banded matrix is a zero partition and they characterized how to determine if a zero partition contains a banded structure. In the numerical analysis field [21, 9, 5] work has focused on minimizing the distance of non-zero entries from the main diagonal of the matrix (bandwidth). This problem differs from the problem we addressed, as we attempt to discover several sub-matrices that have an approximate banded structure, as opposed to minimizing the bandwidth of the entire matrix.

7 Conclusion

In this work we explored the connection between bi-clustering and banded structures in binary data. It was shown that banded sub-matrices of a dataset correspond to paths in the bi-cluster lattice of that dataset. This correspondence formed the basis of the MMBS algorithm which discovers maximally ϵ -banded sub-matrices by exploring paths in the bi-cluster lattice. Experiments on synthetic and real-world datasets indicated three main advantages of MMBS over previous algorithms. First, multiple banded structures with natural interpretations are uncovered in the data as opposed to a single structure. Additionally, MMBS consistently mined higher quality bands, while the performance study illustrated

that the computational cost scaled better to larger datasets.

Future work will focus on more efficient methods of searching the bi-cluster lattice for banded structures through stronger bounding criterion on the error and more effective heuristics.

References

- [1] R. AGRAWAL, T. IMIELIŃSKI, AND A. SWAMI, *Mining association rules between sets of items in large databases*, in SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data, New York, NY, USA, 1993, ACM, pp. 207–216.
- [2] F. ALQADAH AND R. BHATNAGAR, *Detecting significant distinguishing sets among bi-clusters*, in CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management, New York, NY, USA, 2008, ACM, pp. 1455–1456.
- [3] F. ALQADAH AND R. BHATNAGAR, *Discovering substantial distinctions among incremental bi-clusters*, in SDM, 2009.
- [4] A. ASUNCION AND D. NEWMAN, *UCI machine learning repository*, 2007.
- [5] C. AYKANAT AND A. PINAR, *Permuting sparse rectangular matrices into block-diagonal form*, SIAM Journal on Scientific Computing, 25 (2004), pp. 1860–1879.
- [6] R. BAEZA-YATES AND B. RIBEIRO-NETO, *Modern Information Retrieval*, Addison Wesley, 1999.
- [7] P. BECKER, J. HERETH, AND G. STUMME, *Toscanaj - an open source tool for qualitative data analysis*, in Advances in Formal Concept Analysis for Knowledge Discovery in Databases. Proc. Workshop FCAKDD of the 15th European Conference on Artificial Intelligence (ECAI 2002). Lyon, France., V. Duquenne, B. Ganter, M. Liquiere, E. M. Nguifo, and G. Stumme, eds., July 23 2002.
- [8] H. BIAN AND R. BHATNAGAR, *A levelwise algorithm for interesting subspace clusters*, Proceedings of the 2005 IEEE International Conference on Data Mining, (2005).
- [9] E. CUTHILL AND J. MCKEE, *Reducing the bandwidth of sparse symmetric matrices*, in Proceedings of the 1969 24th national conference, New York, NY, USA, 1969, ACM, pp. 157–172.
- [10] K. S. G. LIU AND J. LI, *Efficient mining of large maximal bicliques*, Dawak, (2006), pp. 437–448.
- [11] B. GAMTER AND R. WILLE, *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag, Berlin, 1999.
- [12] G. C. GARRIGA, E. JUNTILA, AND H. MANNILA, *Banded structure in binary matrices*, in KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 2008, ACM, pp. 292–300.
- [13] R. B. H. BIAN, *An algorithm for lattice-structured subspace clustering*, Proceedings of the SIAM International Conference on Data Mining, (2005).
- [14] I. JEN LIN, M. K. SEN, AND D. B. WEST, *Classes of interval digraphs and 0,1-matrices*, in 28th S.E. Conf. Comb. Graph. Th. and Congr. Numer., 1997.

- [15] M. F. KAI PUOLAMAKI AND H. MANNILA, *Seriation in paleontological data using markov chain monte carlo methods*, PLoS Comput Biol., 2 (2006).
- [16] S. O. KUZNETSOV AND S. A. OBIEDKOV, *Algorithms for the construction of concept lattices and their diagram graphs*, in PKDD '01: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, London, UK, 2001, Springer-Verlag, pp. 289–300.
- [17] C. LINDIG, *Fast concept analysis*, 8th International Conference on Conceptual Structures, (2000).
- [18] O. A. MADEIRA S.C., *Biclustering algorithms for biological data analysis: A survey*, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 1 (1) (2004), pp. 24–45.
- [19] H. MANNILA AND E. TERZI, *Nestedness and segmented nestedness*, in KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 2007, ACM, pp. 480–489.
- [20] C.-J. H. MOHAMMED J. ZAKI, *Efficient algorithms for mining closed itemsets and their lattice structure*, IEEE Transactions on Knowledge and Data Engineering, 17 (4) (2005).
- [21] R. ROSEN, *Matrix bandwidth minimization*, in Proceedings of the 1968 23rd ACM national conference, New York, NY, USA, 1968, ACM, pp. 585–595.
- [22] M. SEN AND B. K. SANYAL, *Indifference digraphs: A generalization of indifference graphs and semiorders*, SIAM J. Discret. Math., 7(2) (1994), pp. 157–165.
- [23] I. SHMULEVICH AND W. ZHANG, *Binary analysis and optimization-based normalization of gene expression data*, Bioinformatics, 18 (2002), pp. 555–565.