



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Contributions to Robust Graph Clustering: Spectral Analysis and Algorithms

AM FACHBEREICH ELEKTROTECHNIK UND INFORMATIONSTECHNIK
DER TECHNISCHEN UNIVERSITÄT DARMSTADT
ZUR ERLANGUNG DES AKADEMISCHEN GRADES EINES
DOKTOR-INGENIEURS (DR.-ING.)
GENEHMIGTE DISSERTATION
VON

AYLİN TAŞTAN, M.Sc.

ERSTGUTACHTER: PROF. DR.-ING. ABDELHAK M. ZOUBIR
ZWEITGUTACHTER: PROF. DR.-ING. MICHAEL MUMA

DARMSTADT, 2023

Taştan, Aylin– Contributions to Robust Graph Clustering: Spectral Analysis and Algorithms
Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung der Dissertation auf TUpriints: 2023
URN: urn:nbn:de:tuda-tuprints-240689
Tag der mündlichen Prüfung: 26.05.2023

Veröffentlicht unter CC BY-NC-SA 4.0 International
<https://creativecommons.org/licenses/>

Beiträge zum Robusten Graphenclustering: Spektralanalyse und Algorithmen

KURZFASSUNG

In dieser Dissertation wird der Entwurf von schnellen und parameterfreien Graphen-Clustering-Methoden beschrieben, die robuste Clusterzuordnungen bestimmen können. Die Dissertation bietet eine Spektralanalyse sowie Algorithmen, die die erhaltenen theoretischen Ergebnisse an die Implementierung robuster Graphen-Clustering-Techniken anpassen.

Ein erster Beitrag dieser Arbeit ist die Definition eines spärlichen Graphenmodells, das mit den Zielen des Graphenclustering vereinbar ist. Dieses Modell basiert auf einer vorteilhaften Eigenschaft, die sich aus einer blockdiagonalen Darstellung einer Matrix ergibt, die die Dichte der Verbindungen innerhalb von Clustern und die Spärlichkeit der Verbindungen zwischen ihnen fördert. Es wird eine Spektralanalyse des spärlichen Graphenmodells einschließlich der Eigenwertzerlegung der Laplace-Matrix durchgeführt. Die Analyse der Laplace-Matrix wird durch die Definition eines Vektors vereinfacht, der alle relevanten Informationen enthält, die in der Laplace-Matrix enthalten sind. Die gewonnenen spektralen Eigenschaften spärlicher Graphen werden auf der Grundlage von zwei Methoden, die die Bestimmung des Spärlichkeitsniveaus als Annäherungen an die spektralen Eigenschaften der spärlichen Graphenmodelle formulieren, an das Clustering angepasst.

Ein zweiter Beitrag dieser Arbeit besteht darin, die Auswirkungen von Ausreißern auf das Graphenclustering zu analysieren und Algorithmen vorzuschlagen, die die Robustheit und den Grad der Spärlichkeit gemeinsam berücksichtigen. Die Grundlage für diesen Beitrag ist die Spezifizierung grundlegender Ausreißertypen, die in Fällen extremer Spärlichkeit auftreten, und die mathematische Analyse ihrer Auswirkungen auf dünn besetzte Graphen, um Algorithmen für das Graphenclustering zu entwickeln, die gegenüber den untersuchten Ausreißereffekten robust sind. Basierend auf den erhaltenen Ergebnissen werden zwei verschiedene robuste und spärlichkeitsbasierte Methoden zur Konstruktion von Affinitätsmatrizen vorgeschlagen. Motiviert durch die Auswirkungen von Ausreißern auf Eigenvektoren, werden eine robuste Fiedler-Vektor-Schätzung und eine robuste spektrale Clustermethode vorgeschlagen. Desweiteren wird ein Algorithmus zur Erkennung von Ausreißern, der auf dem Vertex-Grad aufbaut, vorgeschlagen und auf die Ganganalyse angewendet.

Die Ergebnisse dieser Arbeit zeigen, wie wichtig es ist, die Robustheit und den Grad der Spärlichkeit von Graphen-Clustering-Algorithmen gemeinsam zu berücksichtigen. Darüber hinaus liefert die vereinfachte Laplace-Matrix-Analyse vielversprechende Ergebnisse für die Entwicklung von Graphenkonstruktionsmethoden, die durch die Optimierung in einem Vektorraum, anstelle des normalerweise verwendeten Matrixraums, effizient berechnet werden können.

Contributions to Robust Graph Clustering: Spectral Analysis and Algorithms

ABSTRACT

This dissertation details the design of fast, and parameter free, graph clustering methods to robustly determine set cluster assignments. It provides spectral analysis as well as algorithms that adapt the obtained theoretical results to the implementation of robust graph clustering techniques.

Sparsity is of importance in graph clustering and a first contribution of the thesis is the definition of a sparse graph model consistent with the graph clustering objectives. This model is based on an advantageous property, arising from a block diagonal representation, of a matrix that promotes the density of connections within clusters and sparsity between them. Spectral analysis of the sparse graph model including the eigen-decomposition of the Laplacian matrix is conducted. The analysis of the Laplacian matrix is simplified by defining a vector that carries all the relevant information that is contained in the Laplacian matrix. The obtained spectral properties of sparse graphs are adapted to sparsity-aware clustering based on two methods that formulate the determination of the sparsity level as approximations to spectral properties of the sparse graph models.

A second contribution of this thesis is to analyze the effects of outliers on graph clustering and to propose algorithms that address robustness and the level of sparsity jointly. The basis for this contribution is to specify fundamental outlier types that occur in the cases of extreme sparsity and the mathematical analysis of their effects on sparse graphs to develop graph clustering algorithms that are robust against the investigated outlier effects. Based on the obtained results, two different robust and sparsity-aware affinity matrix construction methods are proposed. Motivated by the outliers' effects on eigenvectors, a robust Fiedler vector estimation and a robust spectral clustering methods are proposed. Finally, an outlier detection algorithm that is built upon the vertex degree is proposed and applied to gait analysis.

The results of this thesis demonstrate the importance of jointly addressing robustness and the level of sparsity for graph clustering algorithms. Additionally, simplified Laplacian matrix analysis provides promising results to design graph construction methods that may be computed efficiently through the optimization in a vector space instead of the usually used matrix space.

Publications

The following publications have been produced during the period of doctoral candidacy.

INTERNATIONALLY REFEREED JOURNAL ARTICLES

A. Taştan, M. Muma and A. M. Zoubir, “Fast and Robust Sparsity-Aware Block Diagonal Representation,” Under review in: *IEEE Trans. Signal Process.*, 2023.

A. Taştan, M. Muma and A. M. Zoubir, “Robust Regularized Locality Preserving Indexing for Fiedler Vector Estimation,” Under review in: *Signal Process.*, 2023.

A. Taştan, M. Muma and A. M. Zoubir, “Sparsity-aware Robust Community Detection,” *Signal Process.*, vol. 187, pp. 108147, 2021.

INTERNATIONALLY REFEREED CONFERENCE PAPERS

A. Taştan, M. Muma, E. Ollila and A. M. Zoubir, “Sparsity-Aware Block Diagonal Representation for Subspace Clustering,” in *Proc. 31th European Signal Process. Conf. (accepted)*, 2023.

A. Taştan, M. Muma and A. M. Zoubir, “Eigenvalue-Based Block Diagonal Representation and Application to p -Nearest Neighbor Graphs,” in *Proc. 30th European Signal Process. Conf.*, pp. 1761-1765, 2022.

A. Taştan, M. Muma and A. M. Zoubir, “Robust Spectral Clustering: A Locality Preserving Feature Mapping Based on M-estimation,” in *Proc. 29th European Signal Process. Conf.*, pp. 851-855, 2021.

A. Taştan, M. Muma and A. M. Zoubir, “An unsupervised approach for graph-based robust clustering of human gait signatures,” in *Proc. 2020 IEEE Radar Conf. (Best Student Paper Award, 1st Prize)*, pp. 1-6, 2020.

MATLAB CODES

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for FRS-BDR. 2023.
Online: <https://github.com/A-Tastan/FRS-BDR>

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for SABDR. 2023.
Online: <https://github.com/A-Tastan/SABDR>

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for RRLPI. 2023.
Online: <https://github.com/A-Tastan/RRLPI>

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for EBDR. 2022.
Online: <https://github.com/A-Tastan/EBDR>

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for RLPFM. 2021.
Online: <https://github.com/A-Tastan/RLPFM>

A. Taştan, M. Muma and A. M. Zoubir. MATLAB toolbox for SPARCODE. 2021.
Online: <https://github.com/A-Tastan/SPARCODE>

Acknowledgments

I would like to thank everyone who supported me during the time of my doctoral studies and contributed to the success of this work in various ways. First of all, I sincerely thank Prof. Dr.-Ing. Abdelhak M. Zoubir for giving me the opportunity to accomplish my Ph.D. at the Signal Processing Group. Thank you for your academic guidance and support and for giving me the time and freedom to pursue my research interests.

Further, my deepest gratitude to my co-supervisor Prof. Dr.-Ing. Michael Muma for his valuable feedback on my work and inspiring discussions. Thank you for guidance and continuous support, and for giving me the chance to work with so many outstanding people throughout my Ph.D. study.

I would like to thank Prof. Dr. D.Sc.(Tech) Esa Ollila for providing me collaboration chance and inviting me to Aalto University, Finland. I would like to thank for the enriching discussions and for your substantial comments on my work. It was an inspiration to visit and work with you. I would also like to thank everybody from Department of Signal Processing and Acoustics, Aalto University, Finland for warmly welcoming me and for inspiring me in various ways.

I also wish to thank Prof. Dr. rer. nat. Andy Schürr and Prof. Dr.-Ing. Li Zhang for being my examiners, and Prof. Dr.-Ing. Anja Klein for chairing my Ph.D. committee.

I wish to thank Republic of Turkey Ministry of National Education for financially supporting my Ph.D. study.

A huge thanks goes to all current and former members of the Signal Processing Group: Dr.-Ing. Christian Debes, Dr.-Ing. Michael Fauß, Dr. Roy Howard, Dr.-Ing. Di Jin, Dr.-Ing. Dominik Reinhard, Dr.-Ing. Ann-Kathrin Seifert. It has been a pleasure working with you all. I would like to thank Christian Eckrich and Pertami Kunz for the valuable but short time that we spent in our office. Thanks a lot to Mahmoud El-Hindi, Dr.-Ing. Huiping Huang and Dr.-Ing. Afief Dias Pambudi for the wonderful time that we had in our breaks. I would like to acknowledge, and thank, Martin Gölz and Christian Schroth for their great planning ability.

A special thanks goes to Taulant Koka and Jasin Machkour who are the members of Robust Data Science Group and made the last few years most enjoyable.

I cordially thank to Renate Koschella, Rojin Ulusoy and Hauke Fath for their help and support regarding administrative, technical and all other issues throughout these last four years.

Finally and most importantly, I wish to express my deepest gratitude to my parents Aynur and Fahri, as well as my sister Ayfer, for their unconditional love and support and encouragement to pursue my Ph.D studies. In addition, I would like to thank my friends, especially Mami, from

Turkey for their enthusiasm and encouragement about my future plans!

Darmstadt, May 2023

Contents

1	INTRODUCTION	1
1.1	Motivation	1
1.2	State-of-the-Art	4
1.3	Aims and Contributions	5
1.4	Dissertation Overview	6
2	FUNDAMENTALS OF GRAPH THEORY	7
2.1	Basic Definitions	7
2.2	Similarity Measures for Affinity Matrix Construction	8
2.3	Spectral Graph Theory	9
2.3.1	The Laplacian Matrix	9
2.3.2	Eigen-decomposition of the Laplacian Matrix	10
2.3.3	Fiedler Value and Algebraic Connectivity	10
2.4	Graph Partitioning and Clustering	11
2.4.1	The Graph Partitioning Problem	12
2.4.2	Clustering as Graph Partitioning	13
2.4.3	Modularity	13
2.4.4	Conductance	14
2.5	Spectral Methods for Graph Clustering	14
2.5.1	Unnormalized Spectral Clustering	15
2.6	Sparse Graphs	15
2.6.1	Sparse Affinity Matrix Construction Methods	16
2.6.2	Block Diagonal Representation (BDR)	17
3	SPARSE GRAPH MODELS FOR IDEAL PARTITIONING	18
3.1	Theoretical Design of the Sparse Graph Model	18
3.2	Spectral Analysis of the Sparse Graph Model	19
3.2.1	Laplacian Matrix of the Sparse Graph Model	20
3.2.2	Eigenvalues for the Sparse Graph Model	21
3.2.3	Eigenvectors for the Sparse Graph Model	22
3.2.4	Simplifying Laplacian Matrix Analysis of the Sparse Graph Model	23
3.3	Adapting Sparse Graphs to Sparsity-Aware Clustering	24

3.4	Sparsity-Aware Clustering based on Spectral Properties of the Sparse Graph Model	25
3.4.1	Eigenvalue-based Sparsity Level Control for Clustering	25
3.4.1.1	Introduction	25
3.4.1.2	Problem Formulation	26
3.4.1.3	Methodology	26
3.4.1.4	Experimental Results	28
3.4.1.5	Summary	31
3.4.2	Eigenvector-based Sparsity Level Control for Clustering	32
3.4.2.1	Introduction	32
3.4.2.2	Motivation : DBSCAN for Robust Spectral Analysis	33
3.4.2.3	Problem Formulation	34
3.4.2.4	Methodology	34
3.4.2.5	Experimental Results	37
3.4.2.6	Summary	40
4	OUTLIERS IN GRAPH CLUSTERING AND ROBUST SOLUTIONS	41
4.1	Determining Fundamental Outlier Types for Graph-based Clustering	41
4.1.1	Type I Outliers	42
4.1.2	Type II Outliers	43
4.1.3	Group Similarity	43
4.2	Outlier Effects on Sparse Graphs	43
4.2.1	Outlier Effects on Affinity Matrix	44
4.2.2	Outlier Effects on Overall Edge Weights	44
4.2.3	Outlier Effects on Eigenvalues	46
4.2.3.1	Type I Outliers' Effect on Eigenvalues	46
4.2.3.2	Type II Outliers' Effect on Eigenvalues	46
4.2.3.3	Group Similarity Effect on Eigenvalues	48
4.2.4	Outlier Effects on Eigenvectors	50
4.2.4.1	Type I Outliers' Effect on Eigenvectors	50
4.2.4.2	Type II Outliers' Effect on Eigenvectors	50
4.2.4.3	Group Similarity Effect on Eigenvectors	51
4.2.5	Outlier Effects on Simplified Laplacian Matrix Analysis	52
4.2.5.1	Type I Outliers' Effect on Simplified Laplacian Matrix Analysis	52
4.2.5.2	Type II Outliers' Effect on Simplified Laplacian Matrix Analysis	53
4.2.5.3	Group Similarity Effect on Simplified Laplacian Matrix Analysis	54
4.3	Sparsity and Outlier Occurrence	56
4.4	Robust Graph-based Clustering Methods	56
4.4.1	Robust and Sparsity-Aware Affinity Matrix Construction Methods	58
4.4.1.1	Sparsity-Aware Robust Community Detection	58
4.4.1.2	Fast and Robust Sparsity-Aware Block Diagonal Representation	85
4.4.2	Robust Eigenvector Estimation Methods	101

4.4.2.1	Robust Regularized Locality Preserving Indexing for Fiedler Vector Estimation	101
4.4.2.2	Robust Spectral Clustering: A Locality Preserving Feature Mapping Based on M-estimation	127
4.4.3	Outlier Detection based on Vertex Degree and Application to Gait Analysis	135
4.4.3.1	Introduction	135
4.4.3.2	Problem Formulation	136
4.4.3.3	Proposed Algorithm	137
4.4.3.4	Experimental Results	140
4.4.3.5	Conclusions	146
5	CONCLUSION AND OUTLOOK	147
5.1	Summary and Conclusion	147
5.2	Future Research Directions	148
5.2.1	Assumptions on Sparse Graph Model	148
5.2.2	Fundamental Outlier Types in Random Graphs	149
5.2.3	Robust Graph-based Clustering for Large Graphs	149
5.2.4	Time-Series Analysis Applications based on Visibility Graphs	150
	APPENDIX A PROOFS AND ADDITIONAL THEORETICAL INFORMATION	151
A.1	Spectral Analysis of the Sparse Graph Model	152
A.1.1	Generalized Eigen-decomposition based Analysis	152
A.1.1.1	Proof of Theorem 1	152
A.1.1.2	Proof of Theorem 2	153
A.1.1.3	Proof of Theorem 3	154
A.1.2	Standard Eigen-decomposition based Analysis	154
A.2	Outlier Effects on Sparse Graph Model	156
A.2.1	Generalized Eigen-decomposition based Analysis	156
A.2.1.1	Proof of Theorem 4	156
A.2.1.2	Proof of Theorem 5	160
A.2.1.3	Proof of Preposition 4.2.1	164
A.2.1.4	Proof of Preposition 4.2.2	165
A.2.1.5	Proof of Preposition 4.2.3	166
A.2.1.6	Proof of Theorem 6	167
A.2.1.7	Proof of Theorem 7	169
A.2.1.8	Proof of Corollary 7.1	170
A.2.2	Standard Eigen-decomposition based Analysis	171
A.2.2.1	Type II Outliers' Effect on Eigenvalues	171
A.2.2.2	Group Similarity Effect on Eigenvalues	176
A.3	Outlier Effects on the Fiedler Vector	181
A.3.0.1	Proof of Preposition 4.4.1	181
A.3.0.2	Proof of Preposition 4.4.2	182

A.4	Theoretical Analysis of RRLPI	183
	A.4.0.1 Proof of Theorem 9	183
	A.4.0.2 Proof of Theorem 10	184
	A.4.0.3 Proof of Corollary 10.1	185
A.5	Auxiliary Information	186
	A.5.1 The Generalized Matrix Determinant Lemma	186
	A.5.2 Moore-Penrose Inverse of Weighted Data Matrix	187
APPENDIX B ADDITIONAL INFORMATION FOR ROBUST GRAPH CLUSTERING		
METHODS		188
B.1	Additional Information for SPARCODE	188
	B.1.1 Scenario 1	188
	B.1.2 Scenario 2	190
B.2	Additional Information for FRS-BDR	191
	B.2.1 Visual Summary of FRS-BDR	191
	B.2.2 Sparse Laplacian Matrix Analysis	199
	B.2.3 Computational Complexity Analysis	201
	B.2.4 Additional Algorithms	206
	B.2.5 Experimental Setting and Additional Experimental Results	208
	B.2.5.1 Experimental Setting	208
	B.2.5.2 Additional Experimental Results	209
B.3	Additional Information for RRLPI	219
	B.3.1 Experimental Setting	219
	B.3.2 Additional Results for Synthetic Data Sets	220
	B.3.3 Additional Results for Cluster Enumeration	222
	B.3.4 Additional Results for Image Segmentation	226
LIST OF ACRONYMS		231
LIST OF NOTATION & SYMBOLS		233
LIST OF FIGURES		237
LIST OF TABLES		243
REFERENCES		248

If I had an hour to solve a problem I'd spend 55 minutes thinking about the problem and five minutes thinking about solutions.

—Albert Einstein

1

Introduction

1.1 MOTIVATION

Machine learning is of paramount importance in modern complex intelligent systems. A fundamental machine learning task is to discriminate between similar and dissimilar data points, based on a similarity criterion, to create clusters of points. Clustering arises in diverse areas, e.g. image analysis [LZX20], medical diagnostics [NRG19], bioinformatics [PEC19], information retrieval [JV19] and data mining [ASI20], and is an active area of fundamental research. Despite its application diversity, the notion of a *cluster* does not have a precise definition and the clustering problem has been studied from different perspectives. Graph clustering is one of the most popular techniques for establishing clusters as it is efficient in learning the hidden relationships in a data set.

In a graph, the vertices of the graph represent the data points while the edges measure association relationships between them based on non-zero components of an affinity matrix [XT15]. Similar to the general idea of clustering, the goal of graph clustering is to obtain clusters that are internally dense while being sparsely connected, or ideally unconnected, to the remainder of the graph. Problems potentially arise with graph clustering when, for example, clusters are obscured by

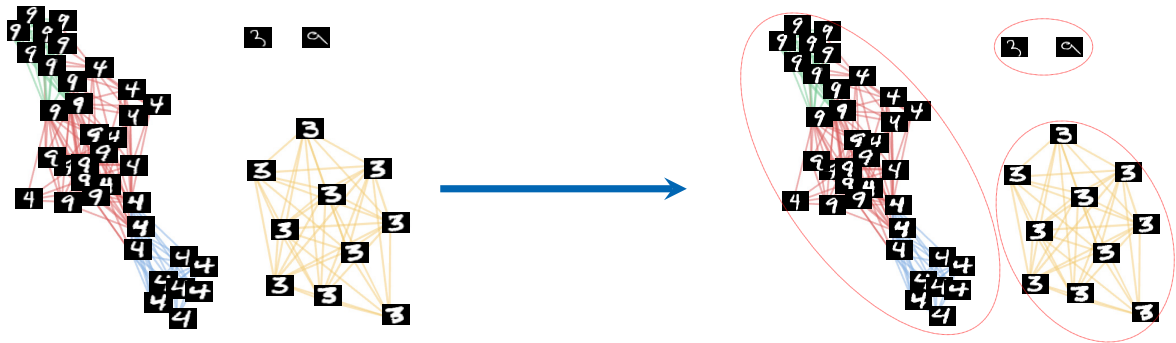


Figure 1.1: Exemplary graph clustering. Left: graph representation of digit samples from the MNIST data base [HS98]. The red edges represent connections to outliers which are ones that have connections to more than one group of digit samples. The green, blue and yellow lines represent the within-cluster edges of digits 9, 4 and 3, respectively. Right: cluster assignment based on the general graph clustering idea that maximizes the number of intra-cluster edges while minimizing the number of inter-cluster edges.

undesired edges between them, and/or when outliers and noisy data points result in performance degradation in graph clustering algorithms.

To provide a visual understanding, an exemplary corrupted graph model and associated cluster assignment are shown in Fig. 1.1 for a defined level of sparsity using the well-known handwritten digit samples from the MNIST data base [HS98]. In the exemplary graph model, the red edges represent connections to outliers while the remaining edges are the informative edges that connect true samples. The green, blue and yellow lines represent the within-cluster edges of digits 9, 4 and 3, respectively. The red ellipses indicate cluster assignments that are computed based on a general graph clustering idea, which maximizes the number of intra-cluster edges while minimizing the number of inter-cluster edges to obtain disjoint clusters. As can be seen, the undesired edges between characters four and nine obscure the clusters and result in assigning unconnected digit samples into a small cluster. An important property of these over-connected or under-connected components is that their occurrence is directly related with the graph construction.

The importance of graph construction is illustrated in Fig. 1.2 where graphs are shown for handwritten digit samples from the MNIST data base for varying sparsity levels. For example, the dense graph in Fig. 1.2a contains many undesired edges between different characters which especially makes the task of separating characters four from nine challenging. Additionally, there are outliers of character three and nine that are not similar to the majority of data. The increased sparsity in Fig. 1.2b reduces the number of undesired edges between different characters and provides a better structure for clustering. However, as is evident in Fig. 1.2c, further increasing the sparsity generates many disconnected components and the underlying structure of the true

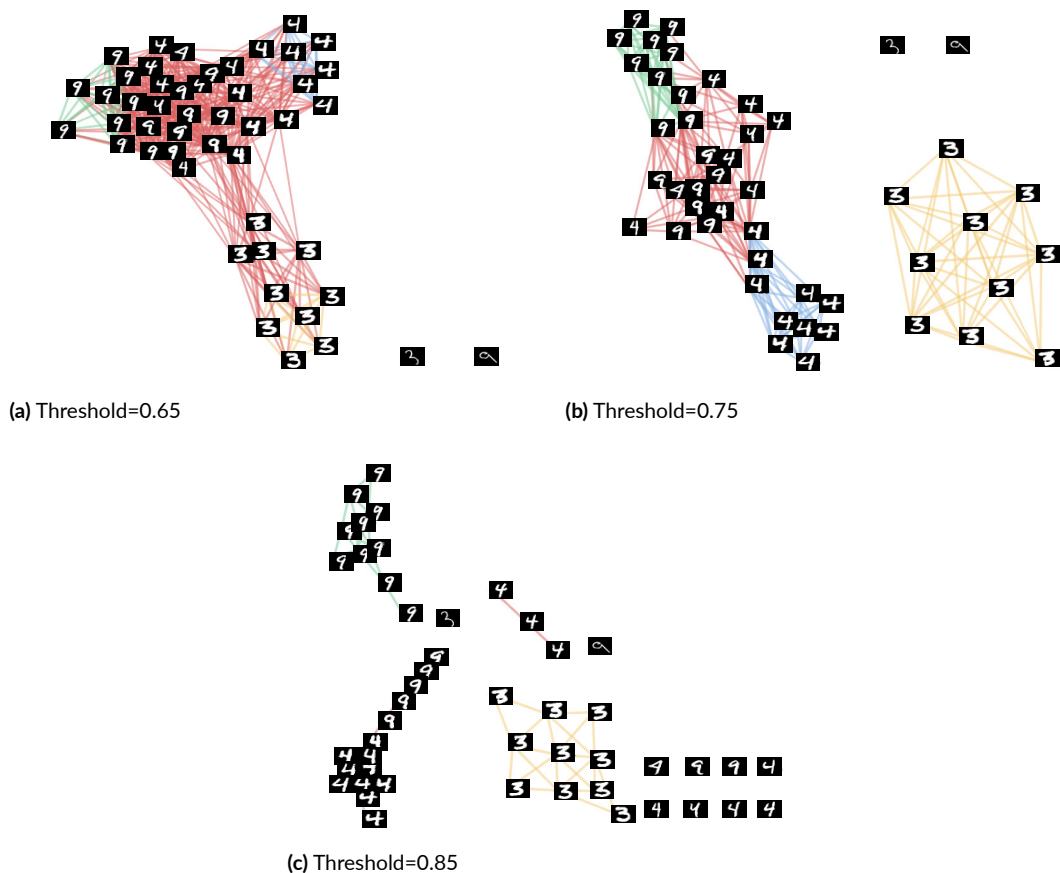


Figure 1.2: Example graph constructions for handwritten digit samples from MNIST data base [HS98].

clusters is completely lost. Summarizing, this means that an inefficient graph construction leads to inaccurate clustering results due to the obscured true clusters or lost informative edges.

In addition to the above challenges, the number of clusters is, in general, unknown in real-world graph clustering applications. According to this, for a densely connected graph structure, the undesired edges between different clusters may result in merging connected clusters into a big cluster and under-estimating the true number of clusters. On the other hand, increasing sparsity may lead to assigning unconnected components into different clusters and, consequently, to over-estimation of the cluster number.

Graph clustering algorithms are often computationally intensive and, especially when analyzing larger data sets, computation efficiency of such algorithms is of high importance. Typically, for graphs that are comprised of N vertices, graph clustering algorithms generally operate on $N \times N$ matrices to construct graph models or to partition them. When the number of vertices in a graph

is increased with an increase in the number of data samples being analyzed, the computational cost associated with graph clustering algorithms can be significant and a limiting factor in data analysis.

The main problems associated with graph clustering are that the number of clusters is generally unknown, the level of sparsity to use is not precisely defined, a large number of vertices leads to high computational complexity, and outliers in the data underpinning the graph are likely to negatively impact the establishment of the true cluster structure. There is clear research interest in addressing these issues and *fast and parameter-free graph clustering methods that jointly address robustness and graph construction* are detailed in this thesis.

1.2 STATE-OF-THE-ART

Graph clustering has been extensively researched during the last decades, e.g. [KWC19, LH18, Sch07]. In particular, robust graph clustering has received significant research attention, see, for example, [YCL20, AGR19, ZCS19, LNC18].

Since graph construction plays a crucial role in obtaining accurate clustering results, one popular approach for integrating robustness into graph construction is to utilize sparse affinity matrices [AGR19, LNC18, ZZL18] whose non-zero components represent the edge weights of the graph. These methods restructure the affinity matrix based on prior information, e.g., the number of clusters [LNC18] and the level of sparsity [AGR19, ZCS19, LNC18, ZZL18] that is significant for the graph structure.

Block diagonal (BD) structure of the affinity matrix is a commonly desired property in graph clustering because it represents clusters of feature vectors by non-zero coefficients that are concentrated in blocks. Consistent with the general idea of graph clustering (for details, see Section 1.1 or Section 2.4), this means that a block diagonally structured affinity matrix is a helpful tool that provides internally dense clusters that are sparsely connected, or ideally unconnected, to the other clusters. To impose this advantageous structure on the affinity matrix, commonly used existing block diagonal representation (BDR) methods use regularization with BD priors, e.g. based on a low-rank property [XTX15, LLY12, LY11], sparsity [FLW21, WZW17], or a known number of blocks K [LFL18, XGL17].

Spectral clustering (SC) algorithms that embed the vertices of a graph into a low dimensional space based on the eigen-decomposition of the Laplacian matrix, are popular alternatives to address the graph clustering problem. Consequently, suppressing outliers' effects in the embedding space, such as in [YCL20, ZCS19, CNW15, PYT15] has become a further popular research interest in the literature. However, most of these approaches require prior knowledge, e.g., the label information

of a data set [YCL20, CNW15] or data dependent parameter tuning to determine the descriptive features [ZCS19].

While all of the above works focus on robustness, none of them consider the relationship between the graph construction and the outliers' occurrence. Additionally, most of the robust graph clustering methods require prior information whose determination is challenging, especially in the presence of outliers and heavy-tailed noise that may obscure the underlying structure. Therefore, the following section states the aim of this work and briefly summarizes our contributions to robust graph clustering that address the above issues.

1.3 AIMS AND CONTRIBUTIONS

The aim of this doctoral project is to develop fast and parameter-free robust graph clustering methods. The original contributions of the thesis are with respect to ideal partitioning of sparse graphs and robustness in graph clustering in the presence of outliers, that are detailed below:

Sparse Graphs for Ideal Partitioning (Chapter 3):

- Motivated by the conformity of BDR to graph clustering objectives, a sparse graph model providing internally dense and externally unconnected clusters is defined.
- Spectral analysis of the sparse graph model is conducted and a vector representing the blocks as a piece-wise linear function is defined. In this way, the relevant information that is contained in the Laplacian matrix, i.e., the similarity coefficients and the block sizes, is transformed into a vector space analysis and spectral analysis of the Laplacian matrix is simplified.
- The defined sparse graph is adapted to sparsity-aware clustering and graph clustering methods that naturally determine the appropriate level of sparsity based on the spectral properties of sparse graph model.

Outliers in Graph Clustering and Robust Solutions (Chapter 4):

- Fundamental outlier types whose occurrence depends on the graph construction are defined for graph clustering. More precisely, unconnected vertices (Type I outliers), vertices that generate false positive connections between different clusters (Type II outliers) and the extreme case that groups of vertices are connected to another one (Group similarity), are considered.

- To jointly address robustness and sparsity, outliers' effects on sparse graphs is theoretically analyzed. In particular, the eigenvalues and the eigenvectors are computed for three fundamental cases. The developed analysis underpins, via the use of a simplified Laplacian matrix, the development of fast, robust and sparsity-aware graph clustering algorithms.
- Robust graph clustering methods that mitigate the outliers' negative impact on the affinity matrix, the simplified Laplacian matrix analysis, the Fiedler vector and SC, are proposed. All proposed methods are robust and parameter-free alternatives to solve the graph clustering problem except for the robust SC method, which requires knowledge of the number of clusters. In addition to these, an outlier detection algorithm built upon the vertex degree is proposed and applied to gait analysis.

1.4 DISSERTATION OVERVIEW

The remaining part of the dissertation is organized in four chapters where the main contributions are detailed in Chapters 3 and 4.

Chapter 2 details the fundamental concepts of graph theory that are relevant for the thesis, and provides an overview of why sparsity in graph clustering is of importance.

Chapter 3 starts with the theoretical design of a sparse graph model which can provide ideally separated clusters through the use of BDR. Next, eigenvalues and the eigenvectors of the sparse graph model are computed and a vector that simplifies the Laplacian matrix analysis by representing blocks as piece-wise linear function is defined. Spectral analysis of the sparse graph model is adapted to sparsity-aware clustering by formulating determination of the sparsity level as approximations to spectral properties of the sparse graph modes.

In Chapter 4, the existence of outliers and robust graph clustering solutions are considered. To understand how to best integrate robustness, first, fundamental outlier types are defined for graph clustering. Then, their effects on sparse graphs are mathematically analyzed in terms of the affinity matrix, overall edge weights, eigen-decomposition and the simplified Laplacian matrix analysis. Based on the obtained results, robust graph clustering methods providing robustness to outliers' effects on the matrix and embedding spaces are proposed. The vertex degree, which is a significant property to determine outliers in sparse graphs, is used for outlier detection and the proposed method is applied to gait analysis.

Finally, a summary of the thesis, conclusions and an outlook representing some open problems and future research directions are detailed in Chapter 5.

If I were again beginning my studies, I would follow the advice of Plato and start with mathematics.

- Galileo Galilei

2

Fundamentals of Graph Theory

This chapter introduces fundamental concepts of graph-based cluster analysis. First, basic definitions and similarity measures for graph construction are presented in [Section 2.1](#) and [Section 2.2](#), respectively. Then, a general introduction to spectral graph theory is provided in [Section 2.3](#) and the main ideas of graph partitioning and clustering are detailed in [Section 2.4](#). Building upon these sections, state-of-the-art spectral methods for graph partitioning are revisited in [Section 2.5](#). Finally, a discussion of sparse graphs is given in [Section 2.6](#).

2.1 BASIC DEFINITIONS

Graph theory is an area of extensive research that comprises numerous concepts and various graph representations [[New18](#), [Wes01](#)]. This section presents the relevant theory for weighted undirected graphs that are used throughout this thesis.

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{M \times N}$ be a data matrix with M denoting the dimension of the feature vectors and N being the number of feature vectors. We assume that the data matrix \mathbf{X} can be represented as an undirected weighted graph $G = \{V, E, \mathbf{W}\}$. Here, V denotes the set of vertices corresponding to the feature vectors, E is the set of edges representing the relationships between

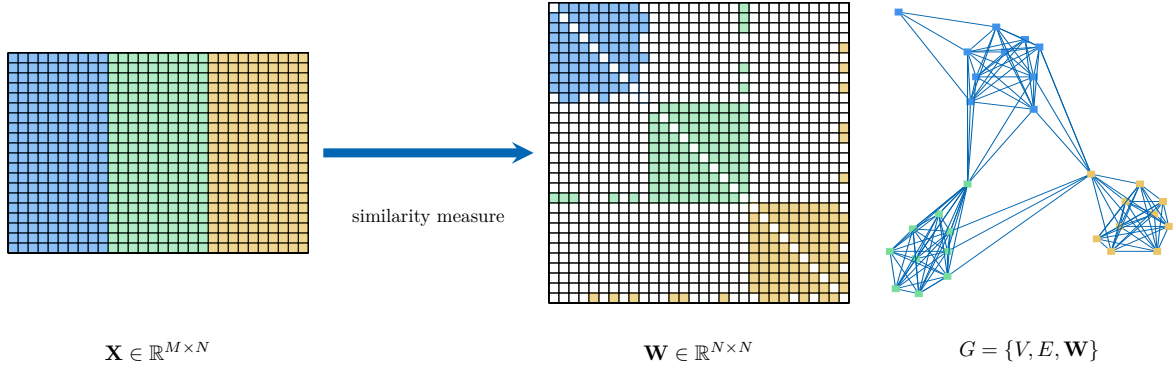


Figure 2.1: Exemplary graph construction process.

these vertices, and $\mathbf{W} \in \mathbb{R}^{N \times N}$ is called the affinity matrix whose m, n th component consists of the edge weight between the m th and the n th vertices that are, respectively, associated with the m th and n th column vectors $\mathbf{x}_m \in \mathbb{R}^M$ and $\mathbf{x}_n \in \mathbb{R}^M$ of \mathbf{X} . $\mathbf{W} \in \mathbb{R}^{N \times N}$ is referred to as the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ when all of its entries take on the value one or zero (for a detailed discussion, see, e.g., Section 6.2 in [New18]).

To provide a visual understanding, the process of transforming a data matrix \mathbf{X} to a graph G is illustrated in Figure 2.1 for $K = 3$ clusters. The column vectors of \mathbf{X} , the similarity coefficients of \mathbf{W} and the vertices of G are highlighted in blue, green and orange, respectively, according to their cluster membership. Additionally, zero-valued similarity coefficients of \mathbf{W} are highlighted in white. Each non-zero entry of \mathbf{W} is represented by an edge between vertices in the resulting graph. Hence, choosing an appropriate similarity measure, which yields the affinity matrix given the data, plays a crucial role in obtaining an informative graph.

2.2 SIMILARITY MEASURES FOR AFFINITY MATRIX CONSTRUCTION

Due to its central role in defining the graph structure, the measure of similarity between vertices is a central area of research in graph theory [New18]. One of the most commonly used similarity measures is known as *cosine similarity*, i.e.,

$$w_{m,n} = \frac{\mathbf{x}_m^\top \mathbf{x}_n}{\|\mathbf{x}_m\|_2 \|\mathbf{x}_n\|_2}, \quad m = 1, \dots, N \text{ and } n = 1, \dots, N, \quad (2.1)$$

where $w_{m,n}$ is the similarity coefficient of the affinity matrix \mathbf{W} for the m th and n th data vectors \mathbf{x}_m and \mathbf{x}_n , respectively. Alternatively, an affinity matrix can be formed, for example, by using *Pearson's*

linear correlation coefficient, as

$$w_{m,n} = \frac{(\mathbf{x}_m - \hat{\mu}_m)^\top (\mathbf{x}_n - \hat{\mu}_n)}{\hat{\sigma}_m \hat{\sigma}_n} \quad (2.2)$$

with associated sample means $\hat{\mu}_m, \hat{\mu}_n$, and sample standard deviations $\hat{\sigma}_m, \hat{\sigma}_n$, respectively, for $m, n = 1, \dots, N$. A more detailed discussion and additional examples of similarity measures can be found in [LHN06].

2.3 SPECTRAL GRAPH THEORY

Spectral graph theory is an extensive research area that studies eigenvalues and eigenvectors of matrices associated with graphs to understand hidden relationships in those graphs [Spi12, Chu97]. As an introduction to spectral graph theory, this section presents fundamental concepts that are briefly explained in the sequel.

2.3.1 THE LAPLACIAN MATRIX

To define the Laplacian matrix, the first step is to compute the matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ which is a diagonal weight matrix with weights given by

$$d_{m,m} = \sum_{n=1}^N w_{m,n} \quad (2.3)$$

on the diagonal and it is equivalent to the diagonal matrix of degrees for an adjacency matrix. Throughout this thesis, $\mathbf{D} \in \mathbb{R}^{N \times N}$ and $d_{m,m}$, respectively, denote the overall edge weight matrix and the overall edge weight that is attached to the m th vertex.

After computing the overall edge weight matrix \mathbf{D} , the unnormalized Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ can be defined as follows:

$$l_{m,n} = \begin{cases} d_{m,m} & \text{if } m = n \\ -w_{m,n} & \text{otherwise} \end{cases}, \quad (2.4)$$

where $l_{m,n}$ denotes the m, n th component of the Laplacian matrix for $\mathbf{L} = \mathbf{D} - \mathbf{W}$. A visual summary of the matrices, that have been defined up to now, is provided in Figure 2.2.

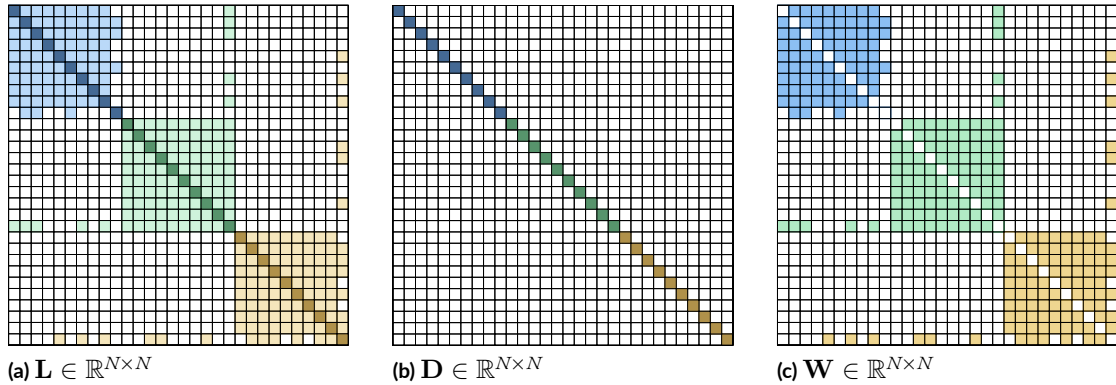


Figure 2.2: Exemplary Laplacian, overall edge weight and affinity matrices.

2.3.2 EIGEN-DECOMPOSITION OF THE LAPLACIAN MATRIX

Let the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ be nonnegative definite with the eigenvalues of the standard eigen-problem

$$\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m, \quad (2.5)$$

or in a generalized eigenvalue problem form

$$\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m, \quad (2.6)$$

with associated eigenvalues $0 \leq \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}$ sorted in ascending order. Here, λ_m denotes the m th eigenvalue and $\mathbf{y}_m \in \mathbb{R}^N$ is the eigenvector associated with λ_m .

2.3.3 FIEDLER VALUE AND ALGEBRAIC CONNECTIVITY

Let $0 \leq \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}$ be the sorted eigenvalues of the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ for $N \geq 2$. Then, the second smallest eigenvalue of the Laplacian matrix λ_1 , that is known as Fiedler value, is the algebraic connectivity $a(G)$ of graph G , and it is zero-valued if and only if G is unconnected [Fie89, Fie73].

Since the Fiedler value provides information about the connectedness of graph, it is directly linked to the graph cut problem which is the separation of graph vertices into disjoint subsets [DHZ01]. For example, if the Fiedler value is small, a graph partitioning that is performed using the associated eigenvector (Fiedler vector), results in a cut in which the ratio of edges crossing the cut to the number of vertices in the cut is similarly small. [KLJ09]. As a natural result of its applicability to the graph cut problem, the Fiedler vector has been the subject of many graph

partitioning approaches, e.g. [New13, OS05, DHZ01], whose main principles will be detailed in the following sections.

2.4 GRAPH PARTITIONING AND CLUSTERING

Clustering is a fundamental research area that has various applications, such as, image analysis [LZX20, SY20], medical diagnostics [NRG19], bioinformatics [PEC19], information retrieval [JV19] and data mining [ASI20]. Despite its application diversity, the precise definition of *cluster* does not exist [SPG17]. As a consequence, the clustering problem has been studied from different perspectives and this section introduces a survey of some important clustering techniques with a particular focus on graph partitioning algorithms that can be applied to the clustering problem.

Clustering approaches have been categorised in different ways [HXZ20, HK17, SPG17, XT15]. For example, in [HXZ20], clustering approaches are grouped in four categories: partitional, hierarchical, density-based and model-based. The hierarchical clustering methods form clusters based on a hierarchy that is built by iteratively dividing the patterns using the top-down or bottom-up approach [SPG17]. Agglomerative clustering methods are the bottom-up approaches, e.g. [MDG21, BYL15], that aggregate the individual points into larger clusters based on the determined termination conditions. By contrast, the top-down approaches, that are also known as divisive hierarchical clustering methods, split the data points into smaller clusters based on certain termination conditions [JZH22]. Unlike hierarchical clustering approaches, partitional clustering assigns data points to clusters by optimizing some criterion functions where Euclidean distance is a commonly used criterion [SPG17]. The most commonly used partitional clustering examples are K -means type methods (e.g., K -means, K -medoids and fuzzy C -means methods), which have a broad range of extensions, e.g. [BPB21, SY20, SR19]. The key idea of K -means clustering is dividing the given data set into K clusters such that the average squared Euclidean distance from the data points to the sample mean of each cluster is minimized. Different from K -means type strategies, density-based approaches attempt to reveal clusters according to the density of regions in the data [HXZ20]. For instance DBSCAN [EKS96], which is a well-known density-based clustering method, assigns data points into clusters for which every cluster member has to contain at least minimum number of neighbors in the given neighborhood radius. OPTICS [ABK99] and DENCLUE [HG07] are the other popular density-based clustering approaches and there are many state-of-the-art approaches that follow this paradigm, see e.g. [BM21, JJ19]. Lastly, model-based clustering approaches use a probability that is derived from the assumed statistical model as the clustering criterion [XT15]. These approaches are mainly based on statistical learning [Ras99] or

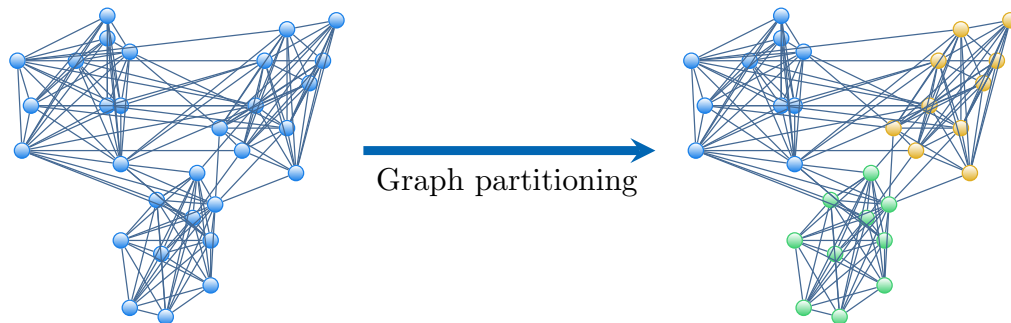


Figure 2.3: Exemplary balanced graph partitioning.

neural network learning [Koh90] and there are many recent works about model-based clustering, e.g. [SM21, TMZ21, TWS19, TMZ18].

Analogous to performing clustering directly on the data points, clustering can be performed on the corresponding graph where the vertices of the graph represent the data points while edges measure association relationships between them [XT15]. One of the most popular graph clustering algorithms is known as minimal spanning tree (MST) [Zha71] and there are numerous advanced graph clustering techniques, e.g. [KWC19, SZL19, WLW18]. A common goal for these approaches is to obtain disjoint clusters where the number of edges within the clusters is maximized in contrast to the number of edges in between the clusters. This problem is directly linked to graph partitioning, which can be considered as graph clustering under certain conditions.

2.4.1 THE GRAPH PARTITIONING PROBLEM

Graph partitioning has been extensively researched in the literature and it has different applications, such as, social networks, road networks and image processing [BMS16]. Beyond its importance in graph theory, graph partitioning is directly related to the clustering problem. To understand the relationship between these concepts, this section explains the main ideas behind graph partitioning with a brief summary of the state-of-the-art approaches.

The graph partitioning problem is the separation of graph vertices into balanced partitions while minimizing the number of edges that cross the cut [NHG19, ARV08]. This problem is NP-hard [ARV09] and thus, most of the graph partitioning approaches are heuristic, e.g., [FJL20, NHG19]. In addition to existing heuristic graph partitioning approaches, there are theoretical approximation algorithms for the sparsest cut, edge expansion, balanced separator and graph conductance problems, such as, [LM14, ARV09, ARV08, LR99].

To provide a better understanding, an exemplary graph partitioning is presented in Figure 2.3 for

a balanced partition in which the number of vertices on the two sides is within a constant factor of each other. As can be seen, the balanced graph partition produces equally sized clusters that share the minimum number of edges.

2.4.2 CLUSTERING AS GRAPH PARTITIONING

Since the goal of clustering is to assign similar data points into the same cluster while assigning dissimilar ones into different clusters based on a given similarity criterion [XT15], for a graph whose edge weights measure similarity of data points, this objective can be transferred to maximizing the number of edges for intra-clusters while minimizing it for inter-clusters. This means that the graph partitioning problem can be interpreted as a clustering problem.

Even though graph clustering and partitioning solve a very similar problem, clustering approaches, generally speaking, do not control the size of clusters [SBH16], while most of the graph partitioning approaches attempt to obtain balanced partitions, e.g. [ARV09, ARV08, LR99]. Additionally, the number of clusters is generally assumed to be unknown in clustering approaches and thus, the number of clusters has to be estimated as well. Herein, a commonly used approach to estimate the number of clusters (or so called communities [Sch07]) is to maximize the quality of the partition, where modularity and conductance are well-known quality measures that have been commonly used in graph clustering, e.g. [YHJ20, GZZ19]. In the sequel, these metrics are briefly revisited.

2.4.3 MODULARITY

Modularity (mod) [New06, GN02], is a metric that evaluates the quality of a partition with respect to the similarity of feature vectors in an affinity matrix. Analogous to the objectives of graph partitioning and clustering, a cluster assignment yields a high modularity score if the vertices have more edges within the assigned cluster while they have fewer edges connected to the other clusters. The modularity score is calculated by

$$\text{mod} = \frac{1}{2g} \sum_{m,n}^N \left[w_{m,n} - \frac{d_m d_n}{2g} \right] 1_{\{c_m=c_n\}} \quad (2.7)$$

where $d_m = \sum_{n=1}^N w_{m,n}$ denotes the sum of the weights of edges attached to vertex m , $g = \frac{1}{2} \sum_{m,n} w_{m,n}$ and the indicator function $1_{\{c_m=c_n\}}$ is equal to one if $c_m = c_n$ and is zero otherwise.

2.4.4 CONDUCTANCE

Conductance (cond) evaluates the quality of a partition in terms of the fraction of edges that point outside the community. Therefore, a small-valued conductance score means good partitioning performance. It is calculated as [YL15]

$$\text{cond} = \frac{\sum_{m,n}^N w_{m,n} \mathbf{1}_{\{c_m \neq c_n\}}}{\sum_{m,n}^N w_{m,n}},$$

where $w_{m,n}$ denotes the weight of the edge between the m th and the n th vertices, and the indicator function $\mathbf{1}_{\{c_m \neq c_n\}}$ equals one if $c_m \neq c_n$ and is zero otherwise.

2.5 SPECTRAL METHODS FOR GRAPH CLUSTERING

An alternative to address the graph clustering problem are spectral clustering algorithms which are mainly built upon eigen-decomposition of Laplacian matrices associated with an affinity matrix. Due to their efficiency and solid theoretical foundation [NC11], they have been the subject of intense scientific research for decades, e.g., [JDX14, Lux07, SM00, DH73] with a wide range of applications, such as in computer vision [LNC18], bioinformatics [PZ18], and medical diagnosis [XGZ71]. From a mathematical point of view, after affinity matrix construction, spectral clustering only requires solving a linear problem [SPG17], and this makes it advantageous in comparison to graph cut algorithms [JDX14]. Beyond its mathematical advantage, it does not make any assumptions on the cluster shapes which makes it applicable to more complex scenarios, such as, intertwined spirals or other arbitrary nonlinear shapes [LH18]. Additionally, spectral clustering is applicable to large data sets as long as the graph is sparse (for details about sparse graphs, see Section 2.6).

Spectral clustering has been studied from different aspects [JDX14], for example, constructing the affinity matrix, forming the Laplacian matrix, computing and selecting the eigenvectors and the number of clusters. Among these different spectral clustering approaches, a well-known spectral clustering algorithm, which is commonly used throughout this thesis, is introduced in the following.

Algorithm 1: Unnormalized Spectral Clustering

Input: The data matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$ and the number of clusters K

Step 1: Affinity Matrix Construction

Construct the affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ (For details, see Section 2.2)

Step 2: Eigen-decomposition of the Laplacian Matrix

Compute the unnormalized Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ for $\mathbf{L} = \mathbf{D} - \mathbf{W}$

Compute $\mathbf{Y} = [\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{K-1}] \in \mathbb{R}^{N \times K}$ whose column vectors are the eigenvectors that are associated to the K smallest eigenvalues of \mathbf{L}

Step 3: K -means Clustering

Treat each row of \mathbf{Y} as a K -dimensional feature vector and perform K -means clustering

Output: A vector $\hat{\mathbf{c}}$ containing the estimated cluster labels

2.5.1 UNNORMALIZED SPECTRAL CLUSTERING

For a given graph G , the main step for spectral clustering is to compute the Laplacian matrix which has not been uniquely determined in the literature [LH18]. As in Section 2.3.1, an unnormalized Laplacian matrix, also called unnormalized graph Laplacian, associated with an undirected graph G is considered. The Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ is symmetric and nonnegative definite with eigenvalues $0 \geq \lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_{N-1}$ that are sorted in ascending order. In the following, the unnormalized spectral clustering algorithm built upon the unnormalized graph Laplacian \mathbf{L} is introduced.

2.6 SPARSE GRAPHS

In numerical analysis, a matrix is called sparse if it has relatively large number of zero-valued elements and it is called dense if most of its elements are non-zero. The ration of zero-valued elements divided by the total number of elements is commonly referred to as the sparsity of a matrix. From a graph clustering point of view, this means that the sparsity of a graph is directly linked to the number of edges and it has many advantages in graph clustering approaches [Sch07]. For example, many graph clustering performance measures aim to find clusters that are internally dense while being sparsely connected, or ideally unconnected, to the rest of the graph. Another advantage of these graphs is the computational efficiency in which having fewer number of edges reduces the number of operations in clustering approaches [SPG17]. Motivated

by its importance in graph clustering, this section presents state-of-the-art sparse affinity matrix construction methods and the main ideas of a commonly desired sparse matrix structure.

2.6.1 SPARSE AFFINITY MATRIX CONSTRUCTION METHODS

By virtue of its crucial role in graph clustering, the construction of sparse affinity matrices is a substantial research area and there are many different modalities, e.g. [LKJ20, LSW16, YLR16, EV13, CYY09].

One of the most traditional ways of constructing a sparse affinity matrix is known as nearest neighbor graphs. For a given global parameter, which is denoted by p throughout this thesis, the idea is to obtain a graph in which every vertex has only edges to its p -nearest neighbors based on the determined metric space, e.g., the Euclidean distance or cosine distance [EPY97]. Similarly, ε -neighborhood graph construction is built upon the idea that every vertex is connected to other vertices whose distances are less than the predefined global parameter ε based on the determined metric space [UST09]. For both methods, a smaller valued global parameter results in greater sparsity. However, using Euclidean or cosine metric spaces makes these approaches sensitive to outliers and/or data noise. Additionally, when the data is distributed in different ways, determining the sparsity based on a single parameter may result in performance degradation for these algorithms [CYY09]. For instance, p -nearest neighbor graph construction may involve inhomogeneous data points that are far away from the true clusters when the data is distributed in different ways.

A popular robust alternative to sparse affinity matrix construction is known as ℓ_1 -graph construction in which every data point is constructed by the linear combination of the remaining data points and a noise term by minimizing the ℓ_1 norm of both reconstruction coefficients and data noise [WMM10, CYY09]. In contrast to the above traditional affinity matrix construction methods, the ℓ_1 -graph construction method is robust against outliers and the number of neighbor selection is adaptive for each data point. As a result, utilizing the ℓ_1 norm for sparse affinity matrix construction has been widely studied, see, e.g., [KXF16, NWD16]. Even though the ℓ_1 -graph is a robust, sparse and data-adaptive method, finding a sparse representation of each data point individually, results in missing higher order relationships. In particular, the ℓ_1 norm constrained sparse representation problem tends to select one variable from a group and ignore the others when there exists a group of highly correlated variables [LSW16]. It has been shown that utilizing mixed ℓ_1 , ℓ_2 and nuclear norm regularizations balances sparsity and connectedness in a graph [YLR16]. Therefore, group sparsity techniques, such as, elastic net, are advantageous alternatives for achieving the group selection effect [LSW16, YLR16].

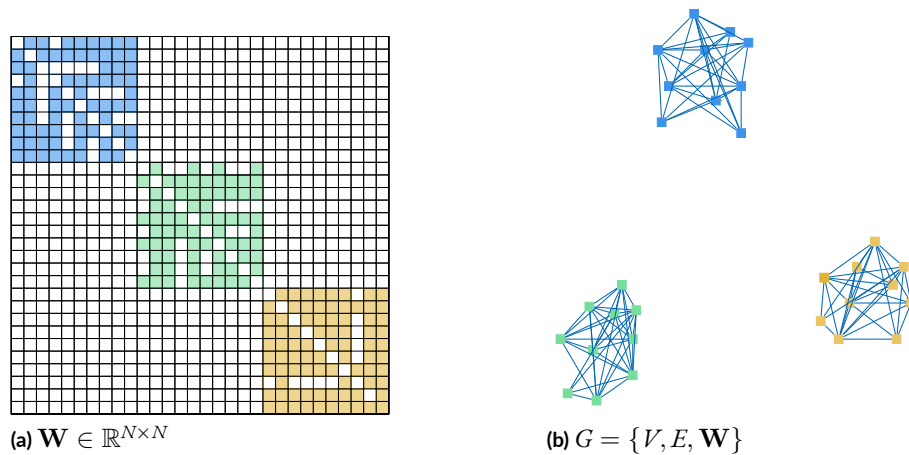


Figure 2.4: Exemplary BD affinity matrix and associated graph.

In addition to the above paradigms, sparse graphs have been constructed by imposing a structure on the affinity matrix, e.g., [LFL18, XGL17, FLX14]. Therefore, the following section provides a basic understanding of BDR, which is one of the most commonly desired structures in graph clustering algorithms.

2.6.2 BLOCK DIAGONAL REPRESENTATION (BDR)

To provide a visual understanding, an exemplary BD affinity matrix and associated graph are shown in Figure 2.4 where the three different colors in \mathbf{W} , respectively, indicate the similarity coefficients associated with the three dense clusters in G . As can be seen, the BD affinity matrix can simply be separated into different distinct blocks. From a graph clustering perspective, clusters of G , which are internally dense and unconnected to any other cluster, will provide good partitioning results with a small-valued conductance and a large-valued modularity.

The generic example also implies that block diagonally structured affinity matrices are desirable in graph clustering approaches. In particular, if the affinity matrix is BD, i.e., the similarity coefficients outside the blocks are all zero-valued, applying spectral clustering may provide perfect clustering results [LFL18]. Due to this fundamental property, the BD structure of the affinity matrix plays a crucial role in spectral-type subspace clustering approaches, e.g. [LFL18, XGL17, FLX14] that perform spectral clustering on the designed affinity matrix to assign the data points into clusters. More detailed information about state-of-the-art BDR approaches and their application to the subspace clustering problem is provided in the following sections.

A theory is only as good as its assumptions. If the premises are false, the theory has no real scientific value. The only scientific criterion for judging the validity of a scientific theory is a confrontation with the data of experience.

-Maurice Allais, 1997.

3

Sparse Graph Models for Ideal Partitioning

After explaining fundamental concepts in graph theory, the goal of this chapter is to present a proposed sparse graph model and its applications to real-world problems. To this end, first the theoretical design of the sparse graph model and its spectral analysis are provided in [Sections 3.1](#) and [3.2](#), respectively. Further, the applicability of designed sparse graphs to sparsity aware clustering is presented in [Sections 3.3](#). Lastly, two different sparsity-aware clustering methods building upon the spectral properties of the sparse graph model are introduced in [Sections 3.4](#).

3.1 THEORETICAL DESIGN OF THE SPARSE GRAPH MODEL

This section provides a theoretical understanding of the presented sparse graph model which is designed to obtain good graph partitioning results. Since a graph model consisting of internally-dense clusters that are unconnected to each other will provide a good partitioning result (for details, see [Sections 2.4](#)), the sparse graph model is defined using the advantageous properties of BD affinity matrix as follows.

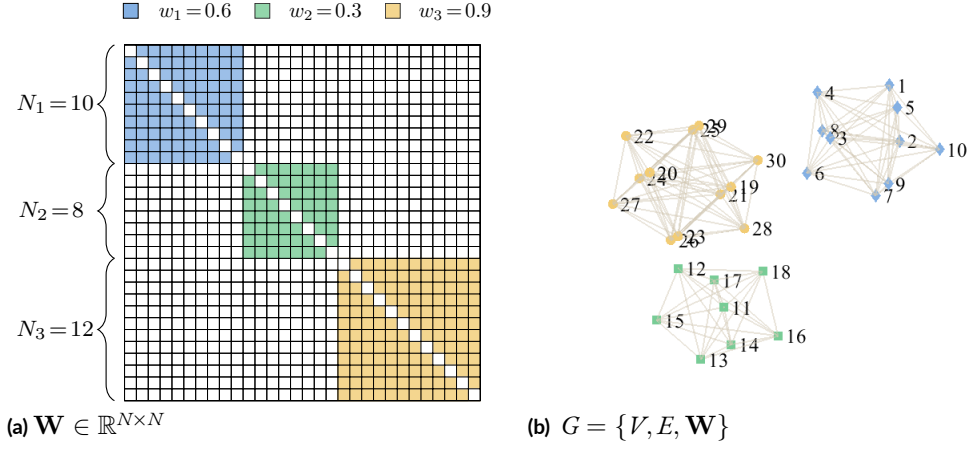


Figure 3.1: Exemplary illustration of Definition 3.1.1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).

Definition 3.1.1. (Sparse Graph for Ideal Partitioning) Let $G = \{V, E, \mathbf{W}\}$ be a sparse graph where V denotes the set of vertices, E is the set of edges and $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a K block zero-diagonal symmetric affinity matrix with blocks $\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_K$ on its diagonal. Each block \mathbf{W}_i , $i = 1, \dots, K$ is associated to a number $N_i \in \mathbb{Z}_+ > 1$ of feature vectors and concentrated around a similarity constant $w_i \in \mathbb{R}_+, i = 1, \dots, K$ with negligibly small variations. G is called a sparse graph for ideal partitioning if and only if the similarity coefficients between different blocks are all zero-valued in \mathbf{W} .

To provide a visual understanding, Definition 3.1.1 is illustrated in Figure 3.1. The colored cells in Figure 3.2a represent non-zero edge weights that the blocks are concentrated around. Similarly, colors in Figure 3.1b represent the cluster associations of vertices. As can be seen, for the BD affinity matrix as given in Definition 3.1.1, the sparse graph model consists of vertices that are connected to all other vertices of the same cluster while they are unconnected to the vertices of different clusters.

Definition 3.1.1 shows a special case of a BD affinity matrix for which the blocks are concentrated around constants. This assumption simplifies the spectral analysis of the sparse graph model that will be detailed in the following sections.

3.2 SPECTRAL ANALYSIS OF THE SPARSE GRAPH MODEL

This section is dedicated to introduce the spectral properties of the sparse graph model. In the sequel, the Laplacian matrix, eigenvalues, eigenvectors are computed for the given ideal model. Then, the analysis of Laplacian matrix is simplified by defining a vector that represents the Laplacian matrix as a piece-wise linear function.

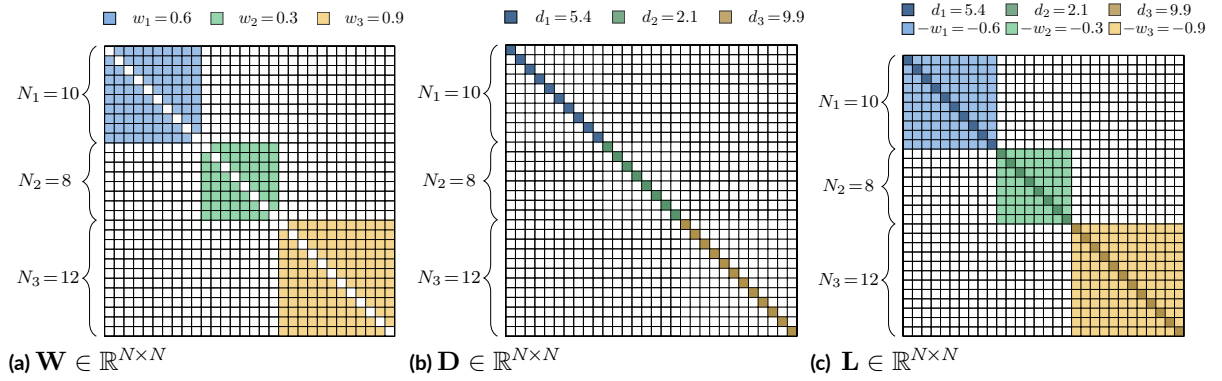


Figure 3.2: Exemplary illustration of the sparse affinity matrix \mathbf{W} and associated matrices \mathbf{D} and $\mathbf{L} \in \mathbb{R}^{N \times N}$ ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).

3.2.1 LAPLACIAN MATRIX OF THE SPARSE GRAPH MODEL

Let $\mathbf{D} \in \mathbb{R}^{N \times N}$ denote the overall edge weight matrix which is a diagonal weight matrix that is computed from sparse affinity matrix \mathbf{W} and has edge weights $d_{m,m} = \sum_{n=1}^N w_{m,n}$ on the diagonal. Then, similar to Section 2.3.1, the unnormalized sparse Laplacian matrix is computed as follows:

$$l_{m,n} = \begin{cases} (N_i - 1)w_i & \text{if } m = n \text{ and that is associated with } \mathbf{L}_i \\ -w_{m,n} & \text{otherwise} \end{cases}, \quad (3.1)$$

where $l_{m,n}$ denotes the m, n th component of the sparse Laplacian matrix for $\mathbf{L} = \mathbf{D} - \mathbf{W}$ and \mathbf{L}_i is the i th block of $\mathbf{L} \in \mathbb{R}^{N \times N}$. An exemplary plot illustrating, respectively, the sparse affinity matrix \mathbf{W} , the overall edge weight matrix \mathbf{D} and the sparse Laplacian matrix \mathbf{L} is given in Figure 3.2.

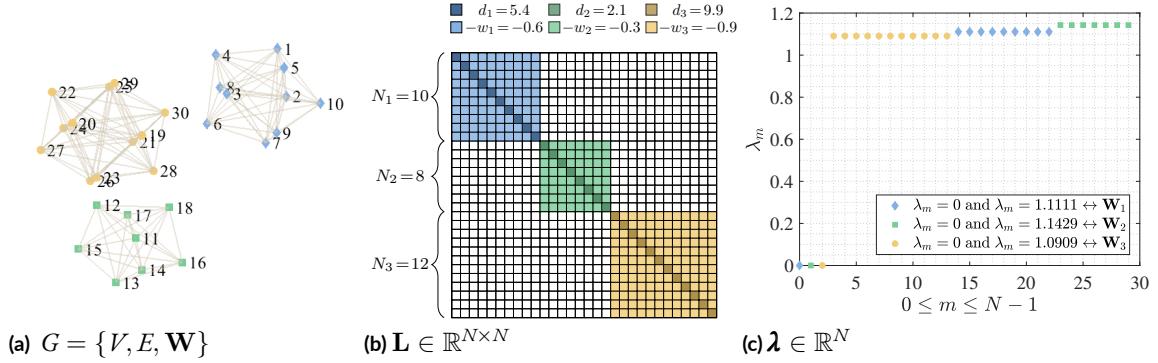


Figure 3.3: Exemplary illustration of Theorem 1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$).

3.2.2 EIGENVALUES FOR THE SPARSE GRAPH MODEL

After computing the sparse Laplacian matrix, our next step is to compute its eigenvalues which is detailed in the following theorem.

Theorem 1. Let $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a sparse affinity matrix in Definition 3.1.1 and let $\mathbf{D} \in \mathbb{R}^{N \times N}$ denote the associated matrix of overall edge weights. Assuming that $\mathbf{L} \in \mathbb{R}^{N \times N}$ is the associated sparse Laplacian matrix, its eigenvalues will be of the following form based on Eq. (2.6)¹

$$\boldsymbol{\lambda} = \text{sort} \left(\underbrace{0, \dots, 0}_K, \underbrace{\frac{N_1}{N_1-1}, \dots, \frac{N_1}{N_1-1}}_{N_1-1}, \dots, \underbrace{\frac{N_K}{N_K-1}, \dots, \frac{N_K}{N_K-1}}_{N_K-1} \right),$$

where $\boldsymbol{\lambda} \in \mathbb{R}^N$ denotes the vector of target eigenvalues and $\text{sort}(\cdot)$ is the sorting operation in ascending order.

Proof. See Appendix A.1.1.1 □

An illustration of Theorem 1 is given in Figure 3.3 for a sparse affinity matrix consisting of $K = 3$ blocks. By definition, each block is assumed to be concentrated around a constant $w_i \in \mathbb{R}_+$, e.g. $\mathbf{w} = [0.6, 0.3, 0.9]^\top$. Figure 3.3c confirms the results of Theorem 1 that for each block $i = 1, \dots, 3$, the smallest eigenvalue is zero and the remaining $N_i - 1$ eigenvalues are $\frac{N_i}{N_i-1}$.

From Theorem 1, it becomes clear that the eigenvalues contain the block size information. This valuable knowledge will be later used to learn the structure of \mathbf{W} based on the eigenvalues in Section 3.4.1.

¹For the eigenvalues of sparse Laplacian matrix \mathbf{L} based on standard eigen-decomposition, see Appendix A.1.2.

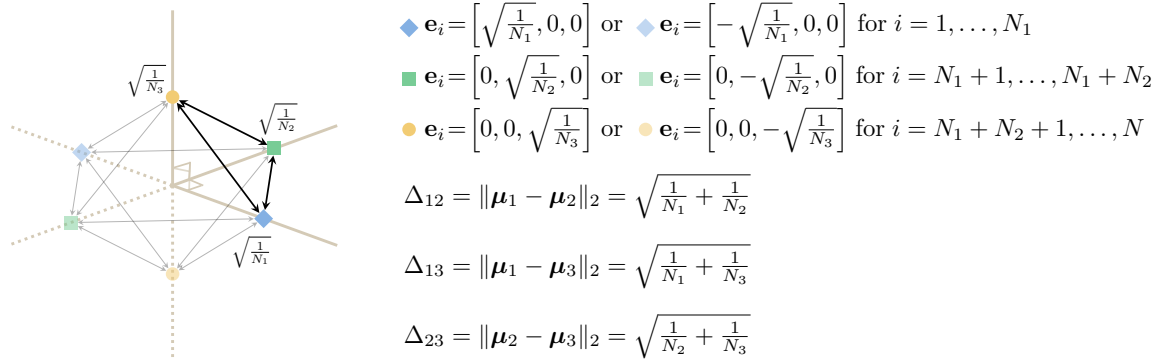


Figure 3.4: Spectral embedding according to the eigenvectors of the Laplacian matrix \mathbf{L} when $K = 3$.

3.2.3 EIGENVECTORS FOR THE SPARSE GRAPH MODEL

The third step of spectral analysis is to compute the eigenvectors of the sparse Laplacian matrix whose results are summarized in the following theorem.

Theorem 2. Let $\mathbf{L} \in \mathbb{R}^{N \times N}$ be the Laplacian matrix corresponding to a K block zero-diagonal symmetric affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ in which every block $k = 1, \dots, K$ is associated to N_k ($N_k \in \mathbb{Z}$) > 1 feature vectors and the affinities outside the blocks are zero-valued. Further, let $\mathbf{Y} = [\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{K-1}] \in \mathbb{R}^{N \times K}$ be the matrix of eigenvectors associated with the K smallest eigenvalues of \mathbf{L} . Finally, let \mathbf{e}_i , the i th row vector of \mathbf{Y} , denote the embedding vector that represents the M -dimensional i th feature vector \mathbf{x}_i in the reduced K -dimensional space. Assuming that the eigenvectors are orthonormal, the Euclidean distance between any embedding vector pairs \mathbf{e}_i and \mathbf{e}_j associated to distinct blocks k and l is equal to $\|\mathbf{e}_i - \mathbf{e}_j\|_2 = \sqrt{1/N_k + 1/N_l}$ for $k \neq l$ and $i \neq j$.

Proof. See Appendix A.1.1.2. □

Theorem 2 is illustrated in Figure 3.4 for an example consisting of $K = 3$ blocks where $\boldsymbol{\mu}_k \in \mathbb{R}^K$ denotes the cluster centroid corresponding to block $k = 1, \dots, K$. As can be seen, the Euclidean distance between embeddings of distinct blocks is a function of their block sizes.

Different from Theorem 1, Theorem 2 addresses a more general case of Definition 3.1.1 by removing the assumption that the blocks are concentrated around a similarity constant. Based on the properties of orthonormality, the embedding results of Theorem 2 show the ideal case of graph clustering approaches that are built upon the eigenvectors of Laplacian matrix, e.g., SC (for details, see Section 2.5). An exemplary application of this important property will be later provided for sparsity-aware subspace clustering in Section 3.4.2.

3.2.4 SIMPLIFYING LAPLACIAN MATRIX ANALYSIS OF THE SPARSE GRAPH MODEL

In the preceding sections, spectral analysis has been performed for $N \times N$ Laplacian matrices, which may lead to computationally heavy methods for large graphs. In this section, we therefore re-formulate the problem in $N \times 1$ vector space. In particular, assuming that \mathbf{W} is symmetric and BD^2 , the analysis is simplified by defining the vector $\mathbf{v} \in \mathbb{R}^N$ as follows

$$v_m = \sum_{n=m}^N l_{m,n}, \quad (3.2)$$

where v_m and $l_{m,n}$, respectively, denote the m th and m, n th components of \mathbf{v} and \mathbf{L} .

After determining vector \mathbf{v} , our next step is its computation for a sparse Laplacian matrix which is detailed in the following theorem.

Theorem 3. *Let $\mathbf{L} \in \mathbb{R}^{N \times N}$ be the sparse Laplacian matrix associated with the sparse affinity matrix in Definition 3.1.1. Then, it follows that the vector \mathbf{v} associated with \mathbf{L} is a piece-wise linear function of the following form*

$$v_m = f(m) = \begin{cases} (m - \ell_1)w_1 & \text{if } \ell_1 \leq m \leq u_1 \\ \vdots & \\ (m - \ell_K)w_K & \text{if } \ell_K \leq m \leq u_K, \end{cases}$$

where $\ell_1 = 1$, $u_1 = N_1$, $\ell_i = \sum_{k=1}^{i-1} N_k + 1$ and $u_i = \sum_{k=1}^i N_k$ for $i = 2, \dots, K$.

Proof. See Appendix A.1.1.3. □

An illustration of Theorem 3 is provided in Figure 3.5 for an example consisting of $K = 3$ blocks. The changepoints of \mathbf{v} define the blocks sizes and the coefficients around which the blocks are concentrated. Consequently, \mathbf{v} provides substantial information about the eigenvalues of \mathbf{L} , which will be used in Section 3.4.1 to design eigenvalue-based affinity matrix estimation methods that may be computed efficiently through the optimization in a vector space.

²A sparse matrix can be transformed into a BD form using the Reverse Cuthill-McKee (RCM) algorithm [CM69].

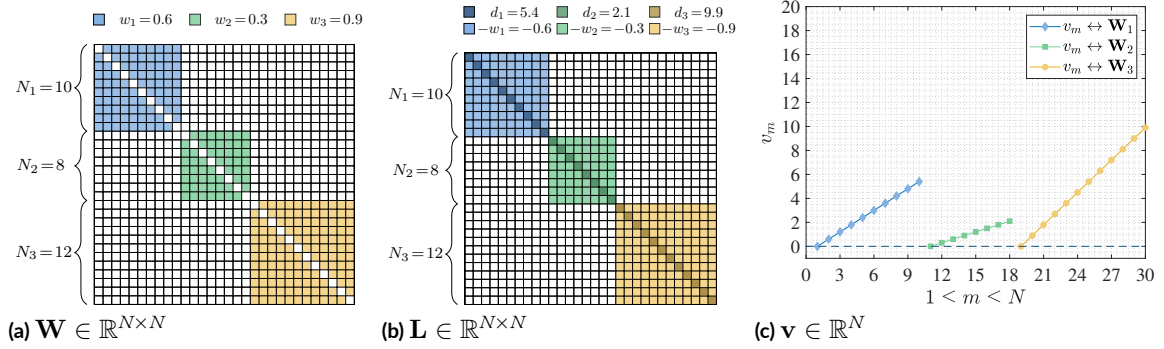


Figure 3.5: Exemplary sparse Affinity matrix, sparse Laplacian matrix and corresponding vector \mathbf{v} ($\mathbf{n} = [10, 8, 12]^T \in \mathbb{R}^K$, $N = 30, K = 3$).

3.3 ADAPTING SPARSE GRAPHS TO SPARSITY-AWARE CLUSTERING

Up to now, the determination of sparse graphs and their spectral analysis have been explained. Beyond the theoretical understanding of sparse graphs, this section discusses the applicability of sparse graphs to real-world clustering problems.

In Section 2.4, it has been explained that graph clustering approaches maximize the number of intra-cluster edges while minimizing the number of inter-cluster edges. Concordantly, this objective can be transferred to structuring an affinity matrix in which the similarity coefficients corresponding to intra-clusters edges are non-zero valued while that of inter-clusters are zero-valued. This means that having a sparse affinity matrix, such as, in Definition 3.1.1 will potentially provide good graph clustering results.

In theory, the sparse affinity matrix in Definition 3.1.1 is an $N \times N$ and the spectral properties of the sparse affinity matrix show that the eigenvalues and the vector \mathbf{v} carry all the relevant information in \mathbf{W} , such as, the block sizes and/or the similarity coefficients around which the blocks are concentrated. In practice, the data may not directly produce an affinity matrix that follows Definition 3.1.1. However, it is reasonable to use the theory as a target towards which we regularize the affinity matrix estimator. Therefore, assuming that the eigenvalues in Theorem 1 as our target, a sparse affinity matrix can be structured by approximating these target eigenvalues. Additionally, Theorem 2 shows the ideal Euclidean distance between different clusters for a clustering algorithm that is performed using the eigenvectors associated with K smallest eigenvalues of a Laplacian matrix, i.e. SC. Practically, this means that an affinity matrix providing the Euclidean distance between different clusters as in Theorem 2 is potentially a good input to the SC algorithm.

3.4 SPARSITY-AWARE CLUSTERING BASED ON SPECTRAL PROPERTIES OF THE SPARSE GRAPH MODEL

Motivated by the adaptiveness of sparse graphs to sparsity-aware clustering, this section introduces two different sparsity-aware clustering methods that are built upon, respectively, the eigenvalues of the sparse Laplacian matrix in Theorem 1 and the eigenvectors in Theorem 2.

3.4.1 EIGENVALUE-BASED SPARSITY LEVEL CONTROL FOR CLUSTERING

3.4.1.1 INTRODUCTION

The construction of an informative graph model plays a crucial role to learn the intrinsic relationships hidden in data and it has numerous applications such as in clustering/classification [OFK18, ZZL18, EV13], subspace learning [EV13, CYY09] and semi-supervised learning [LSW16, CYY09, Zhu08]. In cluster analysis, the graph model represents each feature vector as a vertex and describes the association relationships using an affinity matrix in which BD structure is a commonly desired feature [LFL18, XGL17, FLX14, EV12, LW12].

Partly motivated by the natural occurrence of block diagonally structured affinity matrices in cluster analysis, BDR has been the subject of intense scientific research. Sparse representation is one of the most common ways of constructing a BD affinity matrix [LFL18, XGL17, FLX14, EV12]. An alternative way of constructing BD affinity matrices are p -nearest neighbor graphs which are popular due to their computational simplicity [LW12]. However, a major challenge for all these methods is to determine the level of sparsity, i.e. the number of neighbors or the regularization parameter. The choice of the sparsity level has been researched by analyzing the similarity coefficients' distribution [TMZ21], via supervised learning algorithms [MDD18, GCC15], geometric interpretations [ARV09] and connectedness [NH11].

To the best of our knowledge, an unsupervised BDR method that uses the eigenvalues of a BD affinity matrix to deduce the sparsity level has not been proposed in the literature. Therefore, in [TMZ22], we first analyze the eigenvalues of the Laplacian matrix based on an ideal BD model (Theorem 1). Then, a key idea is to define a vector that represents the blocks as a piece-wise linear function (Theorem 3). This enables a graph construction algorithm building upon the piece-wise linear function that estimates the parameters of the unknown target eigenvalue vector. The proposed *eigenvalues-based block diagonal representation (EBDR)* method [TMZ22] is applied to p -nearest neighbor graph construction. In the following sections, problem formulation, details of the proposed EBDR method, a performance evaluation in comparison to popular BDR methods

and a summary are provided.

3.4.1.2 PROBLEM FORMULATION

Given a dataset of feature vectors $\mathbf{X} \in \mathbb{R}^{M \times N}$ and the number of blocks K , the goal of this work is to efficiently estimate a K block zero-diagonal symmetric affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ using the eigenvalue information from Theorem 1 and Theorem 3.

3.4.1.3 METHODOLOGY

This section proposes a method to represent the data matrix \mathbf{X} as a weighted graph G by finding a K block zero-diagonal affinity matrix \mathbf{W} whose non-zero components in the m th row/column denote the neighbors of the m th vertex. In principle, if there exists a K block zero-diagonal affinity matrix \mathbf{W} as in Theorem 1, the eigenvalues associated with the Laplacian matrix \mathbf{L} will be in the following form

$$\boldsymbol{\lambda}_o = \text{sort} \left(\underbrace{0, \dots, 0}_K, \underbrace{\frac{N_1}{N_1-1}, \dots, \frac{N_1}{N_1-1}}_{N_1-1}, \dots, \underbrace{\frac{N_K}{N_K-1}, \dots, \frac{N_K}{N_K-1}}_{N_K-1} \right), \quad (3.3)$$

where $\text{sort}(\cdot)$ denotes sorting operation in ascending order. According to Eq. (3.3), the estimation of \mathbf{W} can be cast as the following eigenvalue-based optimization program

$$\hat{\boldsymbol{\lambda}} = \boldsymbol{\lambda}^{(\hat{p})} = \underset{p_m \in \mathbf{P}}{\text{argmin}} \|\boldsymbol{\lambda}^{(p_m)} - \boldsymbol{\lambda}_o^{(p_m)}\|_2^2. \quad (3.4)$$

Here, $\hat{\boldsymbol{\lambda}}$ is the estimated vector of eigenvalues which is a function of the estimated number of neighbors (i.e., $\hat{\boldsymbol{\lambda}} = \boldsymbol{\lambda}^{(\hat{p})}$). The estimate is the minimizer of Eq. (2.6), where p_m is the m th candidate of neighbors from a given vector of candidates $\mathbf{p} = [p_1, p_2, \dots, p_{N_p}] \in \mathbb{Z}^{N_p}$. The associated affinity, overall edge weight and Laplacian matrices of dimension $\mathbb{R}^{N \times N}$ are denoted, respectively, by $\mathbf{W}^{(p_m)}$, $\mathbf{D}^{(p_m)}$ and $\mathbf{L}^{(p_m)}$. Finally, $\boldsymbol{\lambda}_o^{(p_m)} \in \mathbb{R}^N$ is the target vector of eigenvalues associated with $\mathbf{L}^{(p_m)}$ whose estimation is detailed in the following step.

3.4.1.3.1 ESTIMATION OF THE TARGET EIGENVALUE VECTOR $\boldsymbol{\lambda}_o^{(p_m)}$

Step 1) Initialization : Possible Block Sizes

Suppose that $\mathbf{v}^{(p_m)} \in \mathbb{R}^N$ denotes the vector \mathbf{v} associated with $\mathbf{W}^{(p_m)}$. Further, let $N_c^{(p_m)} \in \mathbb{Z}_+$ denote the number of changepoints and let $\boldsymbol{\tau}^{(p_m)} = [\tau_1^{(p_m)}, \tau_2^{(p_m)}, \dots, \tau_{N_c}^{(p_m)}]^\top \in \mathbb{Z}_+$ be the vector

containing corresponding locations in $\mathbf{v}^{(p_m)}$ where $\tau_0^{(p_m)} = 0$ and $\tau_{N_c+1}^{(p_m)} = N$. Then, the changepoints in $\mathbf{v}^{(p_m)}$ are detected by minimizing the following penalized least-squares function [KFE12]

$$\sum_{i=1}^{N_c^{(p_m)}+1} \sum_{n=\tau_{i-1}^{(p_m)}+1}^{\tau_i^{(p_m)}} (v_n^{(p_m)} - \hat{v}_n^{(p_m)})^2 + \beta N_c^{(p_m)}, \quad (3.5)$$

where β is a penalty parameter, $v_n^{(p_m)}$ and $\hat{v}_n^{(p_m)}$ are the n th point in the i th linear segment of $\mathbf{v}^{(p_m)}$ and the corresponding least-squares linear fit $\hat{\mathbf{v}}^{(p_m)}$, respectively. If the decrease in residual error is smaller than β , Eq. (4.21) rejects including additional changepoints while all possible changepoints are considered for $\beta = 0$. For a defined maximum number of changepoints $N_{c_{\max}} \in \mathbb{Z}_+$, which is a reasonably small number satisfying $K - 1 \leq N_{c_{\max}}$, β is increased gradually as long as the function finds fewer number of changepoints than $N_{c_{\max}}$. Accordingly, a matrix $\mathbf{N}_{\text{cand}}^{(p_m)} = [\mathbf{n}_1^{(p_m)}, \mathbf{n}_2^{(p_m)}, \dots, \mathbf{n}_\xi^{(p_m)}]^\top \in \mathbb{R}^{\xi \times K}$ whose rows denote the candidate size vectors is designed by combination of all possible size vectors with $\xi = \binom{N_c}{K-1}$. In practice, the candidate size vectors consisting the block sizes that are smaller than a defined minimum number of vertices in the blocks N_{\min} can be removed from $\mathbf{N}_{\text{cand}}^{(p_m)}$.

Step 2) Plane-based Piece-wise Linear Fit $\mathbf{v}^{(p_m)}$

Suppose that N_i denotes the size of the i th segment from a candidate vector of sizes $\mathbf{n}_{\text{cand}}^{(p_m)} = [N_1^{(p_m)}, N_2^{(p_m)}, \dots, N_K^{(p_m)}]^\top \in \mathbb{R}^K$ with $\text{cand} = 1, \dots, \xi$. Further, let $\mathbf{S}_i^{(p_m)} = [\mathbf{s}_{i_1}^{(p_m)}, \mathbf{s}_{i_2}^{(p_m)}, \dots, \mathbf{s}_{i_{N_i}}^{(p_m)}]^\top \in \mathbb{R}^{N_i \times 2}$ denote a sample matrix associated with the i th linear segment such that $\mathbf{s}_{i_n}^{(p_m)} = [i, v_{i_n}^{(p_m)}]^\top \in \mathbb{R}^2$, $n = 1, \dots, N_i$. Then, the goal of this step is to approximate $\mathbf{v}^{(p_m)}$ using a piece-wise linear function that is determined by estimating K planes, i.e.

$$\hat{\mathcal{P}}_i^{(p_m)} = \{\mathbf{s}_{i_n}^{(p_m)} | \mathbf{s}_{i_n}^{(p_m)} \in \mathbb{R}^2, (\hat{\boldsymbol{\vartheta}}_i^{(p_m)})^\top \mathbf{s}_{i_n}^{(p_m)} + \hat{\mathbf{b}}_i^{(p_m)} = 0\}, \quad i = 1, \dots, K, \quad (3.6)$$

where $\hat{\boldsymbol{\vartheta}}_i^{(p_m)} \in \mathbb{R}^2$ and $\hat{\mathbf{b}}_i^{(p_m)} \in \mathbb{R}$ denote, respectively, the normal vector and the bias associated with the estimated i th plane $\hat{\mathcal{P}}_i^{(p_m)}$. The estimation can be performed by solving the K individual ordinary eigenvalue problems as in [YYZ19]

$$\boldsymbol{\Sigma}_i^{(p_m)} \hat{\boldsymbol{\vartheta}}_i^{(p_m)} = \Lambda_i^{(p_m)} \hat{\boldsymbol{\vartheta}}_i^{(p_m)}, \quad i = 1, \dots, K \quad (3.7)$$

and

$$\hat{\mathbf{b}}_i^{(p_m)} = -(\hat{\boldsymbol{\vartheta}}_i^{(p_m)})^\top \boldsymbol{\mu}_i^{(p_m)}, \quad i = 1, \dots, K, \quad (3.8)$$

where $\Lambda_i^{(p_m)} \in \mathbb{R}$ is the smallest eigenvalue associated with the i th plane, and $\Sigma_i^{(p_m)} \in \mathbb{R}^{2 \times 2}$ and $\mu_i^{(p_m)} \in \mathbb{R}^2$ are, respectively, the covariance matrix and the mean vector of $S_i^{(p_m)}$.³ Then, using the estimated parameters of the K planes, each segment in the vector $\mathbf{v}^{(p_m)}$ is estimated as follows

$$(\hat{\mathcal{S}}_i^{(p_m)})^\top \begin{bmatrix} n \\ \hat{v}_{i_n}^{(p_m)} \end{bmatrix} + \hat{\mathbf{b}}_i^{(p_m)} = 0, \quad i = 1, \dots, K, \quad n = 1, \dots, N_i^{(p_m)}, \quad (3.9)$$

where $\hat{v}_{i_n}^{(p_m)}$ denotes the n th estimated point in the i th segment. Assuming that for each $\mathbf{n}_{\text{cand}}^{(p_m)} \in \mathbf{N}_{\text{cand}}^{(p_m)}$, $\text{cand} = 1, \dots, \xi$ there exists a piece-wise linear function, the size vector is optimized as follows:

$$\hat{\mathbf{n}} = \underset{\mathbf{n}_{\text{cand}}^{(p_m)} = \mathbf{n}_1^{(p_m)}, \dots, \mathbf{n}_\xi^{(p_m)}}{\text{argmin}} \quad \|\mathbf{v}^{(p_m)} - \hat{\mathbf{v}}_{\text{cand}}^{(p_m)}\|_2^2, \quad (3.10)$$

where $\hat{\mathbf{n}}$ denotes the estimated block size vector and $\hat{\mathbf{v}}_{\text{cand}}^{(p_m)} \in \mathbb{R}^N$ is the estimate of vector $\mathbf{v}^{(p_m)}$ associated with $\mathbf{n}_{\text{cand}}^{(p_m)}$. The proposed EBDR method is summarized for the p -nearest neighbor graphs in Algorithm 2.

3.4.1.4 EXPERIMENTAL RESULTS

In this section, EBDR is benchmarked against three state-of-the-art BDR approaches, i.e. subspace segmentation with BD prior (BDSSC) [FLX14], BDR using matrix \mathbf{B} (BDR-B) [LFL18] and implicit block diagonal low-rank representation (IBDLR) [XGL17], and robust kernel low-rank representation (RKLRR) method [XTX15] that can be reduced to the BD for independent subspaces and the initial matrix containing all neighbors \mathbf{W}^{N-1} . The performance of different methods is analyzed in terms of their average clustering accuracy \bar{p}_{acc} and computation time t using the following real-world datasets: Fisher's iris (Fisheriris) [Fis36], radar-based human gait (Gait) [TMZ20, SAZ19], ovarian cancer (O. Cancer) [CFR04] and person identification (Person Id.) [TSM18]. The parameters of the competitors are manually tuned to the best possible \bar{p}_{acc} by using 500 samples in total. Then, t is summarized for 100 Monte Carlo experiments using the selected parameters. In all experiments, the initial affinity matrix \mathbf{W}^{n-1} is computed using cosine similarity and SC is performed as partitioning method. EBDR is computed using the following parameters: $N_{\min} = \frac{N}{2K}$, $\mathbf{p} = [5, 10, \dots, N-1]$, $N_{\text{cmax}} \in [K-1, \dots, 20]$.

³The optimal solution to the plane-based piece-wise linear fit problem can be uniquely determined by K covariance matrices and means of the corresponding K clusters (K blocks for our case) when the objective function reaches the optimum. For a detailed information, see Corollary 1-2 in [YYZ19].

Algorithm 2: p -nearest Neighbor Graph Construction

Input: $\mathbf{X} \in \mathbb{R}^{M \times N}$, $\mathbf{p} \in \mathbb{R}^{N_p}$, $N_{c_{\max}}$, N_{\min} (optional)

for $p_m = p_1, p_2, \dots, p_{N_p}$ **do**

Eigenvalue vector $\boldsymbol{\lambda}^{(p_m)}$

 Compute $\mathbf{W}^{(p_m)} \in \mathbb{R}^{N \times N}$ s.t. $w_{m,n} = \mathbf{x}_m^\top \mathbf{x}_n$

 Compute $\boldsymbol{\lambda}^{(p_m)} \in \mathbb{R}^N$ using Eq. (2.6)

Target Eigenvalue Vector Estimation $\boldsymbol{\lambda}_o^{(p_m)}$

Step 1) Initialization: Possible block sizes

 Compute $\mathbf{N}_{\text{cand}}^{(p_m)} \in \mathbb{R}^{\xi \times K}$ using Eq. (3.5)

Step 2) Plane-based Piece-wise Linear Fit

for $\mathbf{n}_{\text{cand}}^{(p_m)} = \mathbf{n}_1^{(p_m)}, \dots, \mathbf{n}_\xi^{(p_m)}$ **do**

for $i = 1, \dots, K$ **do**

 Calculate $\boldsymbol{\Sigma}^{(p_m)} \in \mathbb{R}^{2 \times 2}$ and $\boldsymbol{\mu}^{(p_m)} \in \mathbb{R}^2$ for $\mathbf{S}_i^{(p_m)}$

 Find $\hat{\boldsymbol{\vartheta}}_i^{(p_m)} \in \mathbb{R}^2$ and $\hat{b}_i^{(p_m)} \in \mathbb{R}$ via Eq. (3.7)-(3.8)

 Substitute $\hat{\boldsymbol{\vartheta}}_i^{(p_m)}$, $\hat{b}_i^{(p_m)}$ in Eq. (3.9) to find $\hat{\mathbf{v}}_i^{(p_m)} \in \mathbb{R}^{N_i^{(p_m)}}$

end

 Form $\hat{\mathbf{v}}_{\text{cand}}^{(p_m)} = [(\hat{\mathbf{v}}_1^{(p_m)})^\top, (\hat{\mathbf{v}}_2^{(p_m)})^\top, \dots, (\hat{\mathbf{v}}_K^{(p_m)})^\top]^\top \in \mathbb{R}^N$

end

 Estimate the block size vector \mathbf{n} using Eq. (3.10)

 Design $\boldsymbol{\lambda}_o^{(p_m)}$ using Eq. (3.3)

end

Compute $\hat{\boldsymbol{\lambda}} = \boldsymbol{\lambda}^{\hat{p}}$ using Eq. (3.4) and obtain $\mathbf{W}^{(\hat{p})}$

Output: $G^{(\hat{p})} = \{V, E^{(\hat{p})}, \mathbf{W}^{(\hat{p})}\}$

Dataset	K	N	\mathbf{n}	p^*	\hat{p}
Fisheriris [Fis36]	3	150	$[50, 50, 50]^\top$	50	50
Gait [SAZ19, TMZ20]	5	800	$[160, 160, 160, 160, 160]^\top$	160	165
O. Cancer [CFR04]	2	216	$[95, 121]^\top$	100	110
Person Id. [TSM18]	4	187	$[38, 40, 47, 62]^\top$	45	45

Table 3.1: Numerical results for real-world datasets ($N_{c_{\max}} = 8$).

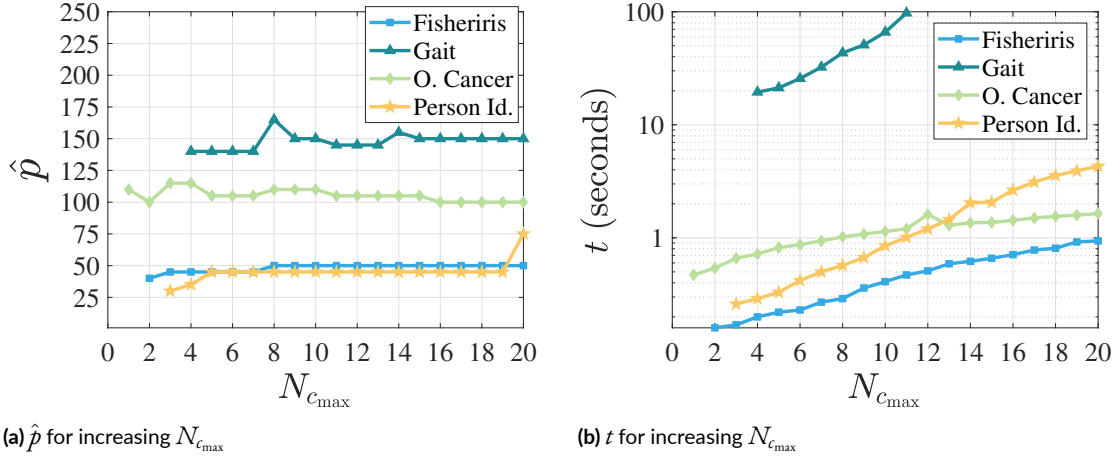


Figure 3.6: Numerical results for the parameter $N_{c_{\max}}$.

In Table 3.1, the EBDR application to p -nearest neighbor graphs is benchmarked using real-world datasets. The number of neighbors that provided the best \bar{p}_{acc} is denoted by p^* . As can be seen, \hat{p} provided similar results to p^* in all cases. To analyze the effect of $N_{c_{\max}}$ on \bar{p}_{acc} and t , the estimated nearest neighbor values and computation time are shown for different $N_{c_{\max}}$ values in Figure 3.6a and Figure 3.6b, respectively. The results demonstrate that EBDR approximates p^* values even for a small number of samples. However, large values of $N_{c_{\max}}$ result in high computational cost, especially in outlier contaminated datasets, e.g. Gait. Lastly, comparisons are drawn in terms of \bar{p}_{acc} and t for the different methods in Table 3.2 and Table 3.3, respectively.

The clustering accuracy \bar{p}_{acc} that has been detailed in Table 3.2 shows the best possible performances for the competitors when the level of sparsity (i.e. the penalty parameter) has been optimally selected. In particular, the competitor clustering accuracy results are the best results according to an oracle selected penalty parameter/s from a grid, while estimating the level of sparsity is part of the optimization for the proposed method. Therefore, Table 3.2 shows that EBDR improves the performance of \mathbf{W}^{N-1} and performs similar to the best results of the competitors

Dataset	\mathbf{W}^{N-1}	BDSSC	BDR-B	RKLRR	IBDLR	EBDR
Fisheriris [Fis36]	78.00	96.00	97.33	94.67	94.67	98.00
Gait [SAZ19]	79.63	83.13	86.25	86.75	82.75	80.75
O. Cancer [CFR04]	75.00	81.48	86.57	89.35	77.31	79.17
Person Id. [TSM18]	x	95.72	97.33	95.72	95.72	97.33

Table 3.2: $\bar{p}_{\text{acc}}(\%)$ for real-world datasets. ‘x’ denotes the results that produce complex-valued eigenvectors, $N_{c_{\max}} = 8$. The numbers indicate the best \bar{p}_{acc} for the competitors.

Dataset	BDSSC	BDR-B	RKLRR	IBDLR	EBDR(\hat{p})	EBDR
Fisheriris [Fis36]	0.174	0.041	0.208	0.573	0.015	0.295
Gait [SAZ19]	6.132	4.748	5.394	826.2	0.495	43.29
O. Cancer [CFR04]	2.590	0.099	3.471	1.397	0.041	1.018
Person Id. [TSM18]	0.235	0.489	0.013	0.564	0.019	0.550

Table 3.3: $t(\text{seconds})$ for real-world datasets. Except for EBDR the level of sparsity assumed to be known and it is defined as \hat{p} for EBDR(\hat{p}). $N_{c_{\max}} = 8$ in all cases.

including an unsupervised sparsity parameter estimation p . In terms of t , the proposed method shows a significantly better performance when the level of sparsity is assumed to be known for the competitors. Even when including the nearest neighbor number estimation, EBDR is competitive in terms of speed, which can be further reduced by tuning \mathbf{p} , $N_{c_{\max}}$ and $n_{k_{\min}}$.

3.4.1.5 SUMMARY

The eigenvalues associated with the block affinity matrix are analyzed for the generalized eigen-decomposition to demonstrate the importance of eigenvalues in block affinity matrix design. Based on our theoretical findings on the eigenvalues and the vector \mathbf{v} , we proposed EBDR, which estimates the number of neighbors by approximating the target eigenvalues. EBDR was benchmarked on different real-world datasets and it showed promising performance compared to four optimally tuned popular approaches in terms of both computation time and the accuracy.

3.4.2 EIGENVECTOR-BASED SPARSITY LEVEL CONTROL FOR CLUSTERING

3.4.2.1 INTRODUCTION

Determining an embedding so that the data points lie in a union of low-dimensional subspaces is crucial in many real-world problems such as in clustering [SLL19, LFL18, XGL17, XTX15, FLX14, EV13], supervised learning [GGK13] and semi-supervised learning [QWZ21, LLZ15]. In particular, subspace clustering has numerous applications e.g. motion segmentation [TV17, RTV08], face clustering [LFL18, EV13], image segmentation [LKJ20] and community clustering in social networks [CJS14]. Motivated by its broad range of applications, SC has been the subject of much research, which can loosely be divided into four main categories, i.e., iterative [RA17], algebraic [TV17, TV17], statistical [RTV08] and SC-based methods [LFL18, XGL17, XTX15, FLX14, EV13, LY11]. In recent years, the latter have attracted increasing interest due to their simplicity and promising performance [LFL18, XGL17].

As discussed in Section 2.6.1, the first step of SC-based methods is to compute an affinity matrix. Block diagonally structured affinity matrices constitute an informative prior, that is frequently used (e.g., [TMZ22, LFL18, XGL17, FLX14]). A popular strategy to construct a BD models is to represent the data as a linear combination of feature vectors while regularizing the affinity matrix coefficients, e.g. with an ℓ_1 , ℓ_2 or nuclear norm [LFL18, XGL17, FLX14]. Recent methods apply mixed norms, such as, the elastic net, which have the advantage of providing a tradeoff between sparsity and connectedness [XGL17, YLR16]. A major challenge for all these approaches is to determine the appropriate level of sparsity which plays a crucial role in SC performance. Different methods building upon supervised learning algorithms [MDD18, GCC15], similarity coefficients' distribution [TMZ21], geometric interpretations [ARV09], connectedness [NH11] and eigenvalues [TMZ22] have been proposed; however, no optimal approach exists, especially in the presence of outliers.

In [TMO23], we have proposed a *Sparsity-Aware Block Diagonal Representation (SABDR)* method to robustly estimate the appropriate level of sparsity for subspace clustering. The proposed SABDR approach leverages upon the geometrical analysis of the low-dimensional structure in SC. In particular, the derived Euclidean distances between the embeddings of different clusters are utilized to construct the BD affinity matrix. Further, we propose a computationally efficient density-based clustering (Con-DBSCAN) algorithm, to obtain a robust estimate of the between-clusters distances that are associated with an available affinity matrix. Unlike the original DBSCAN [EKS96], by leveraging upon Theorem 2 that describes the geometry of the

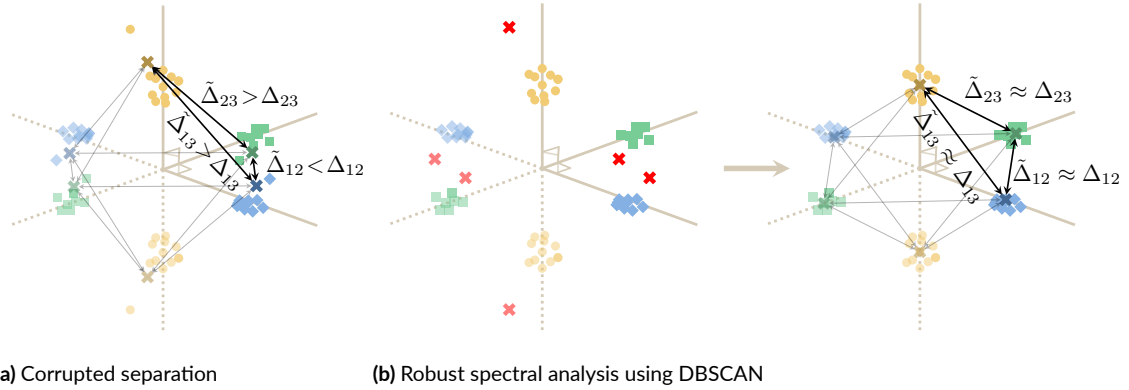


Figure 3.7: Robustness in spectral analysis.

embeddings, Con-DBSCAN determines the neighborhood search radius around given points based on their connectedness, therewith leveraging the derived geometric information. This, in contrast to DBSCAN, enables Con-DBSCAN to efficiently expand clusters with multiple embedding vectors in a single iteration. The proposed modification leads to a considerable speed-up without any performance loss. Building upon our theoretic analysis in Section 3.2, we develop a regularization parameter [LFL18] selection by re-formulating the sparsity level selection problem as an approximation of the target between-clusters distances.

3.4.2.2 MOTIVATION : DBSCAN FOR ROBUST SPECTRAL ANALYSIS

If the affinity matrix of the data is BD, SC may provide excellent results. Furthermore, according to Theorem 2, a BD affinity matrix will lead to densely connected clusters in the embedding space. Hence, a density-based clustering approach, such as, DBSCAN [EKS96], is a natural approach to achieve a BD structure. However, in real-world scenarios the data includes outliers and heavy-tailed noise which may obscure the distance between embeddings of different clusters. Therefore, beyond its computational efficiency that has made DBSCAN very popular, we build upon its intrinsic outlier detection ability to increase robustness for spectral analysis. Figure 3.7 illustrates this with an example of $K = 3$ clusters that are hidden in a matrix of corrupted eigenvectors $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2]$. Even though an appropriate level of sparsity provides densely connected clusters, the outliers obscure the distance between different clusters as it is shown in Figure 3.7a. By contrast, as illustrated in Figure 3.7b, DBSCAN identifies the outliers and robustly estimates the between-clusters distance information.

3.4.2.3 PROBLEM FORMULATION

Given a dataset of feature vectors $\mathbf{X} \in \mathbb{R}^{M \times N}$, the aim of the method that is described in the following is to robustly and efficiently find a K block zero-diagonal symmetric affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ to cluster \mathbf{X} using the spectral information of BD affinity matrices.

3.4.2.4 METHODOLOGY

This section details the proposed SABDR method which estimates a BD affinity matrix in three steps that are detailed in the following sections.

Step 1) Affinity Matrix Construction

Among numerous affinity matrix design methods such as [LFL18, XGL17, XTX15, FLX14, EV13], this method adapts the BDR method in [LFL18], in which the proposed K -block regularizer promotes a nonnegative symmetric matrix to be K -BD so that the spectral analysis described in Section 3.2 is directly applicable.

Let $\mathbf{W}^{(m)}$ and $\mathbf{L}^{(m)} \in \mathbb{R}^{N \times N}$, respectively, be the affinity and Laplacian matrix that are computed by using as a regularization parameter pair $\mathbf{p}_m = [p_{m,1}, p_{m,2}]^T$ from a matrix of candidate regularization parameter pairs $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_p}] \in \mathbb{R}^{2 \times N_p}$. Assuming that for every \mathbf{p}_m there exists a matrix of eigenvectors⁴ $\mathbf{Y}^{(m)} \in \mathbb{R}^{N \times K}$, the following sections present the selection of an appropriate \mathbf{p}_m based on robust spectral analysis of $\mathbf{Y}^{(m)}$ with the proposed Con-DBSCAN.

Step 2) Block Size Estimation using Con-DBSCAN

Step 2.1) Parameter Definition: ε

It follows from Theorem 2 that there exists a specific level of sparsity that allows for an embedding, such that the distance between embeddings of the same cluster is minimal while the distance between embeddings of different clusters is maximal. This important result implies that there exists a minimum neighborhood search radius ε that will provide these highly dense clusters.

To provide a visual understanding, the geometric definition of a minimum search radius is shown in Figure 3.8. Considering a pair of clusters, the two clusters are assumed to have a maximum N_{\max} and a minimum N_{\min} number of samples based on the information that the maximum block size results in the minimum distance from the origin and vice-versa. Clearly, N_{\max} can achieve its greatest value for $N_{\max} = N - N_{\min}$. Then, using Theorem 2, the minimum ball can be simply calculated for the large cluster as $\varepsilon = \sqrt{1/(N - N_{\min}) + 1/N_{\min}} - \sqrt{1/N_{\min}}$.

⁴If the obtained set is not orthonormal, the Gram-Schmidt algorithm [CK10] can be used.

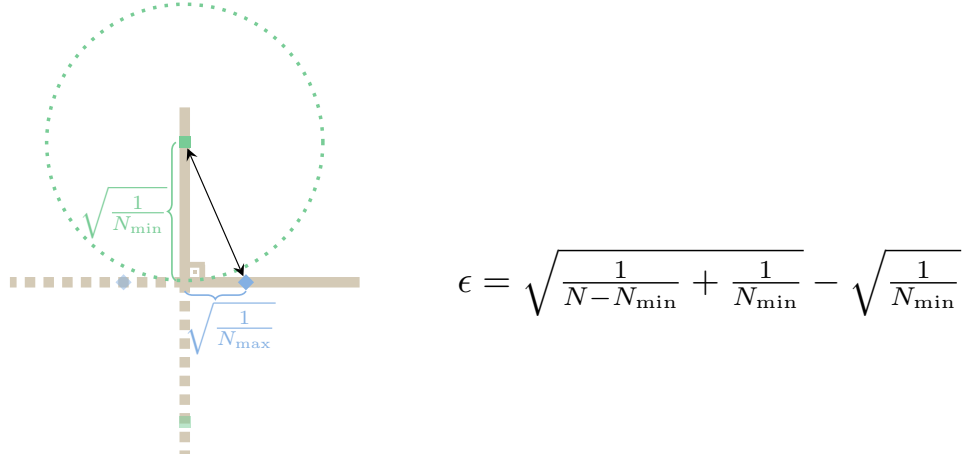


Figure 3.8: Exemplary illustration of ε definition.

Step 2.2) Parameter Definition: N_{\min}

As the parameter ε is a function of the minimum number of points N_{\min} , this section discusses finding N_{\min} . To begin, DBSCAN assigns a sample to be an outlier if its ε -neighborhood does not contain at least N_{\min} neighbors which means that a large value of N_{\min} results in assigning many samples as outlier. On the other hand, a large value of N_{\min} increases the neighborhood search radius which may result in assigning many samples into a big cluster and the remaining samples as outliers. Therefore, N_{\min} must be a reasonably small number which can be easily defined if K is known. In particular, the parameter can be gradually increased as long as the clustering results provide K clusters.

Step 2.3) Con-DBSCAN Algorithm

For a matrix $\mathbf{E}^{(m)} = (\mathbf{Y}^{(m)})^\top = [\mathbf{e}_1^{(m)}, \mathbf{e}_2^{(m)}, \dots, \mathbf{e}_N^{(m)}] \in \mathbb{R}^{K \times N}$ containing embedding vectors associated with \mathbf{p}_m , the goal of this section is to assign embedding vectors into mutually exclusive clusters using the proposed Con-DBSCAN. As in the original DBSCAN [EKS96], the method starts with ε -neighborhood computation of an unlabelled embedding vector $\mathbf{e}_n^{(m)} \in \mathbf{E}^{(m)}$, $n = 1, \dots, N$ as

$$\mathcal{N}_n^{(m)} = \{\mathbf{e}_r^{(m)} \in \mathbb{R}^K : \|\mathbf{e}_n^{(m)} - \mathbf{e}_r^{(m)}\|_2 < \varepsilon\}. \quad (3.11)$$

Here, $\mathcal{N}_n^{(m)}$ is the ε -neighborhood set of $\mathbf{e}_n^{(m)}$ and $\mathbf{e}_r^{(m)}$ is the r th embedding result with $r = 1, \dots, N$ and $r \neq n$. Similar to [EKS96], Con-DBSCAN assigns $\mathbf{e}_n^{(m)}$ into a cluster $c^{(m)}$ if its $\mathcal{N}_n^{(m)}$ includes more neighbors than N_{\min} . However, there is an important difference in

how the neighbors are included. While DBSCAN [EKS96] must compute the ε -neighborhood for every neighbor and iteratively expand cluster $c^{(m)}$ with embedding vectors, using Theorem 2, Con-DBSCAN expands the cluster $c^{(m)}$ in a single iteration, which leads to a considerable speed-up without any loss of performance compared to the original DBSCAN. More specifically, Con-DBSCAN exploits the derived geometric information by comparing the connectedness of a candidate neighbor to that of the least connected embedding in $c^{(m)}$ which are, respectively, defined by

$$\kappa_{\text{cand}}^{(m)} = \sum_{n \in c^{(m)}} w_{\text{cand},n}^{(m)}, \quad (3.12)$$

and

$$\kappa_{\text{min}}^{(m)} = \min \left\{ \kappa_n^{(m)} = \sum_{r \in c^{(m)}} w_{n,r}^{(m)} \right\}, \quad (3.13)$$

where $\kappa_{\text{cand}}^{(m)}$ denotes the connectedness of a candidate neighbor embedding $\mathbf{e}_{\text{cand}}^{(m)} \in \mathcal{N}_n^{(m)}$, $w_{n,r}^{(m)}$ is the n, r th similarity coefficient in $\mathbf{W}^{(m)}$, $\kappa_n^{(m)}$ is the connectedness associated with the n th embedding vector $\mathbf{e}_n^{(m)}$, and $\kappa_{\text{min}}^{(m)}$ is the minimum connectedness in cluster $c^{(m)}$. Based on the embedding idea that connected vertices are embedded closely [BN01], the method expands clusters using all highly connected neighbors $\forall \kappa_{\text{cand}}^{(m)} \in \mathcal{N}_n^{(m)}$, such that, $\kappa_{\text{cand}}^{(m)} > \kappa_{\text{min}}^{(m)}$. Then, it iterates the ε -neighborhood computation on the unlabelled neighbors and repeats the connectedness-based expansion until no new neighbors that can be assigned to $c^{(m)}$ are found.

3) Sparsity Level Estimation

Let $\{\hat{N}_1^{(m)}, \hat{N}_2^{(m)}, \dots, \hat{N}_K^{(m)}\} \in \mathbb{Z}_+$ be the block sizes associated with \mathbf{p}_m that have been estimated in Section 3.4.2.4. Now, using Theorem 2, the components of the target between-clusters matrix estimate $\hat{\Delta}_{\mathbf{T}}^{(m)}$ are determined by

$$\hat{\Delta}_{\mathbf{T}_{i,j}}^{(m)} = \begin{cases} \sqrt{\frac{1}{\hat{N}_i} + \frac{1}{\hat{N}_j}}, & \text{if } i \neq j \\ 0, & \text{otherwise} \end{cases}. \quad (3.14)$$

where $\hat{\Delta}_{\mathbf{T}_{i,j}}^{(m)}$ is the i, j th component of $\hat{\Delta}_{\mathbf{T}}^{(m)}$ that represents the target Euclidean distance between cluster $i = 1, \dots, K$ and $j = 1, \dots, K$. Similarly, the components of the between-clusters matrix

estimate $\hat{\Delta}^{(m)}$ are computed by

$$\hat{\Delta}_{i,j}^{(m)} = \begin{cases} \|\hat{\boldsymbol{\mu}}_i^{(m)} - \hat{\boldsymbol{\mu}}_j^{(m)}\|_2, & \text{if } i \neq j \\ 0, & \text{otherwise} \end{cases}, \quad (3.15)$$

where $\hat{\Delta}_{i,j}^{(m)}$ is the i, j th component of $\hat{\Delta}^{(m)}$ denoting the Euclidean distance between i th and j th estimated cluster centroids $\hat{\boldsymbol{\mu}}_i^{(m)}$ and $\hat{\boldsymbol{\mu}}_j^{(m)} \in \mathbb{R}^K$, respectively. Since for every \mathbf{p}_m there exists a distance matrix $\hat{\Delta}^{(m)}$, the appropriate regularization parameter pair controlling the sparsity level can be estimated as follows:

$$\hat{\mathbf{p}} = \underset{\mathbf{p}_m \in \mathbf{P}}{\operatorname{argmin}} \|\hat{\Delta}_T^{(m)} - \hat{\Delta}^{(m)}\|_F \quad (3.16)$$

The proposed SABDR for subspace clustering is summarized in Algorithm 3.

3.4.2.5 EXPERIMENTAL RESULTS

In this section, the subspace clustering performance of SABDR [TMO23] is benchmarked against five state-of-the-art affinity matrix construction methods, i.e., sparse subspace clustering (SSC) [EV13], elastic net subspace clustering (EnSC) [YLR16], BDSSC [FLX14], BDR-B [LFL18], IBDLR [XGL17] and RKLRR [XTX15] using the real-world data sets of face, object and handwritten digit images. The application details are as follows.

1. *ORL* [SH94]: As in [XGL17], 400 face images of 40 different subjects are resized to 32×32 and \mathbf{X} of size 1024×400 is computed.
2. *JAFFE* [LAK98]: Similarly, 213 images of 10 subjects are resized to 64×64 pixels and \mathbf{X} of size 4096×213 is obtained.
3. *COIL20* [NNM95]: \mathbf{X} of size 1024×400 , whose column vectors contain 32×32 down-sampled images, is generated by selecting 20 images randomly for every object.
4. *USPS* [Hul94]: \mathbf{X} with $M = 256$ and $N = 500$ is generated by randomly selecting 50 handwritten digit images of size 16×16 as feature vectors for every digit.

As in [LFL18], performance analysis of every application is conducted for an increasing value of K , e.g., $K = \{2, 3, 5, 8, 10\}$ using 100 randomly selected subject combinations. To reduce the

Algorithm 3: SABDR-based Subspace Clustering

Input: $\mathbf{X} \in \mathbb{R}^{M \times N}$, $\mathbf{P} \in \mathbb{R}^{2 \times N_p}$, N_{\min} (See, Sec. 3.4.2.4), K (optional)

for $\mathbf{p}_m = \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_p}$ **do**

Step 1) Affinity Matrix Construction:

Obtain the affinity matrix $\mathbf{W}^{(m)} \in \mathbb{R}^{N \times N}$, e.g. using [LFL18].

Compute $\mathbf{Y}^{(m)} \in \mathbb{R}^{N \times K}$ using Eq. (2.6).

Step 2) Block Size Estimation using Con-DBSCAN:

Calculate ε as described in Section 3.4.2.4.

while *there exists an unlabelled embedding vector* **do**

Select the first unlabelled embedding $\mathbf{e}_n^{(m)}$ and compute the ε -neighborhood $\mathcal{N}_n^{(m)}$ via Eq. (3.11).

Initialize the cluster label $c^{(m)}$.

if *the number of ε neighbors* $> N_{\min}$ **then**

Expand cluster $c^{(m)}$ using Eq. (3.12) and Eq. (3.13).

else

Assign $\mathbf{e}_n^{(m)}$ as outlier.

end

end

Step 3) Sparsity Level Estimation:

Calculate $\hat{\Delta}_T^{(m)}$ and $\hat{\Delta}^{(m)}$ using Eq. (3.14) and Eq. (3.15).

Update the estimate based on Eq (3.16).

end

Obtain associated $\hat{\mathbf{Y}}^{(\hat{\mathbf{p}})}$ and perform K -means.

Output: A vector of estimated cluster labels $\hat{\mathbf{c}}$

Subspace Performances on ORL Data Set							
Min-Max Average Clustering Accuracy for Different Regularization Parameters							
K	SSC	EnSC	BDSSC	RKLRR	IBDLR	BDR-B	SABDR
2	52.6 - 87.4	53.9 - 60.4	51.3 - 66.6	54.7 - 64.1	x	55.0 - 97.0	95.3
3	36.7 - 88.1	36.7 - 62.3	36.8 - 59.4	36.7 - 57.5	36.7 - 68.9	36.7 - 92.8	88.5
5	22.0 - 84.3	22.0 - 67.2	23.3 - 48.1	22.0 - 52.7	22.0 - 64.4	22.0 - 84.8	86.6
8	13.8 - 83.0	13.8 - 71.5	15.9 - 40	13.8 - 69.7	13.8 - 73.8	13.8 - 82.1	80.2
10	11.0 - 81.4	11.0 - 70.5	13.2 - 35.3	11.0 - 69.8	11.0 - 73.1	11.0 - 80.6	78.9

Table 3.4: Face clustering performance of different BDR methods on ORL data set. 'x' denotes the failed results.

Subspace Performances on JAFFE Data Set							
Min-Max Average Clustering Accuracy							
K	SSC	EnSC	BDSSC	RKLRR	IBDLR	BDR-B	SABDR
2	51.4 - 99.2	52.0 - 58.7	50.7 - 68.6	52.0 - 61.8	51.4 - 63.6	52.1 - 99.5	97.5
3	35.0 - 97.7	35.0 - 68.9	35.3 - 71.1	35.1 - 53.9	35.1 - 63.7	35.1 - 97.8	95.0
5	21.4 - 97.2	21.4 - 85.3	22.4 - 87.4	21.4 - 70.3	21.4 - 84.2	21.4 - 97.4	96.6
8	13.5 - 93.4	13.5 - 82.8	15.1 - 93.1	13.5 - 85.9	13.5 - 87.7	13.5 - 92.5	89.2
10	10.8 - 85.0	10.8 - 78.9	12.7 - 85.4	10.8 - 84.0	10.8 - 85.4	10.8 - 85.9	76.5

Table 3.5: Face clustering performance of different BDR methods on JAFFE data set. ‘x’ denotes the failed results.

Subspace Performances on COIL20 Data Set							
Min-Max Average Clustering Accuracy for Different Regularization Parameters							
K	SSC	EnSC	BDSSC	RKLRR	IBDLR	BDR-B	SABDR
2	52.2 - 93.8	52.4 - 62.8	51.1 - 71.8	52.5 - 60.9	52.2 - 68.8	52.7 - 96.6	95.2
3	35.0 - 88.3	35.0 - 65.0	35.4 - 67.8	35.0 - 52.7	35.0 - 67.4	35.4 - 91.2	83.3
5	21.0 - 85.5	21.0 - 74.8	22.3 - 69.5	21.0 - 53.7	21.0 - 70.6	22.1 - 88.4	83.2
8	13.1 - 79.9	13.1 - 76.7	15.3 - 72.2	13.1 - 66.2	13.1 - 70.6	14.3 - 81.3	73.3
10	10.5 - 76.3	10.5 - 73.5	12.8 - 70.2	10.5 - 65.7	10.5 - 67.9	11.6 - 76.8	70.3

Table 3.6: Object clustering performances of different BDR methods on COIL20 data set.

cost, the feature spaces are, respectively, reduced to 10, 8, 10 and 13 using Principal Component Analysis (PCA), since using a larger feature space did not provide significant improvements. For the competing methods, the regularization parameters are manually tuned on a grid of 50 values. Finally, SC [Lux07] is performed and the performance is summarized for the average clustering accuracy \bar{p}_{acc} . The performance of SABDR is analyzed for the default parameter choice $N_{min} = N/(2K)$, except for $K = 8$ and $K = 10$ for the USPS data set in Table 3.7, where the parameter was increased until it computes K distinct clusters (see Sec. 3.4.2.4). A MATLAB implementation of SABDR is available at:

<https://github.com/A-Tastan/SABDR>

Tables. 3.4-3.7 summarize the obtained results on face, object and handwritten digit clustering, respectively. As can be seen, SABDR in nearly all cases reaches a performance close to, or in some cases even better than that of optimally tuned competitors. This demonstrates its excellent sparsity level estimation performance.

Subspace Performances on USPS Data Set							
Min-Max Average Clustering Accuracy for Different Regularization Parameters							
K	SSC	EnSC	BDSSC	RKLRR	IBDLR	BDR-B	SABDR
2	50.8 - 81.9	50.8 - 54.0	50.7 - 94.4	51.0 - 61.8	50.8 - 75.2	50.9 - 89.8	82.0
3	34.0 - 74.5	34.0 - 51.8	34.2 - 87.0	34.0 - 59.7	34.0 - 69.0	34.7 - 78.7	77.1
5	20.4 - 62.7	20.4 - 56.2	22.6 - 79.1	20.4 - 57.8	20.4 - 64.4	21.3 - 62.5	80.3
8	12.8 - 57.8	12.8 - 54.4	19.4 - 70.8	12.8 - 59.1	12.8 - 60.3	13.1 - 56.3	45.6
10	10.2 - 55.8	10.2 - 55.0	18.8 - 66.4	10.2 - 56.6	10.2 - 54.4	10.2 - 61.2	17.4

Table 3.7: Handwritten-digit clustering performances of different BDR methods on USPS data set.

3.4.2.6 SUMMARY

Based on the derived theoretical information in Section 3.2, we have proposed SABDR [TMO23] which controls the level of sparsity by robustly estimating the regularization parameter/s. To use the available BD structure in the objective function, we proposed an efficient density-based clustering method Con-DBSCAN. SABDR is benchmarked against popular affinity matrix construction methods and it reached similar or higher performance compared to its optimally tuned competitors.

A theory is more impressive the greater the simplicity of its premises, the more different are the kinds of things it relates, and the more extended its range of applicability.

—Albert Einstein.

4

Outliers in Graph Clustering and Robust Solutions

In the previous chapter, a sparse graph model and its applicability to sparsity-aware clustering has been discussed. Beyond the challenges that are involved in determining the appropriate level of sparsity, real-world data often includes outliers and heavy tailed noise. Therefore, this chapter starts with determining fundamental outlier types for graph-based clustering in Section 4.1. To design robust graph-based clustering algorithms, Section 4.2 analyzes the effect of fundamental outlier types on sparse graphs. Then, the natural occurrence of outliers based on the level of sparsity has been detailed in Section 4.3. Finally, proposed robust graph-based clustering methods are presented in Section 4.4.

4.1 DETERMINING FUNDAMENTAL OUTLIER TYPES FOR GRAPH-BASED CLUSTERING

From Section 3.2.2, it follows that the non-zero eigenvalues of the sparse Laplacian matrix contain the block size information. However, in practice, such a sparse Laplacian matrix is not readily available. Especially for outlier-corrupted Laplacian matrices, the blocks might be obscured, which

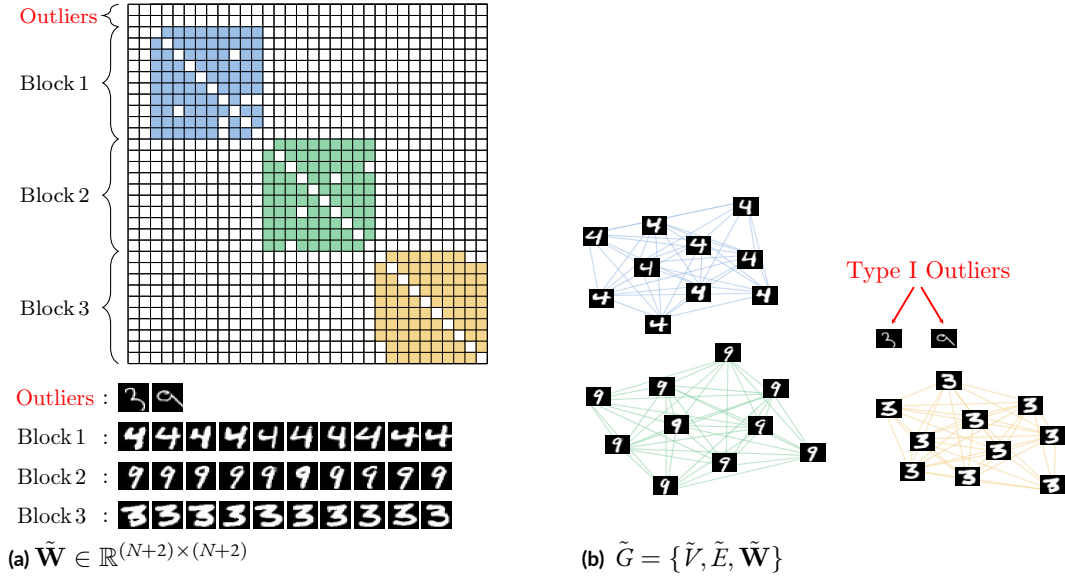


Figure 4.1: Illustration of Type I outliers. The colored cells in the corrupted BD affinity matrix $\tilde{\mathbf{W}}$ represent non-zero edge weights in graph \tilde{G} .

results, e.g., in a performance degradation of an eigenvalue-based block size estimate. To quantify this more precisely, and subsequently derive robust graph-based clustering methods, we define fundamental outlier types in the following.

4.1.1 TYPE I OUTLIERS

Motivated by [EV13], we begin by defining the first fundamental type of outliers as follows.

Definition 4.1.1. (*Type I Outliers, [TMZ23, TMZ22]*) *The feature vectors corresponding to the vertices that do not share edges with any of the samples are called Type I outliers.*

Based on this first definition, the similarity coefficient vectors that are associated to Type I outliers, ideally, are zero vectors. More practically speaking, and motivated by real data examples, the data-points whose similarity coefficients have negligibly small values may also be called Type I outliers.

Definition 4.1.1 is illustrated in Figure 4.1 using the well-known handwritten digit samples from the MNIST data base [HS98]. In the exemplary corrupted graph \tilde{G} , the unconnected vertices are the Type I outliers while the vertices of digits 9, 4 and 3 are connected with within-cluster edges that are highlighted in green, blue and yellow lines, respectively.

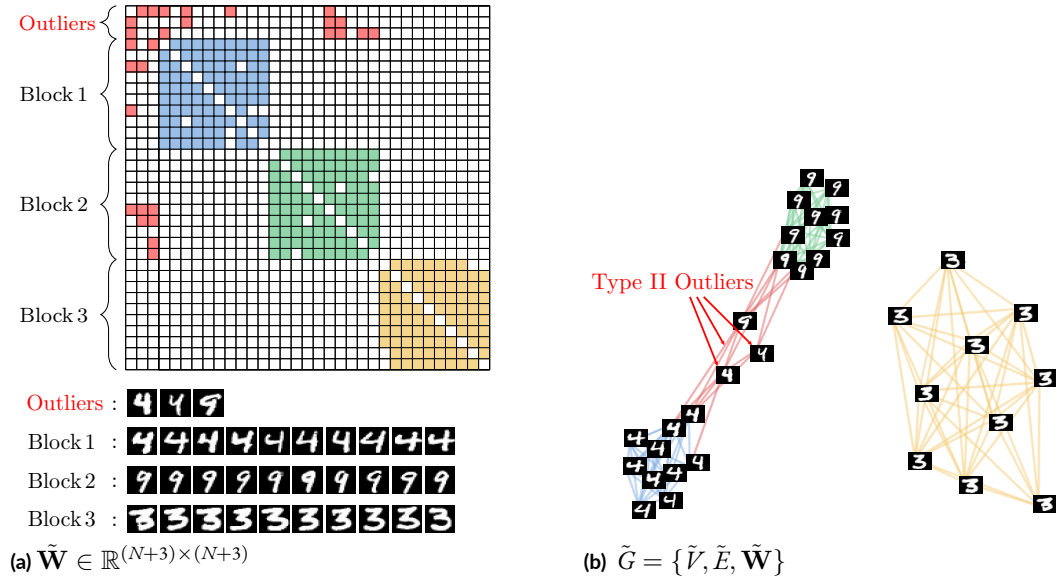


Figure 4.2: Illustration of Type II outliers. The red colored cells in $\tilde{\mathbf{W}}$ correspond to edges of Type II outliers.

4.1.2 TYPE II OUTLIERS

The second fundamental outlier type is called Type II outliers that is quantified in the following.

Definition 4.1.2. (Type II Outliers, [TMZ23, TMZ22]) *The feature vectors corresponding to the vertices that share edges with more than one group of feature vectors are called Type II outliers.*

Definition 4.1.2 is illustrated in Figure 4.2, which shows that the connectedness of Type II outliers to multiple groups of feature vectors obscures the true clusters and poses a challenge to graph-based clustering methods.

4.1.3 GROUP SIMILARITY

Group similarity is an extreme case of Type II outliers which is detailed the following definition.

Definition 4.1.3. (Group Similarity) *If an entire group of vertices shares edges with another group of vertices this is called group similarity.*

4.2 OUTLIER EFFECTS ON SPARSE GRAPHS

To incorporate robustness into graph-based clustering, our next step is to understand the effects of the above introduced fundamental outlier types on sparse graphs. Therefore, the following sections analyze their effects from four different perspectives.

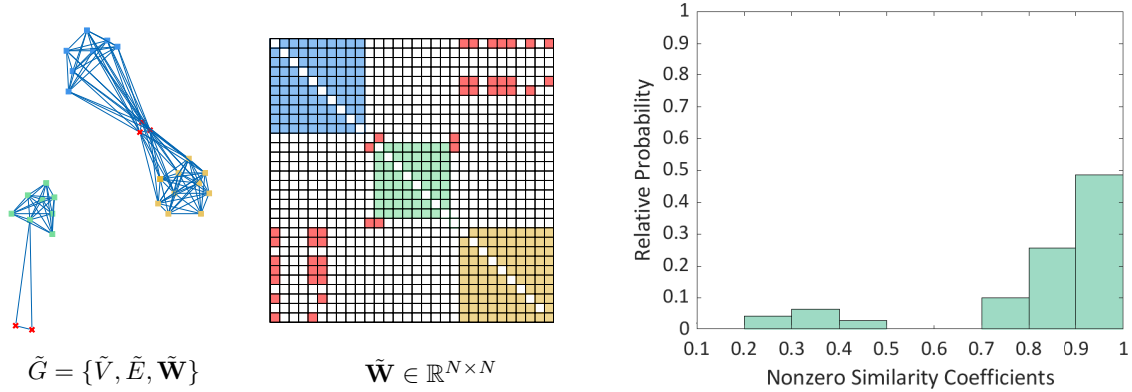


Figure 4.3: Illustration of outliers' effect on the affinity matrix.

4.2.1 OUTLIER EFFECTS ON AFFINITY MATRIX

Since the components of the affinity matrix are directly linked to the edge weights in the associated graph model, it is important to analyze the effect of outliers on the affinity matrix.

To provide a visual understanding of outliers' effects, an exemplary Type I and Type II outlier corrupted graph model, the associated affinity matrix and the empirical distribution of similarity coefficients are shown in Figure 4.3. Herein, Type I outliers are shown for a practical scenario such that they have only a few edges and can be considered as an unconnected component compared to highly connected true samples and Type II outliers. The figure shows that the undesired edges associated with Type II outliers represent undesired similarity coefficients between different blocks for a sparse BD affinity matrix. If we analyze the empirical distribution of these similarity coefficients, it follows that the Type II outliers result in undesired similarity coefficient subspaces. An important property of these undesired similarity coefficients is that, if similarity coefficients within the blocks are valued larger than undesired similarity coefficients between different blocks, the graph structure can be recovered by removing these undesired coefficients. In Section 4.4.1.1, we developed an unsupervised robust clustering algorithm to shrink these undesired similarity coefficients to zero.

4.2.2 OUTLIER EFFECTS ON OVERALL EDGE WEIGHTS

This section introduces and discusses outliers' effects on the overall edge weights and how the overall edge weights can be used as an outlyingness measure to suppress both Type I and Type II outliers. According to Definition 4.1.1 a Type I outlier is a relatively unconnected (or ideally, even totally unconnected) vertex that has noticeably small-valued (or ideally zero-valued) overall

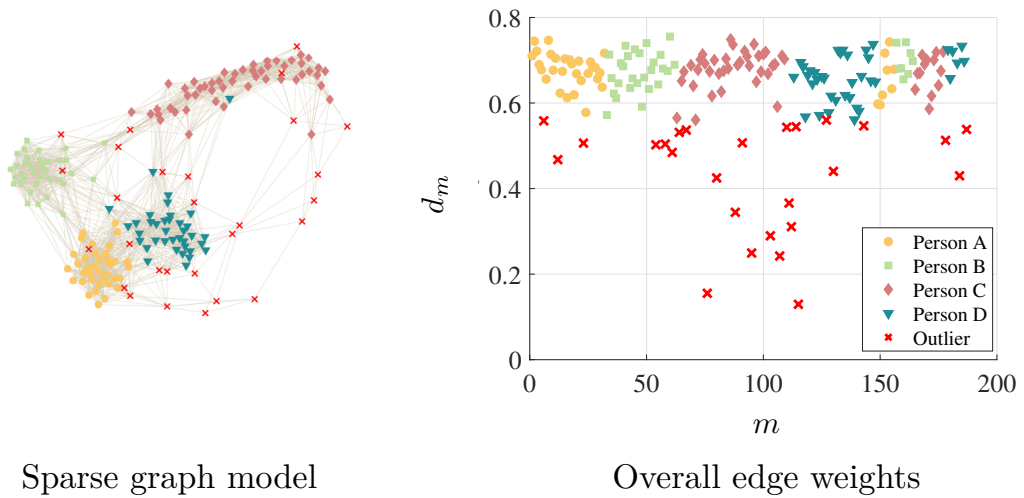


Figure 4.4: Exemplary outlyingness measure of Person Id. [TSM18] data set based on the overall edge weights.

edge weight in comparison to the non-outlying samples of the clusters. This means that, for a graph model including a considerable number of edges within the clusters, clearly, Type I outliers' overall edge weights will deviate from the typical ones. In contrast to this comparably simple characterization of Type I outlyingness, Type II outlyingness determination based on overall edge weight depends on cluster sizes and the parameters of available graph, e.g. the edge weights and the affinity matrix. For example, for a graph model of not extremely imbalanced cluster sizes, the overall edge weight of a Type II outlier is smaller-valued than that of the typical data points when the Type II outlier is connected to multiple groups of vertices with small-valued edge weights. Thus, while both outlier types behave differently, it is important to note for a connected graph model of comparable cluster sizes both types of outliers have a common characteristic: *their overall edge weights deviate from that of the typical vertices.*

To provide a visual understanding, exemplary outlier assignments are shown for sparse graph models of the Person Id. [TSM18] and Gait [SAZ19] real-world data sets in Figure 4.4 and Figure 4.5, respectively. In both graphs, the red crosses depict the outliers that include the 15% of vertices whose overall edge weights deviate maximally from the median of overall edge weights (the median represents the typical overall edge weights). As can be seen, the outlier assignment based on overall edge weight captures both vertices between different clusters (Type II outliers) and the vertices that are far from every cluster (Type I outliers). In Section 4.4.2.1, we present a robust Fiedler vector estimation algorithm suppressing outliers' negative impact based on the overall edge weight.

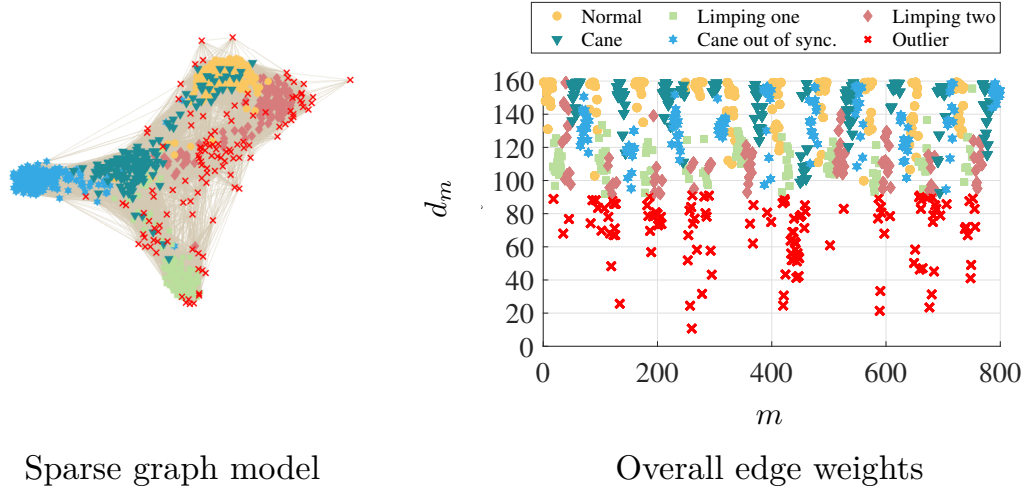


Figure 4.5: Exemplary outlierness measure of Gait [SAZ19] data set based on the overall edge weights.

4.2.3 OUTLIER EFFECTS ON EIGENVALUES

4.2.3.1 TYPE I OUTLIERS' EFFECT ON EIGENVALUES

To understand Type I outliers' effects on the eigenvalues, it is important to remember the relationship between the number of connected components of a K BD affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ and the spectrum of the associated graph Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$. In [Lux07], it has been shown that the multiplicity of the zero-valued eigenvalues of \mathbf{L} equals the number of connected components K . Clearly, considering Type I outliers as isolated blocks of size one, the addition of N_1 number of Type I outliers leads to N_1 additional zero-valued eigenvalues.

The effect of Type I outliers on eigenvalues is visualized in Figure 4.6. As can be seen, a Type I outlier results in an additional zero-valued eigenvalue which is highlighted in dark red color.

4.2.3.2 TYPE II OUTLIERS' EFFECT ON EIGENVALUES

In contrast to Type I outlier effects that were studied in Section 4.2.3.1, understanding the effect of Type II outliers on eigenvalues requires further analysis. Therefore, this section analyzes Type II outliers' effect on the eigenvalues of the Laplacian matrix for the generalized eigen-decomposition in Eq. (2.6).¹

¹For analysis based on the standard eigen-decomposition in Eq. (2.5), see Appendix A.2.2.

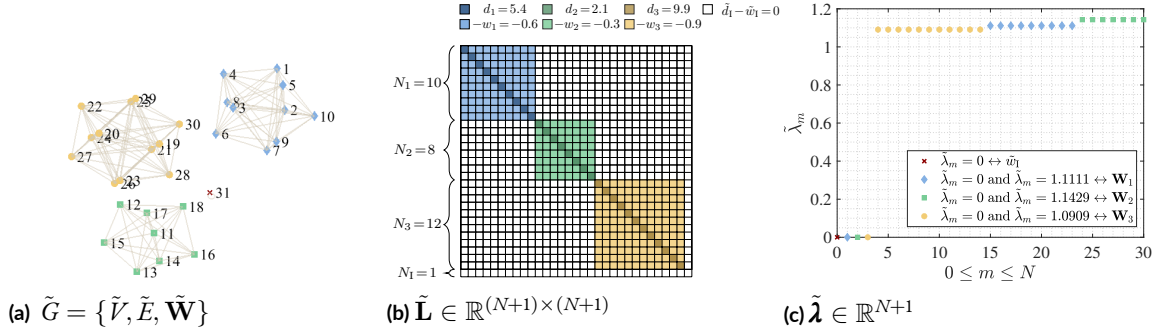


Figure 4.6: Exemplary plot of Type I outliers' effect on eigenvalues ($\mathbf{n} = [10, 8, 12, 1]^T \in \mathbb{R}^{K+1}$, $N+1 = 31$, $K = 3$).

Theorem 4. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ define a symmetric affinity matrix, that is equal to \mathbf{W} , except for an additional Type II outlier that shares similarity coefficients with K blocks where $\tilde{w}_{\Pi,K} > 0$ denotes the similarity coefficient between the outlier \mathbf{o}_{Π} and the K th block. Then, for the associated corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ with eigenvalues $\tilde{\boldsymbol{\lambda}} \in \mathbb{R}^{N+1}$, it holds that

$$\left\{ \begin{array}{l} N_1 - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_1 w_1 + \tilde{w}_{\Pi,1}}{\tilde{d}_1}, \\ N_2 - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_2 w_2 + \tilde{w}_{\Pi,2}}{\tilde{d}_2}, \\ \vdots \\ N_K - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_K w_K + \tilde{w}_{\Pi,K}}{\tilde{d}_K}, \\ \text{the smallest element of } \tilde{\boldsymbol{\lambda}} \text{ is equal to zero,} \end{array} \right.$$

and the remaining K eigenvalues are the roots of

$$\prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda} \tilde{d}_j) \left(- \sum_{j=1}^K \frac{N_j \tilde{w}_{\Pi,j} \tilde{d}_j}{\tilde{w}_{\Pi,j} - \tilde{\lambda} \tilde{d}_j} - \tilde{d}_{\Pi} \right) = 0,$$

where $\tilde{d}_{\Pi} = \sum_{j=1}^K N_j \tilde{w}_{\Pi,j}$ and $\tilde{d}_j = (N_j - 1)w_j + \tilde{w}_{\Pi,j}$.

Proof. See Appendix A.2.1.1. □

Theorem 4 is illustrated in Figure 4.7. The figure confirms the results of Theorem 4 that Type II outliers result in an increase in the eigenvalues.

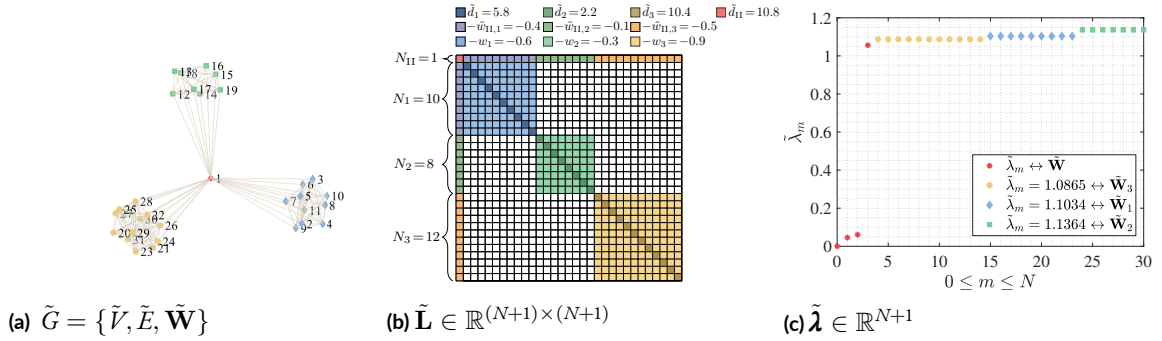


Figure 4.7: Exemplary illustration of Theorem 4 ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}, N + 1 = 31, K = 3$).

4.2.3.3 GROUP SIMILARITY EFFECT ON EIGENVALUES

The Laplacian matrix of Definition 4.1.3 can be considered as a single connected component which means that the number of zero-valued eigenvalues equals to one [Lux07]. In contrast to this simple interpretation, the remaining eigenvalues can be formulated as a function of intra-blocks and inter-blocks similarity coefficients where inter-blocks similarity coefficients are generally smaller-valued than that of intra-blocks in real-world scenarios. To provide a mathematical understanding of this, the following theorem quantifies the effect of group similarity on the target eigenvalues.

Theorem 5. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ define an affinity matrix, that is equal to \mathbf{W} , except that block i has similarity with the remaining $K - 1$ blocks with $\tilde{w}_{i,j} = \tilde{w}_{j,i} > 0$ denoting the value around which the similarity coefficients between blocks i and j are concentrated for $j = 1, \dots, K$ and $i \neq j$. Then, the eigenvalues $\tilde{\boldsymbol{\lambda}} \in \mathbb{R}^N$ of $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ are as follows:

$$\left\{ \begin{array}{l} N_i - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_i w_i + \sum_{\substack{j=1, \\ j \neq i}}^K N_j \tilde{w}_{i,j}}{\tilde{d}_i}, \\ N_j - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_j w_j + N_i \tilde{w}_{i,j}}{\tilde{d}_j}, \\ \vdots \\ N_K - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } \frac{N_K w_K + N_i \tilde{w}_{i,K}}{\tilde{d}_K}, \\ \text{the smallest element of } \tilde{\boldsymbol{\lambda}} \text{ is equal to zero,} \end{array} \right.$$

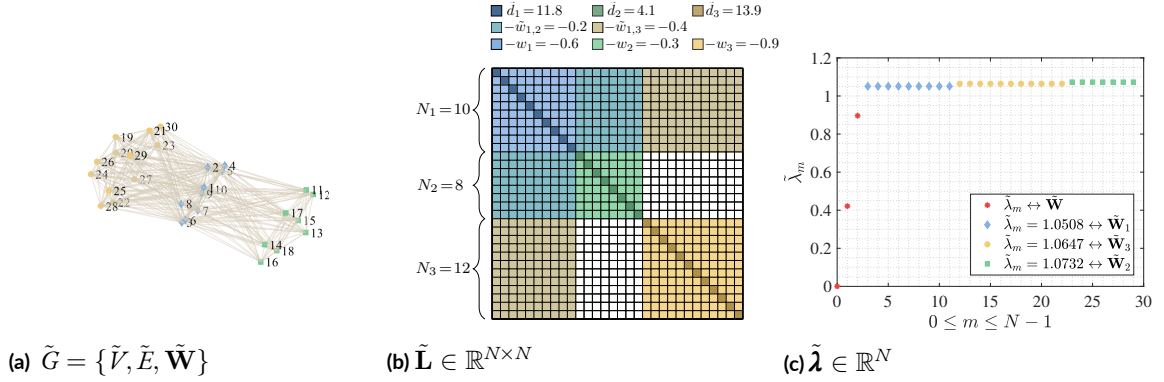


Figure 4.8: Exemplary illustration of Theorem 5 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$, $i = 1$).

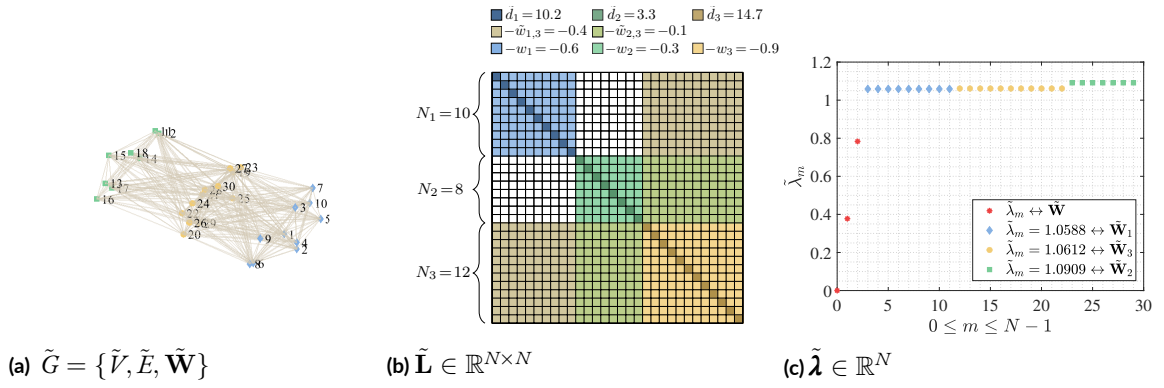


Figure 4.9: Exemplary illustration of Theorem 5 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$, $i = K$).

and the remaining $K - 1$ eigenvalues in $\tilde{\boldsymbol{\lambda}}$ are the roots of

$$\prod_{\substack{j=1 \\ j \neq i}}^K (N_i \tilde{w}_{i,j} - \tilde{\lambda} \tilde{d}_j) \left(- \sum_{\substack{j=1 \\ j \neq i}}^K \frac{\tilde{d}_j N_j \tilde{w}_{i,j}}{N_i \tilde{w}_{i,j} - \tilde{\lambda} \tilde{d}_j} - \tilde{d}_i \right) = 0,$$

where $\tilde{d}_j = (N_j - 1)w_j + N_i \tilde{w}_{i,j}$, $\tilde{d}_i = (N_i - 1)w_i + \sum_{\substack{j=1 \\ j \neq i}}^K N_j \tilde{w}_{i,j}$.

Proof. See Appendix A.2.1.2. □

The illustration of Theorem 5 is given in Figs 4.8 and 4.9, respectively, for $i = 1$ and $i = K$. As can be seen, the number of zero-valued eigenvalues equals to one while the remaining $N - 1$ eigenvalues are valued as it has been explained in Theorem 5.

4.2.4 OUTLIER EFFECTS ON EIGENVECTORS

Motivated by the advantages of spectral methods in graph clustering (see, Section 2.5), this sections analyzes the effect of outliers and group similarity for SC [NJW01] which is one of the most popular graph clustering method building upon the eigenvectors associated with the K smallest eigenvalues of the Laplacian matrix.

4.2.4.1 TYPE I OUTLIERS' EFFECT ON EIGENVECTORS

According to [Lux07], the eigenspace of zero-valued eigenvalues is spanned by indicator vectors of connected components in \mathbf{L} . Since Type I outliers can be considered as isolated blocks of size one, it follows that the eigenspace of N_I additional zero-valued eigenvalues associated with Type I outliers is spanned by the indicator vectors of the N_I outliers.

4.2.4.2 TYPE II OUTLIERS' EFFECT ON EIGENVECTORS

In contrast to simple location understanding of Type I outliers, Type II outliers' possible location necessitates further analysis that is conducted in the sequel.

Proposition 4.2.1. *Let $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a K block zero-diagonal symmetric affinity matrix as in Definition 3.1.1. Further, let $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ define a symmetric affinity matrix, which is equal to \mathbf{W} , except for an additional Type II outlier \mathfrak{o}_{II} that is connected to the vertices associated with i th and j th blocks. Finally, let $\mathbf{Y} = [\mathbf{y}_0, \dots, \mathbf{y}_{K-1}] \in \mathbb{R}^{N \times K}$ and $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_0, \dots, \tilde{\mathbf{y}}_{K-1}] \in \mathbb{R}^{(N+1) \times K}$ be the matrix of orthonormal eigenvectors associated with the K smallest eigenvalues of \mathbf{L} and $\tilde{\mathbf{L}}$, respectively. Then, for \mathbf{e}_m and $\tilde{\mathbf{e}}_m$, respectively, the m th row vector of \mathbf{Y} and $\tilde{\mathbf{Y}}$, denoting the embedding vectors that represents the M -dimensional m th feature vector in the reduced K -dimensional space, the embedding vector associated to Type II outlier $\tilde{\mathbf{e}}_{\text{II}}$ is centered between mappings of blocks i and j if the distance between every pair of embedding vectors correspond to true samples are preserved, i.e. $\|\mathbf{e}_m - \mathbf{e}_n\|_2 = \|\tilde{\mathbf{e}}_m - \tilde{\mathbf{e}}_n\|_2$ for $m \neq \text{II}$ and $n \neq \text{II}$.*

Proof. See Appendix A.2.1.3. □

Proposition 4.2.1 is illustrated for the eigenvectors of the corrupted Laplacian matrix $\tilde{\mathbf{L}}$ in Figure 4.10. As can be seen, a Type II outlier is located between the embeddings of blocks i and j . In the sequel, a further analysis is conducted for Type II outlier similarity to K blocks.

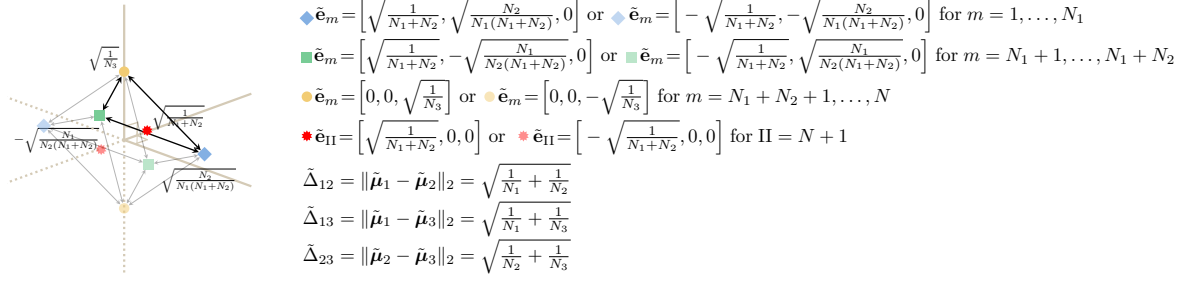


Figure 4.10: Spectral embedding according to the eigenvectors of the corrupted Laplacian matrix $\tilde{\mathbf{L}}$ when $K = 3$, $i = 1$ and $j = 2$.

Proposition 4.2.2. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ define a symmetric affinity matrix, which is equal to $\mathbf{W} \in \mathbb{R}^{N \times N}$ in Definition 3.1.1, except for an additional Type II outlier \circ_{Π} correlated with K blocks. Further, let $\mathbf{Y} = [\mathbf{y}_0, \dots, \mathbf{y}_{K-1}] \in \mathbb{R}^{N \times K}$ and $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_0, \dots, \tilde{\mathbf{y}}_{K-1}] \in \mathbb{R}^{(N+1) \times K}$ be the matrix of orthonormal eigenvectors associated with the K smallest eigenvalues of \mathbf{L} and $\tilde{\mathbf{L}}$, respectively. Then, for \mathbf{e}_m and $\tilde{\mathbf{e}}_m$, respectively, the m th row vector of \mathbf{Y} and $\tilde{\mathbf{Y}}$, denoting the embedding vectors that represents the M -dimensional m th feature vector in the reduced K -dimensional space, the embedding vector associated to Type II outlier $\tilde{\mathbf{e}}_{\Pi}$ converges to the origin, i.e. $\|\tilde{\mathbf{e}}_{\Pi}\|_2 \rightarrow 0$, if and only if the embedding vectors associated to true samples in Theorem 2 are preserved.

Proof. See Appendix A.2.1.4. □

Propositions 4.2.1 and 4.2.2 explain an ideal case for the location of Type II outliers in the reduced K dimensional space. In real-world scenarios, the distance between the true samples may not be preserved due to similarity coefficients between Type II outliers and true samples. However, if the clusters are internally dense and sparsely connected to the rest of the graph, the embedding operation may provide approximate results to the Propositions 4.2.1 and 4.2.2. In more details, the embedding idea that minimizes the distance between similar objects while maximizing it for that of dissimilar ones results in easily separable clusters as in Theorem 2. In such cases, Type II outliers are more likely to be located as in Propositions 4.2.1 and 4.2.2.

4.2.4.3 GROUP SIMILARITY EFFECT ON EIGENVECTORS

The analysis of the effect of outliers on eigenvectors is extended for group similarity in the following proposition.

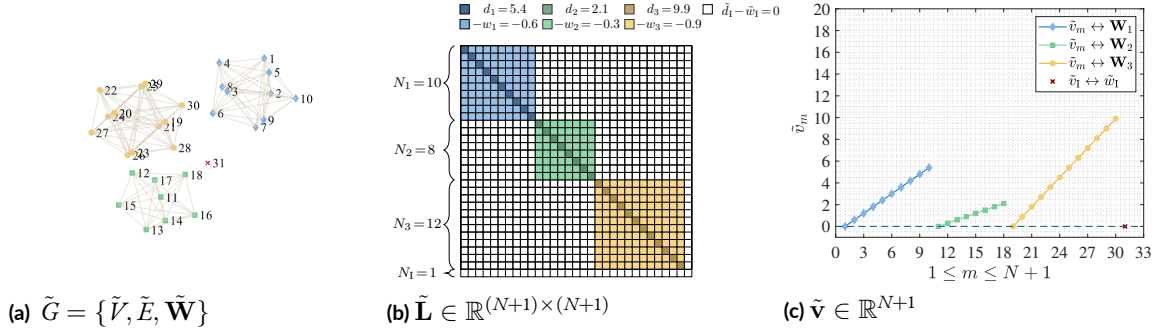


Figure 4.11: Exemplary plot of Type I outliers' effect on vector \mathbf{v} ($\mathbf{n} = [10, 8, 12, 1]^T \in \mathbb{R}^{K+1}, N+1 = 31, K = 3$).

Proposition 4.2.3. Let $\tilde{\mathbf{W}}$ define an affinity matrix, which is equal to \mathbf{W} as in Definition 3.1.1, except that we impose a constant $\tilde{w}_u > 0$ around which the similarity coefficients between blocks i and j are concentrated. Further, let $\mathbf{L} \in \mathbb{R}^{N \times N}$ and $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ denote the Laplacian matrices associated with $\mathbf{W} \in \mathbb{R}^{N \times N}$ and $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$, respectively. The eigenvectors associated with the K smallest eigenvalues of \mathbf{L} and $\tilde{\mathbf{L}}$, respectively, be the column vectors of the matrices $\mathbf{Y} \in \mathbb{R}^{N \times K}$ and $\tilde{\mathbf{Y}} \in \mathbb{R}^{N \times K}$ where K denotes the number of clusters. Finally, let $\mathbf{e}_m \in \mathbb{R}^K$ and $\tilde{\mathbf{e}}_m \in \mathbb{R}^K$, the m th row vector of \mathbf{Y} and $\tilde{\mathbf{Y}}$, respectively, denote the embedding vectors that represent the M -dimensional m th feature vector in the reduced K -dimensional space. Assuming that the column vectors of \mathbf{Y} and $\tilde{\mathbf{Y}}$ are valued in a range $\{y_{\min}, y_{\max}\}$, the squared Euclidean distance between any embedding vector pair \mathbf{e}_m and \mathbf{e}_n associated with different blocks is greater than that of $\tilde{\mathbf{e}}_m$ and $\tilde{\mathbf{e}}_n$, i.e. $\|\mathbf{e}_m - \mathbf{e}_n\|_2^2 > \|\tilde{\mathbf{e}}_m - \tilde{\mathbf{e}}_n\|_2^2$.

Proof. See Appendix A.2.1.5. □

4.2.5 OUTLIER EFFECTS ON SIMPLIFIED LAPLACIAN MATRIX ANALYSIS

4.2.5.1 TYPE I OUTLIERS' EFFECT ON SIMPLIFIED LAPLACIAN MATRIX ANALYSIS

For a Type I outlier-corrupted affinity matrix $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ that is identical to \mathbf{W} except for a single Type I outlier \mathbf{o}_I , the overall edge weight associated with \mathbf{o}_I is zero-valued, i.e. $\tilde{d}_I = 0$. Based on Theorem 3 and Eq. (3.2), it is straight-forward to show that the component in the associated corrupted vector $\tilde{\mathbf{v}} \in \mathbb{R}^{N+1}$ that is associated with the Type I outlier is zero valued, i.e., $\tilde{v}_I = 0$.

The vector $\tilde{\mathbf{v}} \in \mathbb{R}^{N+1}$ associated with a Type I outlier corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ is shown in Figure 4.11. As can be seen, the component that is associated with the Type I outlier is zero valued in $\tilde{\mathbf{v}}$, i.e., $\tilde{v}_I = 0$.

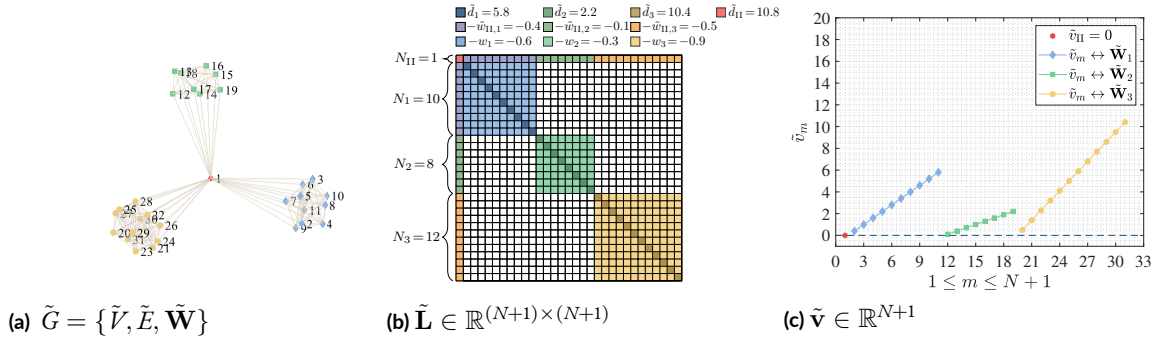


Figure 4.12: Exemplary illustration of Theorem 6 ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}$, $N+1 = 31$, $K = 3$, $m_{\text{II}} = 1$).

4.2.5.2 TYPE II OUTLIERS' EFFECT ON SIMPLIFIED LAPLACIAN MATRIX ANALYSIS

Type II outliers' effect on \mathbf{v} is as follows.

Theorem 6. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ define a Type II outlier-corrupted BD affinity matrix that is identical to $\mathbf{W} \in \mathbb{R}^{N \times N}$ except for a single Type II outlier that has non-zero similarity coefficients with all blocks. Assuming that the similarity coefficients associated with the outlier \circ_{II} and the blocks $j \in \{1, \dots, K\}$ are concentrated around $\tilde{w}_{\text{II},j}$, the components, whose indexes are valued between the outlier index and the largest index of the j th block, such that $m_{\text{II}} < m \leq u_j$, increase by $\tilde{w}_{\text{II},j}$ in the corrupted vector $\tilde{\mathbf{v}} \in \mathbb{R}^{N+1}$. Further, the component associated with the Type II outlier is given by

$$\tilde{v}_{\text{II}} = \begin{cases} 0 & \text{if } 0 < m_{\text{II}} < \ell_1 \\ (m_{\text{II}} - \ell_1)\tilde{w}_{\text{II},1} & \text{if } \ell_1 < m_{\text{II}} < \ell_2 \\ \vdots & \\ \sum_{j=1}^{K-1} N_j \tilde{w}_{\text{II},j} + (m_{\text{II}} - \ell_K)\tilde{w}_{\text{II},K} & \text{if } \ell_K < m_{\text{II}} \leq N+1 \end{cases},$$

where ℓ_j denotes the lowest index of the j th block.

Proof. See Appendix A.2.1.6. □

Theorem 6 is illustrated in Figures 4.12 and 4.13, respectively, for two different locations of Type II outlier, i.e. $m_{\text{II}} = \ell_2 - 1$ and $m_{\text{II}} = 1$. As the figures imply, Type II outliers result in deviations on the vector \mathbf{v} .

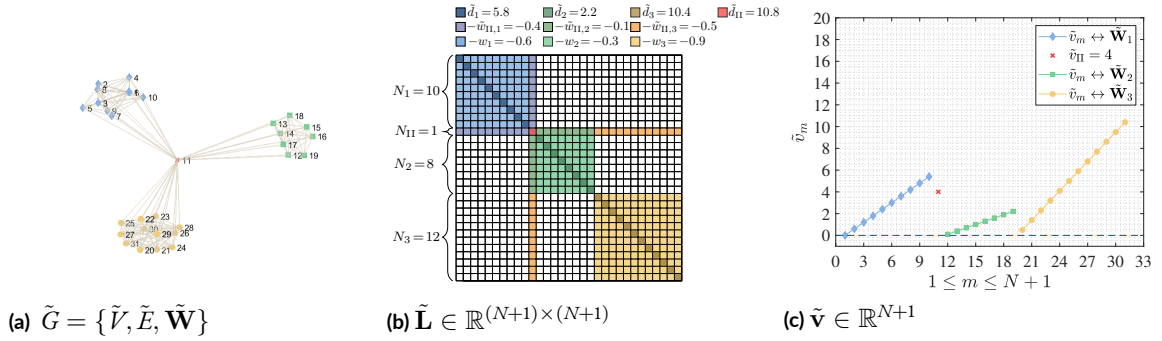


Figure 4.13: Exemplary illustration of Theorem 6 ($\mathbf{n} = [10, 1, 8, 12]^\top \in \mathbb{R}^{K+1}$, $N+1 = 31$, $K = 3$, $m_{II} = \ell_2 - 1$).

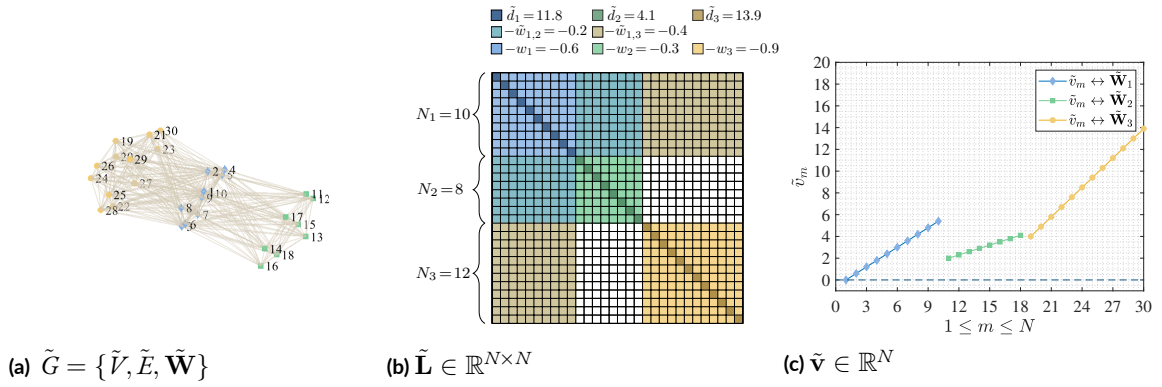


Figure 4.14: Exemplary illustration of Theorem 7 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$, $i = 1$).

4.2.5.3 GROUP SIMILARITY EFFECT ON SIMPLIFIED LAPLACIAN MATRIX ANALYSIS

The effect of group similarity on \mathbf{v} is as follows.

Theorem 7. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ define a corrupted affinity matrix that is identical to $\mathbf{W} \in \mathbb{R}^{N \times N}$ except that block i has non-zero similarity coefficients with the remaining $K - 1$ blocks with $\tilde{w}_{i,j} = \tilde{w}_{j,i} > 0$ denoting the similarity coefficients around which blocks i and j are concentrated. These similarities result in an increase by $N_i \tilde{w}_{i,j}$ in the components associated with the blocks $j = i + 1, \dots, K$ of $\tilde{\mathbf{v}} \in \mathbb{R}^N$ while the components of $j < i$ remain the same. Further, the components associated with block i remain the same for $i = 1$ and increase by $\sum_{j=1}^{i-1} N_j \tilde{w}_{i,j}$ for $2 \leq i \leq K$.

Proof. See Appendix A.2.1.7. □

The illustration of Theorem 7 is given in Figures 4.14 and 4.15, respectively, for $i = 1$ and $i = K$. Consistent with Theorem 7, group similarity leads to an increase in the target vector \mathbf{v} .

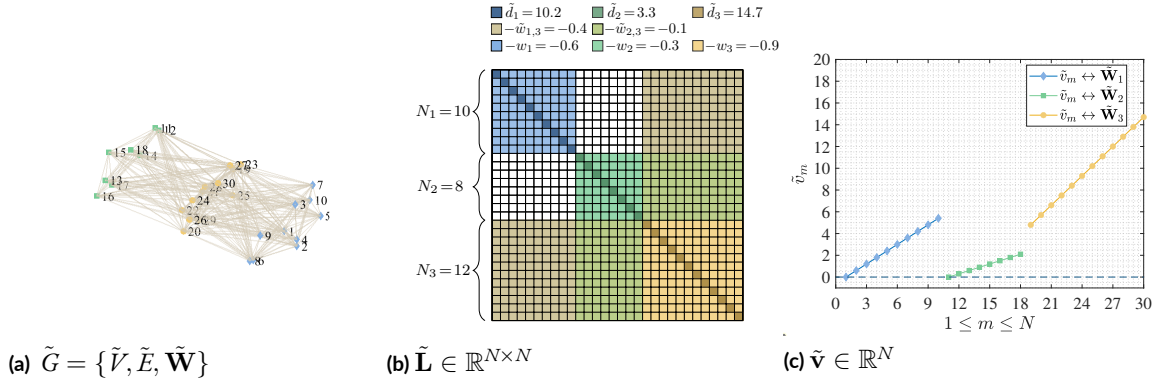


Figure 4.15: Exemplary illustration of Theorem 7 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$, $i = K$).

In the following, the worst case of group similarity, i.e., similarity of all blocks is analyzed. Note that, in this case, eigenvalues can not even be formulated as a function of similarity coefficients due to the impossibility of simplifying determinants of full matrices via Gaussian elimination. However, recovering the structure of \mathbf{W} based on \mathbf{v} is possible based on the following result.

Corollary 7.1. *Let $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ define a corrupted affinity matrix that is identical to $\mathbf{W} \in \mathbb{R}^{N \times N}$ except that each block $i = 1, \dots, K$ has non-zero similarity coefficients with the remaining $K-1$ blocks with $\tilde{w}_{i,j} = \tilde{w}_{j,i} > 0$ denoting the similarity coefficients around which blocks i and j are concentrated for $j = 1, \dots, K$ and $i \neq j$. This leads to a piece-wise linear function given by*

$$\tilde{v}_m = \begin{cases} (m - \ell_1)w_1 & \text{if } \ell_1 \leq m \leq u_1 \\ (u_1 - \ell_1 + 1)\tilde{w}_{1,2} + (m - \ell_2)w_2 & \text{if } \ell_2 \leq m \leq u_2 \\ \vdots & \\ \sum_{i=1}^{K-1} (u_i - \ell_i + 1)\tilde{w}_{i,K} + (m - \ell_K)w_K & \text{if } \ell_K \leq m \leq u_K \end{cases}$$

where $\ell_1 = 1$, $u_1 = N_1$, $\ell_i = \sum_{k=1}^{i-1} N_k + 1$ and $u_i = \sum_{k=1}^i N_k$ for $i = 2, \dots, K$.

Proof. See Appendix A.2.1.8. □

The worst case of group similarity is illustrated in Figure 4.16 for a $K = 3$ BD Laplacian matrix in which every similarity coefficient is non-zero valued. Consistent with the theoretical results of Corollary 7.1, undesired similarity coefficient between different blocks result in shifts starting from the second linear segment.

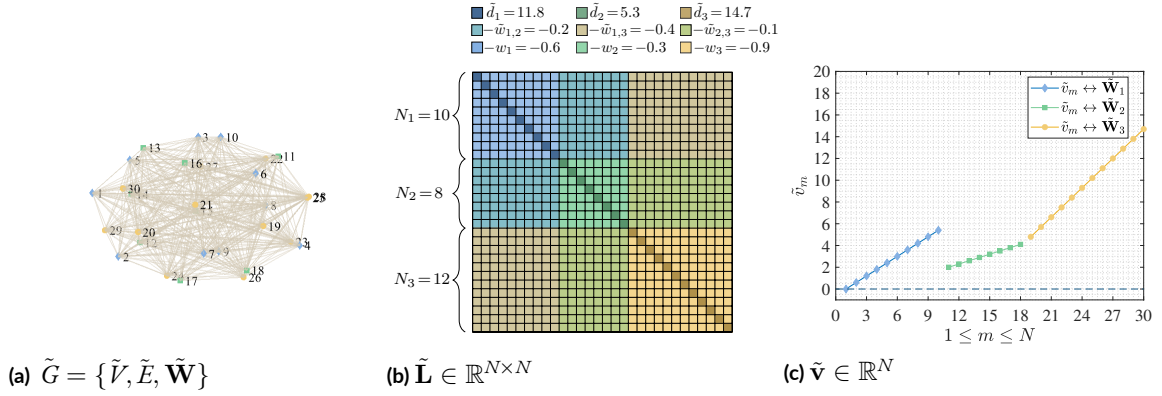


Figure 4.16: Exemplary illustration of Corollary 7.1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$).

4.3 SPARSITY AND OUTLIER OCCURRENCE

With the results of Sections 4.1 and 4.2 in place, we are ready to understand the relationship between the level of sparsity and the previously defined outlier types to highlight the importance of jointly addressing robustness and sparsity. In a generic example, Figure 4.17 shows that a dense graph (top) contains high amounts of group similarity while increasing sparsity reduces the number of Type II outliers (middle). Finally, further increasing sparsity generates Type I outliers until at some point the underlying true cluster structure is completely lost. This means that an inaccurate determination of the sparsity level leads to the above discussed outlier effects for non-robust graph clustering approaches. In the following sections, we therefore introduce the variety of proposed robust graph clustering approaches which address robustness from three different perspectives.

4.4 ROBUST GRAPH-BASED CLUSTERING METHODS

After analysing the effect of outliers and group similarity, our goal is to suppress their effects on the graph structure. As the affinity matrix directly refers to the graph structure, we first present the proposed robust and sparsity-aware affinity matrix construction methods. Motivated by the negative impact of outliers and group similarity on the eigenvectors, the proposed eigenvectors estimation methods are explained subsequently. Finally, the proposed outlier detection algorithm building upon vertex degree is introduced.

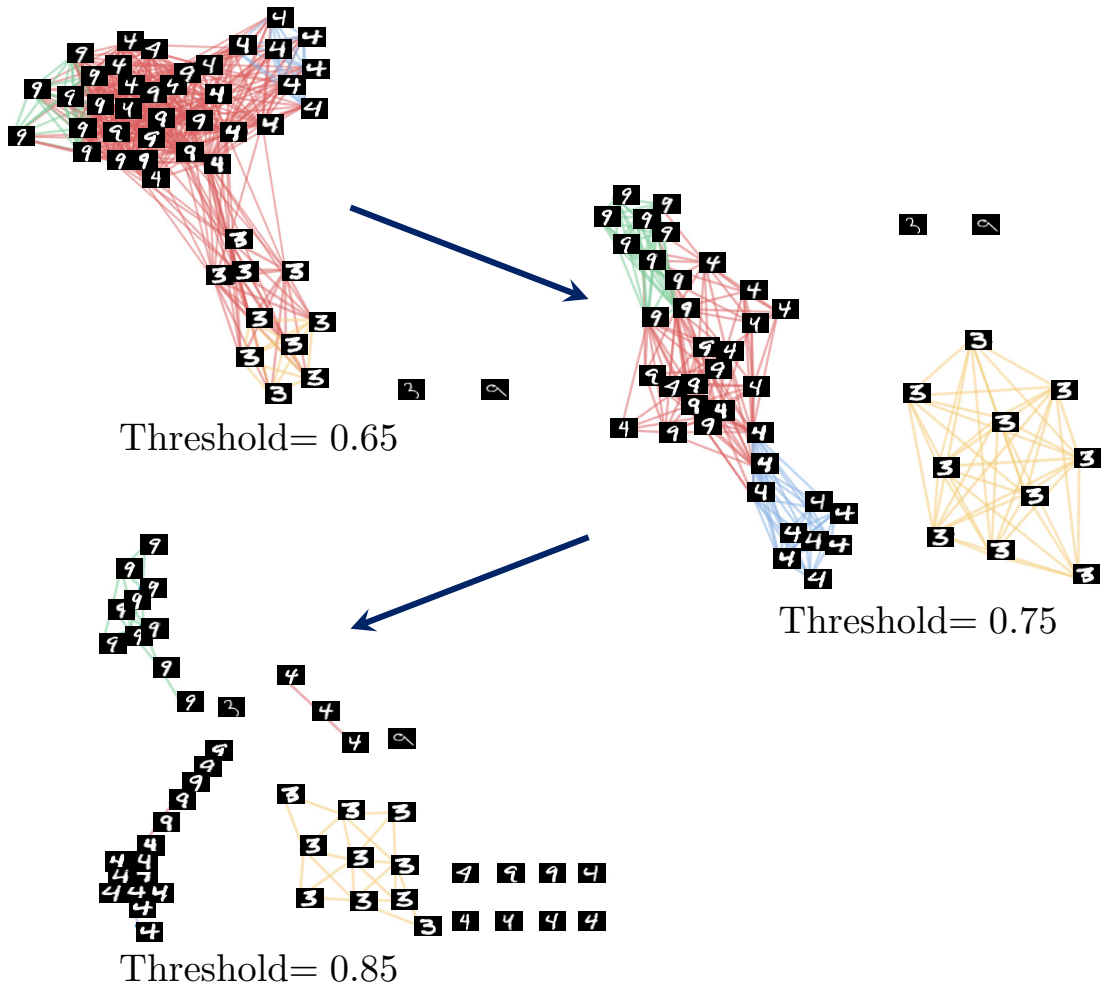


Figure 4.17: Example graphs for increasing sparsity ($W = X^T X$).

4.4.1 ROBUST AND SPARSITY-AWARE AFFINITY MATRIX CONSTRUCTION METHODS

This section presents three different proposed robust sparsity-aware affinity matrix methods which are, respectively, built upon the outliers and group similarity effect analysis on affinity matrices, vector \mathbf{v} and eigenvectors that have been detailed in Section 4.2.

4.4.1.1 SPARSITY-AWARE ROBUST COMMUNITY DETECTION

4.4.1.1.1 INTRODUCTION

Inferring a graph model from empirical observations is a fundamental data science task, and a large number of graph construction algorithms have been proposed, e.g. [LSW16, EV13, CYY09, WYG08]. In network modelling, graphs are used to represent the interactions between components of a system [FSC04], and in cluster analysis, the similarity between features can be expressed by a weighted graph [TMZ20, WYG08]. Graph models play a crucial role, for example, in subspace learning [EV13, LLY12], manifold learning [BN01, RS00] and semi-supervised learning [Zhu08]. In particular, the inference of a graph model forms the basis of graph partitioning [KN11, SM00] and community detection algorithms, which has been a very active area of research in recent years [BYS17, BEL14, SCB14, MH11, PRE11].

Community detection refers to finding densely connected groups of vertices, which helps to deduce the underlying structure and relationships that are inherent to the data. A rather important and typical situation is when the data is corrupted by heavy-tailed noise and outliers [SM21, TMZ21, LP19, MMY19, ZKO18, OT14]. This may lead to a performance degradation for popular graph-based community detection methods, using modularity optimization e.g. [MH11, BGL08, New06, New04], which is the most widely used objective function for partitioning [CNM04]. Further approaches include flow-based algorithms [BEL14], modularity total variation-based approaches [PRE11] and methods that are built upon local densities of the communities [BYS17]. A reason for the performance loss is that these methods apply optimization directly on a graph that is possibly corrupted with undesired edges that may be caused by outliers and noise. Therefore, in real-world settings with large and densely connected graphs, including graphs with a considerable amount of outliers, classical community detection methods may not be capable of recovering a graph that well-represents the underlying structure of the clean data because they give too much influence to atypical vertices. For example, methods that use pair-wise Euclidean distances, such as K -nearest neighbor and ε -ball, are particularly sensitive to noise and outliers [CYY09].

Graph-construction algorithms that use sparse representation may provide a performance gain compared to Euclidean distance-based methods [LSW16, THW16, CYY09, WYG08] and sparsity-

based characterization of locality relations can be valuable for cluster analysis [TMZ20, CLY13, BN01]. However, the performance of these clustering methods is sensitive to the level of sparsity which is essential in graphs. Sparsity in graphs has been extensively researched, e.g. in terms of the geometry of graphs and there are several approaches [ARV08, LR99] for sparsity approximation. Nevertheless, sparse methods are also affected by outliers, and determining the suitable level of sparsity becomes especially challenging. Furthermore, finding such a sparse embedding is often NP hard [ARV09].

A popular approach to promote sparsity is the LASSO regularization, which brings a relaxation for increasing sparsity on a graph without necessitating dimension reduction [FHT08]. However, the performance of the graphical LASSO critically depends on the selection of the penalty parameter that controls the sparsity of the graph. It is well-known that the selection of the penalty parameter is a challenging problem in both semi-supervised [LLO20], and unsupervised settings [MB10, HTF09] and often supervised approaches [OP09, FHT08] or neighborhood selection [MB06], are used. An interesting approach to sparsity control using the penalty parameter for graphical LASSO was made in [TWS15] by utilizing knowledge of the number of connected components of the graph. The approach controls the sparsity based on the a priori knowledge of the number of clusters, which may be difficult to estimate in the presence of outliers. To the best of our knowledge, robust sparsity control for graphical LASSO has not been applied to community detection.

To address the above challenges, this section describes a new method for Sparsity-aware Robust Community Detection (SPARCODE). The method that we presented in [TMZ21] begins with a densely connected graph and produces a preliminary sparsity-improved graph, obtained via an ℓ_1 -penalized precision matrix estimation. We also proposed a method to optimize the penalty to provide a mapping of the feature vectors from different communities in such a way that they are embedded as far as possible on the real line. Then, undesired and negligible edges are removed from the sparsity improved graph model and the graph construction is performed in a robust manner by detecting the outliers based on connectivity of vertices in the improved sparse graph model. Finally, fast spectral partitioning is performed on the outlier-free vertices of the robust sparse graph model. The number of communities is estimated using modularity optimization on partitions.

4.4.1.1.2 SIMILARITY MEASURES FOR GRAPHS

The well-known similarity measures has been detailed in Section 2.2. However, in real-world problems with large and noisy data sets, directly inferring clusters from Eq. (2.1) or (2.2) may be

inefficient, or even infeasible. For such settings, sparse graphs provide a more suitable approach to unveil the underlying data structure. It has been shown that ℓ_1 graphs, in which the edge weights of each vertex are constructed from the remaining vertices and the noise using the ℓ_1 norm (see Section 2.6.1), can provide a linear sparse representation of the data with respect to an overcomplete dictionary of basis elements [CYY09]. If the solution is sufficiently sparse [Don06], the problem can be generalized over all atoms in \mathbf{X} , using the data matrix itself as dictionary [EV13], i.e.,

$$\hat{\mathbf{A}} = \arg \min \|\mathbf{A}\|_1 \text{ s.t. } \mathbf{X} = \mathbf{XA}, \text{ diag}(\mathbf{A}) = 0, \quad (4.1)$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{N \times N}$ denotes the estimated coefficient matrix and $\text{diag}(\mathbf{A}) \in \mathbb{R}^N$ is the vector of diagonal elements of coefficient matrix \mathbf{A} .

4.4.1.1.3 PROBLEM STATEMENT

Given a data set $\mathbf{X} \in \mathbb{R}^{M \times N}$, the aim of this work is to find a label vector $\mathbf{c}_K \in \mathbb{R}^N$ that partitions \mathbf{X} into K independent and mutually exclusive communities. The true community number K is assumed to be unknown. Further, we assume that $\mathbf{X} \in \mathbb{R}^{M \times N}$ may be subject to heavy-tailed noise and outliers which obscure the data structure. Computational efficiency is also of practical interest. *Summarizing, the overall aim is to develop a fast and robust clustering algorithm based on computing a sparse graph model.*

4.4.1.1.4 SPARSITY-AWARE ROBUST COMMUNITY DETECTION (SPARCODE)

The main ideas of the proposed method are briefly summarized to provide a general understanding before going into the details. A high-level flow diagram to illustrate the key steps is provided in Figure 4.18. The community detection problem is addressed as spectral partitioning of sparse graphs. The proposed algorithm, which we call Sparsity-aware Robust Community Detection (SPARCODE), starts with a densely connected weighted graph. Assuming that edges within the communities are more densely connected than the remaining edges, the first step is to increase the sparsity of a given graph by pushing towards zero the similarity coefficients that belong to undesired and negligible edges. The sparsity improved graph model is obtained via an ℓ_1 -penalized precision matrix estimation. The penalty parameter is optimized to provide a mapping of the feature vectors from different communities in such a way that they are embedded as far as possible on the real line. In particular, we split the Fiedler vector and optimize according to what we call the polarization score. Then, undesired and negligible edges are removed from the sparsity improved graph model by applying a threshold, and the graph construction is performed in a robust

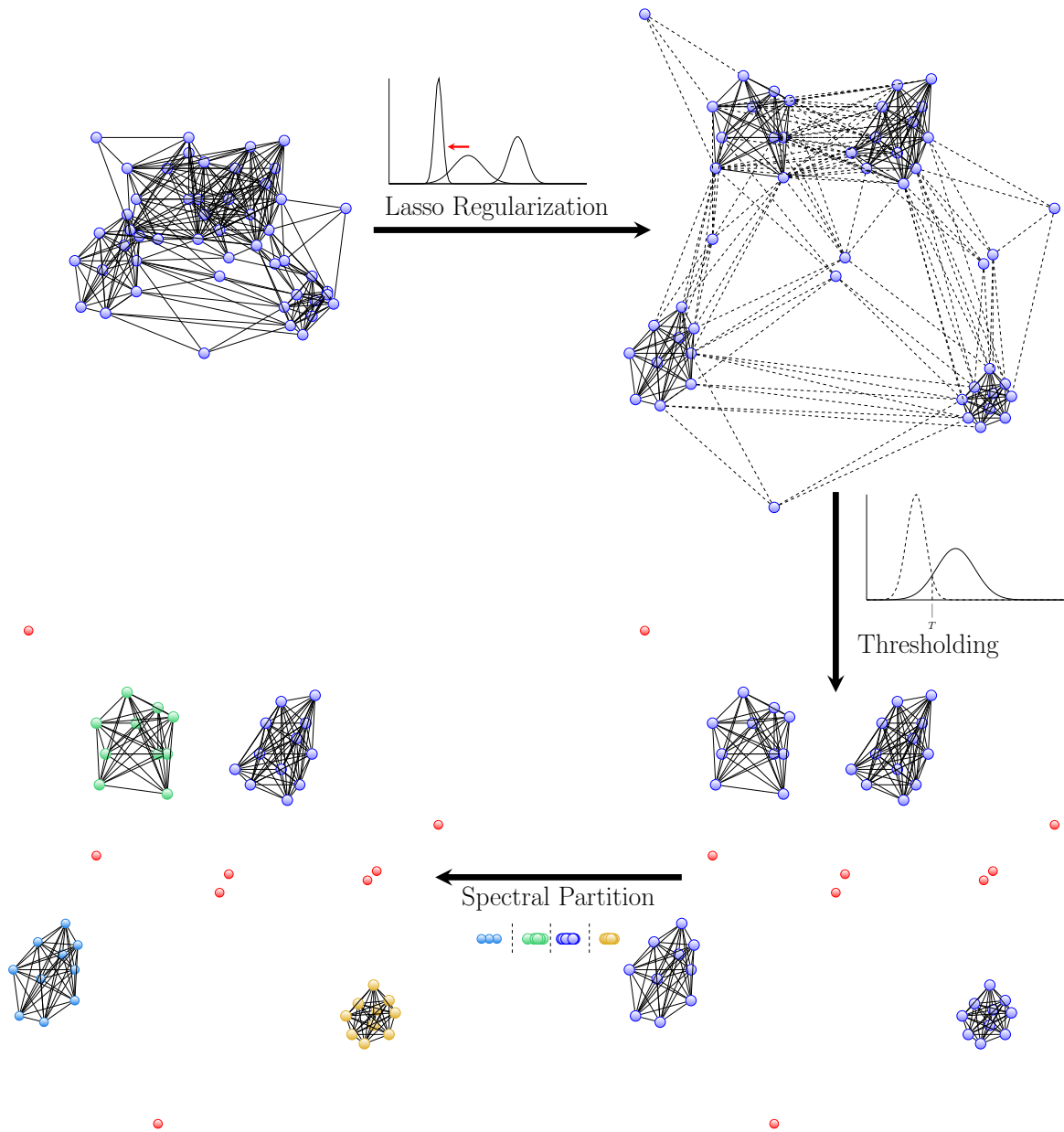


Figure 4.18: The main steps of SPARCODE.

manner. The outliers, which are represented as red points in Figure 4.18, are detected based on the connectivity of vertices in the improved sparse graph model. Finally, fast spectral partitioning is performed on the outlier-free vertices of the robust sparse graph model by mapping each vertex onto a line and applying a balanced partition. The number of communities is estimated using modularity optimization on the partitions. The steps are detailed in the following sections, a pseudocode summary is given in Algorithm 4.

Compute Sparsity Improved Graph Model

Graphical models over undirected graphs are a popular method to exhibit conditional independence structures in multivariate distributions [FHT08, YL07]. Undirected Gaussian graphical models are of particular interest within this PhD project, because in this case, revealing the conditional independence structure is equivalent to the recovery of the support of the precision matrix Θ . The Gaussian graphical model is, therefore, defined as $G = \{V, E, \Theta\}$ with a vertex set $V = \{1, \dots, N\}$ corresponding to random variables, an edge set $E = \{(m, n) \in V | m \neq n, \Theta_{m,n} \neq 0\}$ and a precision matrix Θ . Thus, if the m, n th entry of Θ equals zero, the two corresponding variables are conditionally independent. A very popular approach to estimate a sparse precision matrix is the graphical LASSO [FHT08], which maximizes a penalized Gaussian likelihood

$$\hat{\Theta} = \arg \max_{\Theta \in \mathbb{S}_{++}^o} \{\log |\Theta| - \text{tr}(\mathbf{S}\Theta) - \rho \|\Theta\|_1\}, \quad (4.2)$$

where tr denotes trace, $\mathbf{S} \in \mathbb{R}^{N \times N}$ is the sample covariance matrix of \mathbf{W} and ρ is a sparsity inducing penalty parameter. According to Eq. (4.2), $\hat{\Theta}$ is nonnegative definite.

The graphical LASSO attempts to find the precision matrix Θ based on the penalized Gaussian likelihood function. However, it has been shown that the graphical LASSO solution may not satisfy $\hat{\Theta} \in \mathbb{S}_{++}^o$ [MH12]. Additionally, recent researches on the graph error model showed that for a graph with a constant number of communities, the expected values of the eigenvalues depend on the probability of edge existence within and between different communities [MVO20]. In the SPARCODE algorithm the sign of the coefficients is not relevant for reconstructing edges in the sparsity improved graph model because the sign of the coefficients in the similarity matrix does not effect edge existence probability. Therefore, a sparsely connected graph $\hat{G} = \{V, \hat{E}, \hat{\mathbf{W}}\}$ is obtained by using the (element-wise) absolute value of the estimated inverse covariance matrix $\hat{\Theta}$ as affinity matrix, i.e. $\hat{\mathbf{W}} = |\hat{\Theta}|$.

While graphical LASSO provides an efficient way to compute a solution to Eq. (4.2), the resulting graph structure critically depends on the value of ρ . A key contribution of this work is to provide a simple but effective clustering-oriented strategy to optimize ρ . The intuition behind the SPARCODE approach is to determine a value for ρ so that a sparse graph is obtained that can be easily partitioned by SC techniques. SC techniques map the vertices of a graph onto the real line. They minimize the sum of the squared Euclidean distances between the endpoints of the edges while maintaining the average Euclidean distance between random pairs of mapping points [ARV08].

According to [ARV08], maintaining an average unit Euclidean distance between random pairs of these mapping points may lead to an excellent partition by cutting the line at a random point. Achieving such a mapping is NP hard, though there are several approximations in literature that achieve the sparsest cut [AHK10, ARV09, LR99]. SPARCODE relies on spectral partitioning methods, which use a relaxation to map graph vertices onto the real line, providing connected points stay as close together as possible using squared Euclidean distance [BN01]. Let $\rho_i \in \{\rho_{\min}, \dots, \rho_{\max}\}$ denote the i th candidate penalty parameter in Eq. (4.2). Then, the embedding result $\hat{\mathbf{y}}_i \in \mathbb{R}^N$ of the i th candidate penalty parameter ρ_i can be approximated by minimizing the following objective function as in [BN01],

$$\hat{\mathbf{y}}_i = \arg \min_{\mathbf{y}_i} \frac{1}{2} \sum_{m,n} \|\mathbf{y}_m - \mathbf{y}_n\|_2^2 \hat{w}_{m,n}(\rho_i) \quad s.t. \quad \begin{aligned} \mathbf{y}_i^\top \mathbf{D}_i \mathbf{y}_i &= 1 \\ \mathbf{y}_i^\top \mathbf{D}_i \mathbf{1} &= 0, \end{aligned} \quad (4.3)$$

where $\mathbf{D}_i \in \mathbb{R}^{N \times N}$ is a diagonal weight matrix of $\hat{\mathbf{W}}(\rho_i)$ for a given ρ_i , with weights $d_{m,m} = \sum_n \hat{w}_{m,n}(\rho_i)$ on the diagonal. The vector estimate $\hat{\mathbf{y}}_i = (\hat{y}_1, \dots, \hat{y}_N)^\top$ is known as the Fiedler vector and it shows the algebraic connectivity of a graph [Fie75]. The graph can be partitioned into two subsets by splitting the Fiedler vector, such that $\hat{y}_m \in \hat{\mathbf{y}}_{i,1}$ for $\hat{y}_m \leq 0$ and $\hat{y}_m \in \hat{\mathbf{y}}_{i,2}$ otherwise. Here, $\hat{\mathbf{y}}_{i,j}$ denotes the j th subset of $\hat{\mathbf{y}}_i$ with $j = 1, 2$ and $m = 1, \dots, N$ [ST07].

Based on the Fiedler vector, we propose to measure the polarization score for each candidate ρ_i by evaluating

$$P_{sc}(\rho_i) = \text{med}(\hat{\mathbf{y}}_{i,1}) - \text{med}(\hat{\mathbf{y}}_{i,2}), \quad (4.4)$$

where $\text{med}(\hat{\mathbf{y}}_{i,j})$ denotes the median of the j th subset for $j = 1, 2$. The median is used as a robust location estimate [MMY19, ZKO18] for each subset $\hat{\mathbf{y}}_{i,j}$. Given a set of candidate penalty

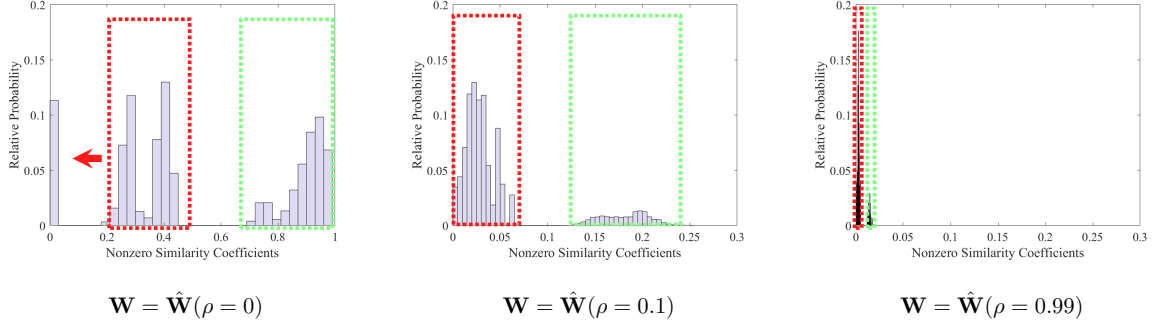


Figure 4.19: The empirical distribution of similarity coefficients for growing penalty parameter values.

parameters $\rho_i \in \{\rho_{\min}, \dots, \rho_{\max}\}$, we estimate ρ by maximizing the polarization score as

$$\hat{\rho} = \arg \max_{\rho = \rho_{\min}, \dots, \rho_{\max}} P_{sc}(\rho), \quad (4.5)$$

where $\hat{\rho}$ denotes the estimated penalty parameter that provides maximum polarization in the given space. The ideas underlying maximizing polarization score are visualized in Figure 4.19. Starting from a densely connected graph model including undesired edges between different communities, the empirical distribution of similarity coefficients is shown for growing values of penalty parameters where similarity coefficients associated with undesired and target edges are highlighted with red and green rectangular boxes, respectively. As can be seen, when the sparsity is further increased, the distinction of undesired and target edges becomes challenging. This is because now edges from both modes are shrunken to zero. Thus, in SPARCODE the penalty parameter is estimated as the value which provides maximum polarization in coefficient space.

To reduce computational complexity, we first evaluate Eq. (4.5) on a coarse grid and then use cubic spline interpolation between $\hat{\rho}$ and its neighboring samples to find the final value $\hat{\rho}$. By means of Eq. (4.5), we propose a problem-dependent tuning of the level of sparsity in Eq. (4.2). This provides us with an initial robust sparse graph model that we will improve in the following step.

Compute a Robust Sparse Graph Model

By maximizing the polarization score via Eq. (4.5), we estimated a graph model $\hat{G} = \{V, \hat{E}, \hat{\mathbf{W}}(\hat{\rho})\}$, for which the edges between different communities are large, while the edges within the community are small. However, the true graph structure may still be hidden due to noise and outliers. It is intuitively clear that sparse outlying entries and noise have fewer

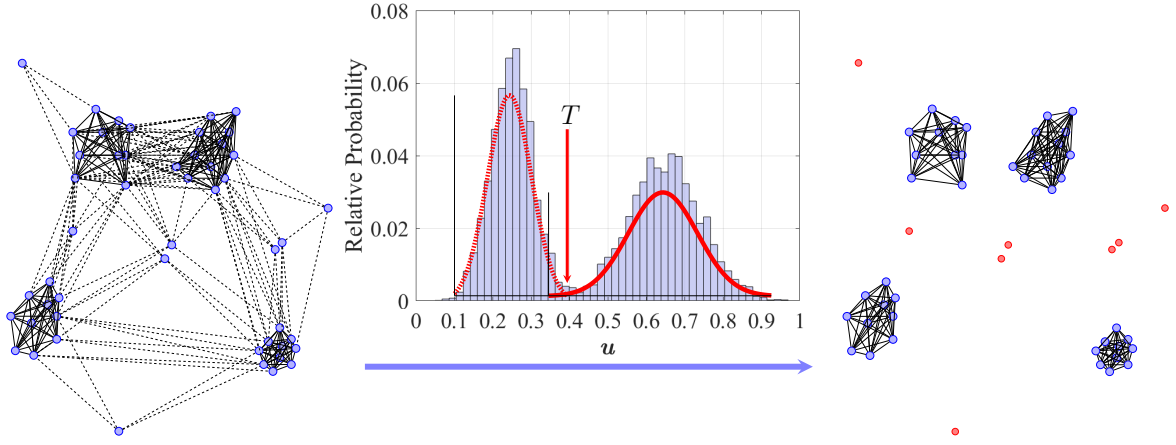


Figure 4.20: The thresholding operation of SPARCODE.

non-zero coefficients in the affinity matrix compared to typical data points, and the value of these coefficients is negligibly small for noise [EV13]. SPARCODE builds upon this graph property to detect outliers and noisy feature vectors by analyzing the connectivity of graph vertices.

Let $\hat{\mathbf{W}}^{(s)}(\hat{\rho}) = [\hat{\mathbf{w}}_1^{(s)}(\hat{\rho}), \dots, \hat{\mathbf{w}}_N^{(s)}(\hat{\rho})] \in \mathbb{R}^{N \times N}$ be the set of column-wise sorted similarity vectors whose n th element denotes the sorted n th similarity vector of the estimated affinity matrix $\hat{\mathbf{W}}(\hat{\rho})$ in ascending order. Then, we obtain the aggregated set of similarity coefficients $\mathbf{u} = \{u_1, u_2, \dots, u_N\} \in \mathbb{R}^N$ by computing

$$u_m = \frac{1}{N} \sum_{n=1}^N \hat{w}_{m,n}^{(s)}(\hat{\rho}), \quad m = 1, \dots, N, \quad (4.6)$$

where u_m denotes the m th element of \mathbf{u} and $\hat{w}_{m,n}^{(s)}(\hat{\rho})$ is the element of $\hat{\mathbf{W}}^{(s)}(\hat{\rho})$ that belongs to the m th row and n th column. From Eq. (4.6), a two-mode Gaussian mixture distribution of the i th similarity coefficient u_m can be written as

$$f(u_m | \Psi) = \sum_{l=1}^2 \chi_l g(u_m; \mu_l, \Sigma_l) \quad (4.7)$$

where $\Psi = \{\mu_l, \Sigma_l\}$ denotes the parameter set of the model for $l = 1, 2$ and $g(u_m; \mu_l, \Sigma_l)$ is the univariate Gaussian probability density function with mean μ_l and variance Σ_l , and χ_l is the mixing coefficient of the l th subset.

As illustrated in Figure 4.20, the two-mode Gaussian mixture model for the coefficient vector

\mathbf{u} may be used to separate the outliers from the typical data points based on their aggregated edge-weights. The left mode will represent the outliers, while the right mode corresponds to typical data points. The parameters μ_l , Σ_l and χ_l in Eq. (4.7) are estimated using an EM algorithm. After a K -means initialization step, the EM algorithm maximizes the log-likelihood function of the coefficient vector \mathbf{u} with respect to the parameters of interest in coupled equations using alternating expectation and maximization steps. In expectation step the probability estimate that u_m belongs to the l th subset, with $m = 1, 2, \dots, N$ and $l = 1, 2$, can be calculated as

$$\hat{\phi}_{m,l}^t = \frac{\hat{\chi}_l^{t-1} g(u_m; \hat{\mu}_l^{t-1}, \hat{\Sigma}_l^{t-1})}{\sum_{n=1}^2 \hat{\chi}_n^{t-1} g(u_m; \hat{\mu}_n^{t-1}, \hat{\Sigma}_n^{t-1})}, \quad (4.8)$$

where $\hat{\phi}_{m,l}^t$ denotes the probability estimate of u_m that belongs to the l th subset at the t th iteration, $\hat{\mu}_l^{t-1}$, $\hat{\Sigma}_l^{t-1}$ and $\hat{\chi}_l^{t-1}$ are the estimated mean, variance, and mixing coefficient of the l th subset in iteration $t-1$, respectively. In the maximization step, the parameters are updated using the current values of $\hat{\phi}_{m,l}$ as

$$\begin{aligned} \hat{\mu}_l^{(t)} &= \frac{\sum_{m=1}^N \hat{\phi}_{m,l}^t u_m}{\sum_{m=1}^N \hat{\phi}_{m,l}^t} \\ \hat{\Sigma}_l^{(t)} &= \frac{\sum_{m=1}^N \hat{\phi}_{m,l}^t (u_m - \hat{\mu}_l^{(t)}) (u_m - \hat{\mu}_l^{(t)})^\top}{\sum_{m=1}^N \hat{\phi}_{m,l}^t} \\ \hat{\chi}_l^{(t)} &= \frac{\sum_{m=1}^N \hat{\phi}_{m,l}^t}{N}. \end{aligned} \quad (4.9)$$

Then, the threshold can be simply evaluated as the interconnection point of the two distributions, i.e.

$$T = \arg \min_{u_m} |\hat{v}_{m,2} - \hat{v}_{m,1}|. \quad (4.10)$$

This thresholding operation is visualized in Figure 4.20, where we show the estimated Gaussian mixture model and the empirical distribution of the aggregated similarity coefficients. The effect of applying the threshold defined in Eq. (4.10), is to remove the edges that are associated to the

left mode of the mixture from the estimated graph model $\hat{G} = \{\hat{V}, \hat{E}, \hat{\mathbf{W}}(\hat{\rho})\}$. As a result, the feature vectors in data matrix \mathbf{X} that have fewer nonzero coefficients with negligibly small values will be isolated in the graph as can be seen from the right graph model in Figure 4.20. Thus, by giving them zero degree, outliers and noisy feature vectors are detected and removed, resulting in a cleaned data matrix $\check{\mathbf{X}} \in \mathbb{R}^{\check{N}}$ and a graph $\check{G} = \{\check{V}, \check{E}, \check{\mathbf{W}}(\hat{\rho})\}$ with \check{V} denoting the estimated clean vertices, and \check{E} and $\check{\mathbf{W}}(\hat{\rho})$ representing the corresponding edges and similarity matrix whose $\check{w}_{m,n}$ th element denotes the weight of the edge between m th and n th feature vector of $\check{\mathbf{X}}$, respectively.

A Fast Spectral Partition based on the Robust and Sparse Graph

Assuming that $\check{G} = \{\check{V}, \check{E}, \check{\mathbf{W}}(\hat{\rho})\}$ is cleaned from outliers and that it is sufficiently sparse, the graph can be partitioned into K communities based on the Fiedler vector with a fast partitioning method. In practice, however, K is unknown and must be estimated. We therefore present an approach to estimate the range $K_{\min} \leq K \leq K_{\max}$ using typical degrees in the graph.

Let $\mathfrak{d} \in \mathbb{Z}^{+\check{N}}$ denote a degree vector whose m th element corresponds to the degree of the m th feature vector in $\check{\mathbf{X}} \in \mathbb{R}^{\check{M} \times \check{N}}$. Further, let $\mathbf{p} \in \mathbb{R}^{\check{N}}$ be the empirical probabilities of occurrence of these degrees in \mathfrak{d} . Finally, let $\mathbf{h} \in \mathbb{R}^{\check{N}/2-1}$ denote the vector of degrees, whose probability is greater than the median of probabilities, i.e. $\text{med}(\mathbf{p})$. Now, the minimum and maximum number of communities $\{K_{\min}, K_{\max}\} \in \mathbb{Z}_+$ can be estimated as

$$\hat{K}_{\min} \approx \frac{\check{N}}{b_{\max} + 1} \text{ and } \hat{K}_{\max} \approx \frac{\check{N}}{b_{\min} + 1}, \quad (4.11)$$

where $b_{\min} = \min\{\mathbf{h}\}$, $b_{\max} = \max\{\mathbf{h}\}$, $b_{\min} + 1$ and $b_{\max} + 1$ represent the minimum and maximum number of vertices in each community, respectively. The intuition underlying Eq. (4.11) is to define a range for a candidate number of communities from the typical connectedness of the graph. Now, for a set of estimated candidate communities $\hat{K}_{\text{cand}} \in \{\hat{K}_{\min}, \dots, \hat{K}_{\max}\}$, the graph can be partitioned using the ascending order sorted Fiedler vector $\check{\mathbf{y}}^{(s)} \in \mathbb{R}^{\check{N}}$ of $\check{G} = \{\check{V}, \check{E}, \check{\mathbf{W}}(\hat{\rho})\}$ as follows.

First, for each mapping result $\check{y}_m^{(s)}$, with $m = 1, \dots, \check{N}$, in the Fiedler vector, the standard deviation of the set containing its two immediate neighbors is computed. The set of immediate neighbors for mapping results $\check{y}_m^{(s)}$, $m \in \{2, \dots, \check{N}-1\}$, is $\{\check{y}_{m-1}^{(s)}, \check{y}_m^{(s)}, \check{y}_{m+1}^{(s)}\}$. For the special cases of $m = 1$ and $m = \check{N}$, the sets are defined as $\{\check{y}_1^{(s)}, \check{y}_2^{(s)}, \check{y}_3^{(s)}\}$ and $\{\check{y}_{\check{N}-2}^{(s)}, \check{y}_{\check{N}-1}^{(s)}, \check{y}_{\check{N}}^{(s)}\}$, respectively.

The standard deviations are collected in a vector defined by

$$\boldsymbol{\sigma}_{\check{\mathbf{y}}^{(s)}} = (\sigma_{\check{y}_1^{(s)}}, \dots, \sigma_{\check{y}_N^{(s)}})^\top. \quad (4.12)$$

Then, to split the graph into an initial balanced partitioning, where each community has the same number of members, we compute the number q_{cand} of mapping results that we ignore in the initial partitioning as

$$q_{\text{cand}} = \text{mod}(\check{N}, \hat{K}_{\text{cand}}). \quad (4.13)$$

The vector $\check{\mathbf{y}}_{\text{ign}}^{(s)} \in \mathbb{R}^{q_{\text{cand}}}$ of initially ignored mapping results is composed of the mapping results that are associated to the largest q_{cand} entries of the vector $\boldsymbol{\sigma}_{\check{\mathbf{y}}^{(s)}}$. The complementary vector $\check{\mathbf{y}}_{\text{rem}}^{(s)}$ is defined by the remaining $\check{N} - q_{\text{cand}}$ entries. Then, the initial partition can be computed by splitting the $\check{\mathbf{y}}_{\text{rem}}^{(s)}$ into \hat{K}_{cand} equally sized communities. Finally, the initially ignored q_{cand} number of mapping results $\check{\mathbf{y}}_{\text{ign}}^{(s)}$ can be assigned by minimizing the distance to the center of initially defined communities on the real line. Now, the estimated label vector of a given candidate number of community $\check{\mathbf{c}}_{\hat{K}_{\text{cand}}} \in \mathbb{R}^{\check{N}}$ is available for all feature vectors of $\check{\mathbf{X}}$ where \check{c}_n denotes the n th feature vector label.

As the goal of SPARCODE is creating a graph that only includes edges within the communities, the number of communities K can be estimated by comparing the candidate models in terms of a suitable clustering quality metric. Modularity is a metric that evaluates the quality of partition with respect to the similarity of feature vectors in an affinity matrix. It gives a high modularity score if a vertex has more edges within the assigned community. The estimator for K , therefore, maximizes the quality of different partitions of the robust graph model, i.e.

$$\hat{K} = \arg \max_{\hat{K}_{\text{cand}} = \hat{K}_{\text{min}}, \dots, \hat{K}_{\text{max}}} \{\text{mod}_{\hat{K}_{\text{cand}}}\}, \quad (4.14)$$

where $\text{mod}_{\hat{K}_{\text{cand}}}$ denotes the modularity score of candidate number of communities \hat{K}_{cand} that can be computed using Eq. 2.7.

Algorithm 4: SPARCODE

Input: An affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$

Step 1: Robust Sparse Graph Model

Step 1.1: Sparsity Improvement

Initialization:

for $\rho_i^{(0)} = \rho_{\min}^{(0)}, \dots, \rho_{\max}^{(0)}$ **do**
 Estimate $\tilde{\mathbf{W}}(\rho^{(0)})$ via Eq. (4.2)
 Map each vertice onto a line using Eq. (4.3)
 Get the Fiedler vector $\hat{\mathbf{y}}_i^{(0)}$
 Split $\hat{\mathbf{y}}_i^{(0)}$ into two subsets for a splitting value $s = 0$
 Calculate the polarization score $P_{\text{sc}}(\rho_i^{(0)})$ via Eq. (4.4)
 Stack $P_{\text{sc}}(\rho_i^{(0)})$ into $\mathbf{p}_{\text{sc}}^{(0)} \in \mathbb{R}^{N_{\rho}^{(0)}}$
end

end

Penalty Parameter Selection:

Find $\rho_i^{(0)}$ which maximizes Eq. (4.5) for an initial set

Regenerate a penalty parameter set $\rho_i = \rho_{\min}, \dots, \rho_{\max}$ over equally spaced N_{ρ} samples

Apply the same framework as in the initialization step

Obtain $\mathbf{p}_{\text{sc}} \in \mathbb{R}^{N_{\rho}}$ for $\rho_i = \rho_{\min}, \dots, \rho_{\max}$

Apply cubic spline interpolation to obtain $\hat{\rho}$

Step 1.2: Robustness and Outlier Detection

Create $\tilde{\mathbf{W}}^{(s)}(\hat{\rho}) \in \mathbb{R}^{N \times N}$ over a set of sorted similarity vectors from $\tilde{\mathbf{W}}(\hat{\rho})$

Get $\mathbf{u} \in \mathbb{R}^N$ via Eq. (4.6)

Estimate $\hat{v}_{m,l}$ for each coefficient where $m = 1, 2, \dots, N$ and $l = 1, 2$

Calculate T using Eq. (4.10)

Cut undesired edges in \tilde{G} using T

Reject outliers whose degree equals zero, $\mathfrak{d} = 0$

Form $\tilde{\mathbf{W}}(\hat{\rho}) \in \mathbb{R}^{\tilde{N} \times \tilde{N}}$ over estimated outlier-free vectors

Step 2: Fast Spectral Partition

Estimate $\hat{K}_{\text{cand}} \in \{\hat{K}_{\min}, \dots, \hat{K}_{\max}\}$ using Eq. (4.11)

for $\hat{K}_{\text{cand}} = \hat{K}_{\min}, \dots, \hat{K}_{\max}$ **do**

 Create $\tilde{\mathbf{y}} \in \mathbb{R}^{\tilde{N}}$ using Eq. (4.3)

 Compute $\sigma_{\tilde{\mathbf{y}}^{(s)}}$ and q_{cand} via Eq. (4.12) and Eq. (4.13)

 Define $\tilde{\mathbf{y}}_{\text{ign}}^{(s)} \in \mathbb{R}^{q_{\text{cand}}}$ and $\tilde{\mathbf{y}}_{\text{rem}}^{(s)} \in \mathbb{R}^{\tilde{N}-q_{\text{cand}}}$

 Apply an initial partition on $\tilde{\mathbf{y}}_{\text{rem}}^{(s)} \in \mathbb{R}^{\tilde{N}-q_{\text{cand}}}$

 Assign $\tilde{\mathbf{y}}_{\text{ign}}^{(s)} \in \mathbb{R}^{q_{\text{cand}}}$

 Form $\check{\mathbf{c}}_{\text{cand}} \in \mathbb{R}^{\tilde{N}}$ for \hat{K}_{cand}

 Get $\text{mod}_{\hat{K}_{\text{cand}}}$ via Eq. (2.7) and stack into $\mathbf{q} \in \mathbb{R}^{N_K}$
end

end

Estimate \hat{K} using Eq. (4.14)

Output: A vector $\check{\mathbf{c}}$ for \hat{K}

4.4.1.1.5 EXPERIMENTAL EVALUATION

In this section, the community detection performance of SPARCODE is benchmarked against state-of-the-art community detection algorithms. We consider a variety of clustering and graph partitioning data sets to demonstrate the applicability of SPARCODE in both of these community detection settings. We select as competitors the following community detection algorithms: Newman’s Greedy Algorithm (NGA) [New04], Le Martelot (Martelot) [MH11], Newman’s eigenvector method (NE) [New06], singular value decomposition-based community detection (SVD) [SD11], the Louvain method (Louvain) [BGL08], the Bayesian approach (BC) [HW08], Bayesian nonnegative matrix factorization (BNMF) [PRE11], the Combo method [SCB14], the Infomap method (MAP) [BEL14] and the density peak-based overlapping community detection method (DenPeak) [BYS17]. The SVD method is applicable only on bipartite graphs, for details see [SD11]. Bayesian cluster enumeration (BCE) [TMZ18], dip-means and kernel dip-means (K. dip-means) [KL12], x-means [PM00], Gaussian k-means (g-means) [HC04] and DBSCAN [EKS96] are used as cluster-based competing approaches. The performance measures are evaluated both on synthetic and real-world data sets.

All SPARCODE implementations use the same default parameters as follows: $\rho_{\min}^{(0)} = 0.1$, $\rho_{\max}^{(0)} = 0.99$, $N_{\rho}^{(0)} = N_{\rho} = 5$. A MATLAB code for SPARCODE is available at:

<https://github.com/A-Tastan/SPARCODE>

Performance Measures

The empirical probability of detection p_{det} , the conductance cond , the modularity score mod and the computation time t are used as performance measures. The empirical probability of detection is estimated as

$$p_{\text{det}} = \frac{1}{N_E} \sum_{m=1}^{N_E} \mathbb{1}_{\{\hat{K}=K\}}, \quad (4.15)$$

where N_E denotes the total number of performed experiments, \hat{K} is the estimated number of communities and $\mathbb{1}_{\{\hat{K}=K\}}$ is the indicator function that is defined as

$$\mathbb{1}_{\{\hat{K}=K\}} = \begin{cases} 1, & \text{if } \hat{K} = K \\ 0, & \text{otherwise} \end{cases}. \quad (4.16)$$

The conductance and modularity score of the estimated community number can be calculated using Eqs. (2.4.4) and (2.7), respectively. The modularity score of SPARCODE is computed based on the affinity matrix of the sparsity improved robust graph model.

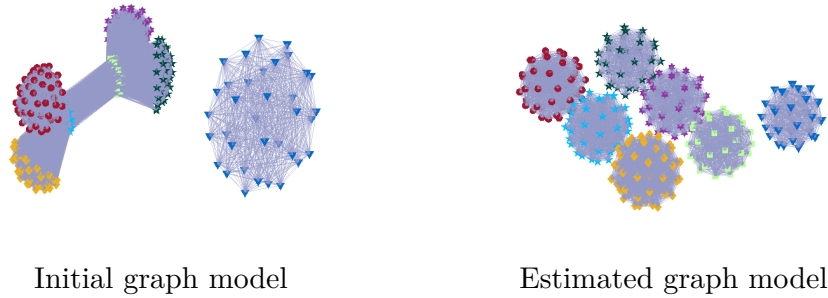


Figure 4.21: Graphical models of Scenario 1.

Method	\hat{K}	mod	cond	t
NGA [New04]	4	0.595	0.069	404.389
Martelot [MH11]	4	0.576	0.120	0.046
NE [New06]	1	0	0	0.023
Louvain [BGL08]	5	0.597	0.116	0.023
BC [HW08]	7	0.525	0.313	1.910
BNMF [PRE11]	5	0.597	0.116	3.855
Combo [SCB14]	5	0.597	0.116	0.910
MAP [BEL14]	4	0.540	0.048	0.092
DenPeak [BYS17]	5	0.597	0.116	0.023
SPARCODE	7	0.643	0.201	0.768

Table 4.1: Performance of 10 graph-based approaches on Scenario 1 where $K = 7$.

Synthetic Graph Model 1: Correlated Communities Study with $K = 7$

An undirected weighted graph model is considered. It is a variant of the stochastic block model (SBM) with constant densities within and between communities, respectively, where the vertices belonging to the same community are more densely connected. The SBM of Scenario 1, i.e. $\text{SBM}_1(N, \boldsymbol{\alpha}, \mathbf{Q})$, is defined for $N = 300$ vertices, a probability vector $\boldsymbol{\alpha} \in \mathbb{R}^{\hat{K}}$ specifying the distribution of the vertices in the $K = 7$ communities, and a symmetric connectivity matrix $\mathbf{Q} \in \mathbb{R}^{K \times K}$ whose $q_{i,j}$ th element denotes the probability of an edge between i th and j th community block. In the designed undirected weighted graph model, the first six communities inhibit correlation between communities in addition to correlations within the community. In contrast, the seventh community only has correlations within the community. Moreover, the density of the seventh community is assumed to be higher compared to that of the other six. For illustration purposes, the graph for the designed affinity matrix is shown on the left side of Figure 4.21, all parameters to generate the data set are given in Appendix B.1.1.

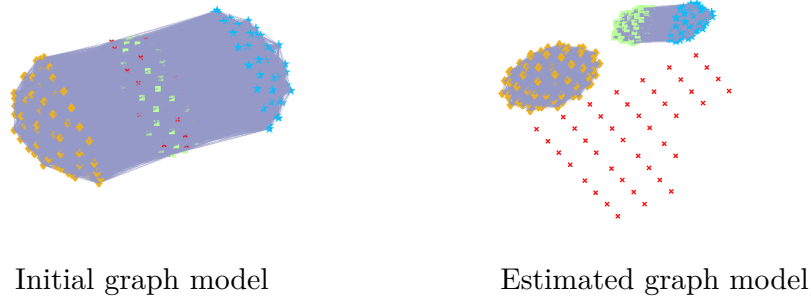


Figure 4.22: Graphical models of Scenario 1.

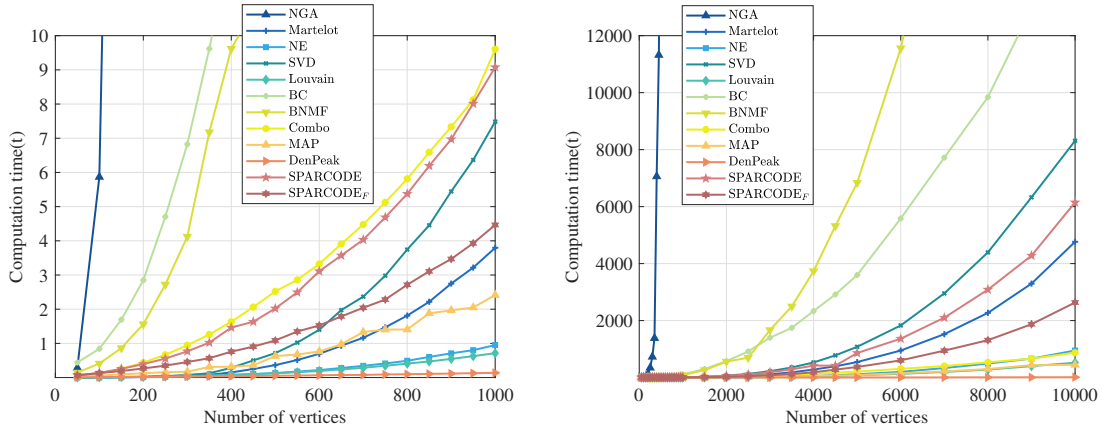
Tab. 4.1 summarizes the community detection performance results for Scenario 1 in terms of the estimated community number, the modularity score, the conductance score and the computation time. The considered setting is challenging for most algorithms, with the exception of BC, all benchmark community detection algorithms underestimate the number of clusters due to the correlations between communities. SPARCODE correctly finds the number of communities and outperforms BC in terms of the computation time, the modularity and conductance scores. As shown on the right side of Figure 4.21, the estimated sparsity improved robust graph model clearly partitions the network into seven communities.

Synthetic Graph Model 2: Robustness Study with $K = 3$

An undirected weighted graph consisting of three communities in the presence of outliers is created by again using the community block density-based variant of the SBM for Scenario 2, $\text{SBM}_2(N, \alpha, \mathbf{Q})$. The communities are correlated with each other in addition to exhibiting strong correlations within. The outliers correlate equally with all communities with negligibly small correlation coefficients. The graph for the designed affinity matrix of Scenario 2 is shown on the left side of Figure 4.22, all parameters to generate the data set are given in Appendix B.1.2. While the NE, NGA, BC and BNMF community detection algorithms overestimate the community number because of outliers, the remaining competitor community detection methods estimate the community number correctly. However, for the competing methods, the outliers cause a noticeable drop in modularity with a poor performance in terms of conductance except for MAP method. Although the MAP method shows the best performance with respect to conductance, it has a poor performance in terms of modularity. By contrast, the SPARCODE algorithm, overcomes these problems by detecting these outliers in the graph model and providing robustness by minimizing the effect of outliers on the quality of the partition. The estimated graph model in

Method	\hat{K}	mod	cond	t
NGA [New04]	115	0.219	0.632	775.639
Martelot [MH11]	3	0.326	0.297	0.054
NE [New06]	4	0.288	0.429	0.036
Louvain [BGL08]	3	0.326	0.297	0.041
BC [HW08]	10	0.323	0.301	6.879
BNMF [PRE11]	4	0.325	0.300	3.946
Combo [SCB14]	3	0.325	0.297	0.88
MAP [BEL14]	3	0.208	0.089	0.162
DenPeak [BYS17]	3	0.236	0.336	0.025
SPARCODE	3	0.496	0.151	0.753

Table 4.2: Performance of 10 graph-based approaches on Scenario 2 where $K = 3$.



(a) Computation time for growing number of vertices (zoom on (b) Computation time for growing number of vertices. small networks).

Figure 4.23: Computation time for a growing number of vertices for Scenario 2. The results are reported in seconds. The upper figure zooms into the region concerning networks up to a size of 1000 vertices.

Figure 4.22 shows that, based on the robust sparse graph model, a separation of the communities is possible and outliers can easily be distinguished from other data points.

Computation Time

The computation time is reported as function of the number of vertices in the network. All experiments are performed based on Scenario 2 that was explained in the previous section. All implementations are in MATLAB using the default parameters given by the authors, except for the Combo and the MAP algorithms, for which we use the available C and Python implementations, respectively. For the proposed SPARCODE algorithm, we compare two

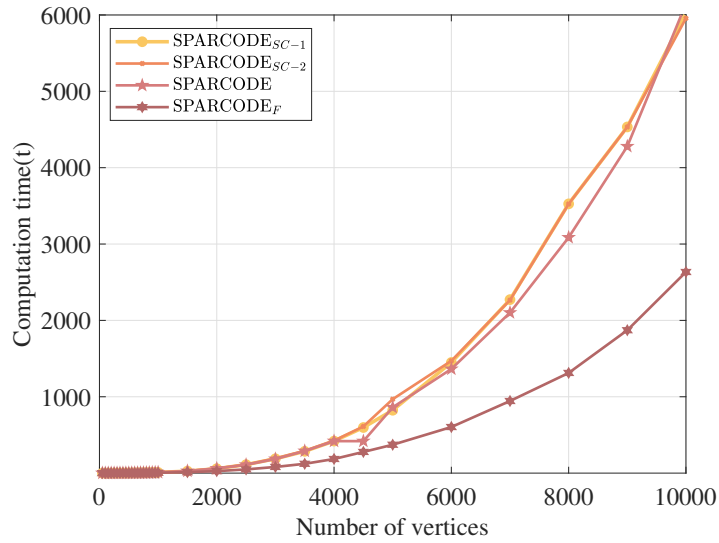


Figure 4.24: Performance of different partitioning methods in terms of computational complexity. The results are reported in seconds.

possible implementations: The method referred to as SPARCODE estimates the Fiedler vector by using the MATLAB function `eig` for a complete eigen-decomposition and a faster version SPARCODE_F estimates the Fiedler vector from a subset of eigenvectors using the `eigs` function for the two smallest eigenvalues. The results are summarized in Figure 4.23. For sample sizes up to 1000 vertices, SPARCODE is comparable to Combo (see Figure 4.23a), though for large networks, the computation time of SPARCODE grows quicker (see Figure 4.23b). SPARCODE_F is faster than Martelot for large networks whereas the computation time for Martelot is smaller for small sample sizes.

The SPARCODE method would allow for different options to do the partitioning. We therefore also compare the computation time of the proposed partitioning approach to that of the well-known SC method [NJW01] in the original form using *K*-means SPARCODE_{SC-1} and a simple plug-in robustification version that use *K*-medoids SPARCODE_{SC-2}. All implementations use a subset of eigenvectors, except for SPARCODE, which uses a complete eigen-decomposition. As can be seen, SPARCODE, SPARCODE_{SC-1} and SPARCODE_{SC-2} have a similar execution time, although SPARCODE uses a complete eigen-decomposition. Finally, SPARCODE_F provides a significant improvement in terms of the computation time.

ζ	\hat{K} for Different Community Detection Methods									
	NGA	Martelot	NE	Louvain	BC	BNMF	Combo	MAP	DenPeak	SPARCODE
0.1	3	3	3	3	3	3	3	3	3	3
0.2	3	3	3	3	3	3	3	1	3	3
0.3	1	3	4	3	3	3	3	1	3	3
0.4	1	3	4	3	3	2	3	1	3	3
0.5	1	3	4	3	2	1	3	1	3	2
0.6	1	2	20	4	2	1	3	1	2	2
0.7	1	2	22	4	2	1	3	1	2	2

Table 4.3: Performance of graph-based approaches on LFR data sets for $\beta_w = 1$ and $K = 3$.

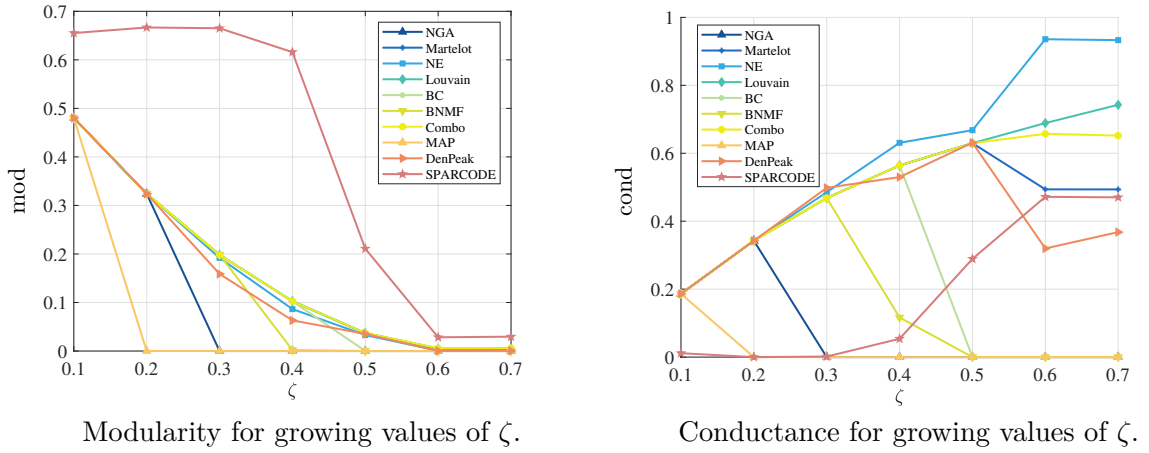


Figure 4.25: Modularity and conductance for a growing values of ζ for LFR networks.

Synthetic Graph Model: Lancichinetti–Fortunato–Radicchi Networks (LFR)

Lancichinetti–Fortunato–Radicchi networks [LFR08] are adapted for undirected weighted graphical models using the following parameters in addition to the default setting: number of vertices $N = 300$, average degree $\bar{d} = 100$, maximum degree $d_{\max} = 100$, mixing parameter for weights $\zeta = 0.1, \dots, 0.7$, exponent for the weight distribution $\beta_w = 1, 1.5, 2$. As it is indicated in [LFR08], for the mixing parameters beyond $\zeta = 0.5$ the communities are not defined in the strong sense such that each vertex has more neighbors in its own community than in the others. Therefore, the target networks can be determined such that $\zeta < 0.5$.

An example of the community detection results, for $\beta_w = 1$, is summarized in Tab. 4.3. Changing the values of β_w did not lead to a considerable performance difference and thus,

the results are not reported in detail in terms of β_w . The community detection performances are summarized using modularity and conductance metrics in Figure 4.25. As can be seen, SPARCODE estimates the community number in the target region with a considerably high modularity and low conductance. The results also show that even though Martelot, Louvain, Combo and DenPeak algorithms estimate the community number correctly after $\zeta = 0.5$ bound, they show noticeably poor performance both in terms of modularity and conductance, which indicates that vertices may assigned to the wrong communities.

Real-World Benchmark Graph Models

The performance is benchmarked on the following seven real-world networks:

Zachary's Karate Club (Karate): The network is a social bipartite network that consists of friendship between 34 members of a karate club [Zac77].

Dolphins: The network is a social network that consists of social interactions of 62 dolphins with $K = 2$ communities based on the reaction after a dolphin left from the group [LSB03].

American College Football (Football): The network represents 115 US college teams and the games that they played [GN02]. The ground truth for the community number is 12.

Political Blogs (P. Blogs): The network consists of blogs about US politics with 1490 vertices and $K = 2$ communities as "liberal" and "conservative" [AG05].

Jazz Musicians (Jazz): The network consists the collaboration of jazz musicians which can be divided into $K = 4$ communities based on cities where bands are recorded [GD05].

Carpinteria Salt Marsh (C.S.M.): The network is a type of food web which can be divided into $K = 2$ based on species as parasites and free-living or $K = 4$ subwebs based on links, e.g. parasites-parasites, predator-parasites [LDK06].

Caenorhabditis Elegans (C. Elegans): The biological network examines the neuronal layout of C. Elegans for 279 neurons that can be partitioned into $K = 3$ communities [CHC06].

The cosine similarity, defined in Eq. (2.1), is used to obtain the affinity matrix \mathbf{W} for all networks except for the political blogs network, which is examined with Pearson's linear correlation coefficients as defined in Eq. (2.2) because none of the algorithms estimated the community number correctly with the cosine similarity.

Network		K	NGA	Martelot	NE	SVD	Louvain	BC	BNMF	Combo	MAP	DenPeak	SPARCODE
Karate [Zac77]	\hat{K}		2	2	3	2	2	3	2	2	2	2	2
	mod	2	0.292	0.292	0.244	0.262	0.292	0.276	0.292	0.292	0.292	0.164	0.237
	cond		0.201	0.201	0.404	0.237	0.201	0.241	0.201	0.201	0.201	0.247	0.179
Dolphins [LSB03]	\hat{K}		2	3	6	2	4	2	3	4	2	4	2
	mod	2	0.364	0.415	0.373	0.364	0.421	0.364	0.371	0.421	0.364	0.342	0.382
	cond		0.054	0.242	0.408	0.054	0.305	0.054	0.126	0.306	0.054	0.263	0.113
Football [GN02]	\hat{K}		7	4	8	-	7	5	5	7	7	12	8
	mod	12	0.422	0.406	0.371	-	0.427	0.429	0.424	0.427	0.427	0.351	0.499
	cond		0.405	0.316	0.476	-	0.409	0.370	0.373	0.407	0.406	0.539	0.373
P. Blogs [AG05]	\hat{K}		-	541	1	1	505	7	503	4	16	499	2
	mod	2	-	0.478	0	0	0.531	0.261	0.521	0.531	0.469	0	0.395
	cond		-	-0.05	0	0	0.112	0.418	0.117	0.117	-	0	0.087
Jazz [GD05]	\hat{K}		20	21	7	-	21	10	22	3	3	21	4
	mod	4	0.268	0.376	0.316	-	0.379	0.346	0.376	0.379	0.261	0.266	0.322
	cond		0.094	0.278	0.462	-	0.271	0.296	0.264	0.275	0.088	0.297	0.404
C.S.M. [LDK06]	\hat{K}		10	12	8	1	13	10	14	5	6	11	2
	mod	2,4	0.199	0.261	0.218	0	0.269	0.237	0.242	0.269	0.226	0.228	0.272
	cond		0.019	0.342	0.593	0	0.287	0.075	0.064	0.287	0.036	0.031	0.163
C. Elegans [CHC06]	\hat{K}		81	3	10	-	3	5	3	3	1	2	3
	mod	3	0.170	0.236	0.178	-	0.246	0.180	0.227	0.246	0	0.034	0.246
	cond		0.716	0.265	0.651	-	0.355	0.273	0.284	0.355	0	0.076	0.341
Average Results	\bar{P}_{det}		0.333	0.286	0	0.5	0.286	0.143	0.286	0.286	0.286	0.286	0.857
	mod		0.286	0.352	0.243	0.156	0.366	0.299	0.351	0.367	0.291	0.198	0.336
	cond		0.248	0.227	0.428	0.073	0.277	0.247	0.204	0.278	0.109	0.207	0.237

Table 4.4: Performance of graph-based approaches on well-known networks. The results whose computation takes more than 12 hours and nontarget networks for SVD method are denoted as "-".

The community detection results are summarized in Tab. 4.4. The results include the estimated community number, the modularity and the conductance scores for all cases for which the computation time was lower than 12 hours. The community detection results of the SVD approach are given for bipartite networks, only. As can be seen, the SPARCODE algorithm shows the best overall performance, and estimates the community number correctly for all networks, except for the Football network. In particular, it provides reasonably good modularity and conductance scores as being the fourth algorithm that provides maximum modularity with a minimum conductance.

Real-World Radar-Based Gait Analysis Data Sets

The performance is benchmarked on the following two radar data sets:

Gait Data Set: As detailed in [SAZ19], the experimental data was collected in an office environment at Technische Universität Darmstadt using a 24 GHz radar system. The

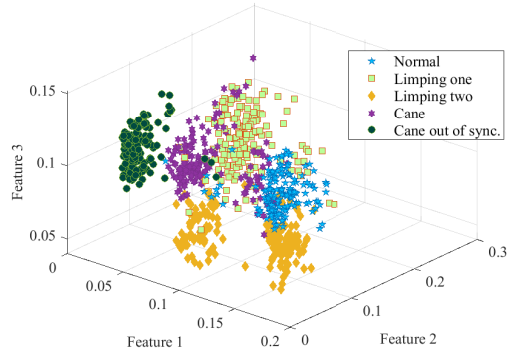


Figure 4.26: Scatter plot for three important features of Gait data belonging to five object communities.

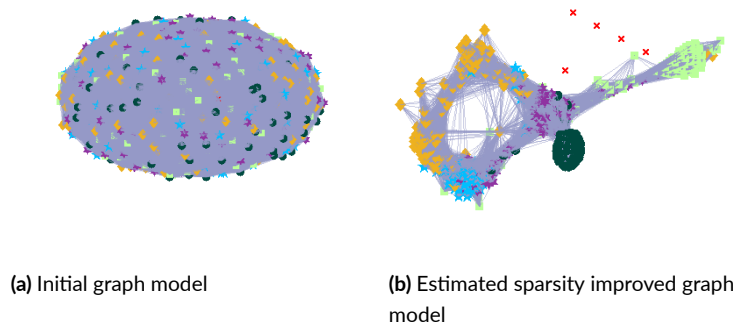


Figure 4.27: Graphical models of Gait data set.

data can be grouped into $K = 5$ different object communities, which are normal walk (‘Normal’), limping with one leg (‘Limping one’), limping with two legs (‘Limping two’), walking with a cane (‘Cane’) and walking with a cane out of synchronization (‘Cane out of sync.’). The data contains 16 measurements per subject (8 towards, 8 away from the radar) and ten subjects. The duration of each measurement is equal to six seconds. In total, 800 measurements for five different gait communities of ten subjects were used in our experiments. As an illustration, a scatter plot of three important features of the radar-based human gait data is shown in Figure 4.26. It can be seen that the outlying observations corresponding to the ‘Cane’ community overlap with the true observations of ‘Normal’. Moreover, the outliers of ‘Limping two’ have a considerable sample size and strong correlations with ‘Normal’, which makes it a challenging scenario for any community detection algorithm.

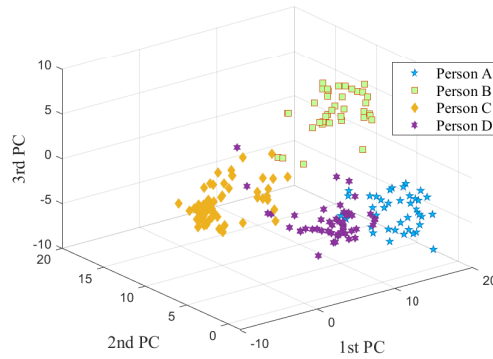


Figure 4.28: Scatter plot for the first three principal components of Person Id. data over four object communities.

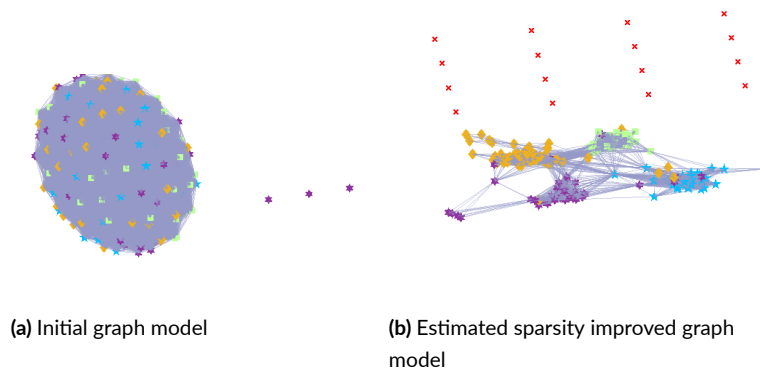


Figure 4.29: Graphical models of Person Id. data set.

In order to represent the underlying structure, for SPARCODE, the similarity measures that are defined in Eq. (4.1) and Eq. (2.2) are used, respectively, and the resulting graph is shown in Figure 4.27a. Clearly, the separation into communities from such a graph is extremely difficult. In contrast, the sparsity improved graph model shown in Figure 4.27b better reveals separated communities.

Person Id. Data Set: As detailed in [TSM18], the experimental data has been collected in an office environment at Technische Universität Darmstadt using the same radar system as explained for the previous data set. The data can be grouped into $K = 4$ object communities, each representing a different person. The data set includes the measurements of four test subjects that are walking slowly and without swinging their arms, towards and away from the radar. The duration of recordings is equal to six seconds and the number of measurements is equal to 13 for each person. In total, 187 stride pairs that are obtained

from 52 observations of the four subjects are used in community detection.

The scatter plot of the Person Id. data set containing $K = 4$ different persons is shown in Figure 4.28. Even though, judging based on visual impression, the examination of the Person Id. data set seems easier in comparison to human gait data, it still has communities that correlate with each other which makes the detection of the community number very difficult.

Just like for the previous radar data set, SPARCODE uses the similarity measures that are defined in Eq. (4.1) and Eq. (2.2), respectively. The initial and the sparsity improved robust graph models of the Person Id. data set are shown in Figure 4.29a and Figure 4.29b, respectively. As can be seen, the estimated sparsity improved graph model is separable into four communities by eliminating the outliers that are of zero degree. Therefore, a simple graph partitioning method is sufficient to partition such a robust sparse graph model into the correct number of communities.

Real-World Cluster Analysis Benchmark Data Sets:

In this section, we benchmark the performance of SPARCODE on eight well-known data sets from the UCI Machine Learning Repository. These are:

Fisher's Iris Data Set (Fisheriris): The data set includes 150 observations from three species of the Iris flower [Fis36].

Ionosphere Data Set: The data set includes 351 radar returns from the ionosphere in order to define quality for further analysis [SWH89]. The subspace number is equal to two.

Parkinson Acoustic Data Set (Parkinson A.): The data set consists of replicated acoustic features of Parkinson's disease with 240 instances from two communities which are "healthy" and "patient" [NPC16].

Diabetic Retinopathy Debrecen Data Set (D. Retinopathy): The data set includes image-based features of diabetic retinopathy with 1151 observations from two object communities [AH14].

Connectionist Bench Data Set (Sonar): The data set includes 208 observations of $K = 2$ communities based on sonar returns collected from a metal cylinder and a cylindrically shaped rock positioned on a sandy ocean floor [GS88].

Data Set	\hat{K} for Different Community Detection Methods											
	NGA	Martelot	NE	SVD	Louvain	BC	BNMFCombo	MAP	DenPeak	SPARCODE	K	
Gait [SAZ19]	-	3	1	-	3	5	5	3	5	17	5	5
Person Id. [TSM18]	9	11	1	-	8	8	8	5	6	3	4	4
Fisheriris [Fis36]	1	2	2	-	2	2	2	2	1	3	3	3
Ionosphere [SWH89]	109	106	1	1	101	10	99	6	2	124	2	2
Parkinson A. [NPC16]	1	1	55	1	1	2	1	1	1	2	2	2
D. Retinopathy [AH14]	-	2	2	1	2	2	2	2	1	2	2	2
Sonar [GS88]	1	2	28	1	3	2	2	2	1	2	2	2
QSAR Bioconcentration [GCV16]	-	3	1	-	3	7	3	3	2	80	3	3
Cardiotocography [ABG00]	-	2	206	-	2	15	3	2	1	1	3	3,10
Divorce Predictors [YAI19]	3	3	1	1	3	7	1	2	1	1	2	2

Table 4.5: Performance of graph-based approaches on clustering data sets. The results whose computation takes more than 12 hours and nontarget networks for SVD method are denoted as "-".

QSAR Bioconcentration Classes Data Set (QSAR Bioconcentration): The data set consists of the bioconcentration factor of 779 chemicals to determine mechanism of bioconcentration [GCV16]. The data set can be partitioned into three communities.

Cardiotocography Data Set: The Cardiotocography data set consists of 2126 observations of fetal cardiotocograms which can be partitioned into three communities in terms of fetal state or ten communities based on morphologic pattern [ABG00].

Divorce Predictors Data Set: The data set consists of 170 observations from two object communities using divorce predictors scale [YAI19].

Comparisons with Graph-based Approaches

The graphs for all clustering data sets are designed using Pearson’s linear correlation coefficients as in Eq. (2.2), except for the Ionosphere and Cardiotocography data sets, where the graphs are designed as the same procedure with Gait and Person Id. data sets.

The estimated community numbers, for all community detection algorithms whose computation time is less than 12 hours, are summarized in Tab. 4.5. Again, the SVD is only applicable for bipartite networks. SPARCODE correctly estimates the number of clusters for all data sets and outperforms all its competitors. None of the competitor community detection algorithms is able to correctly estimate the community number of the Person Id. data set correctly. These overestimate it in most cases, which can be explained by the considerable number of outliers.

In addition to the accuracy in terms of the estimated number of communities, the results are also evaluated with respect to partitioning quality and computation time. To report the

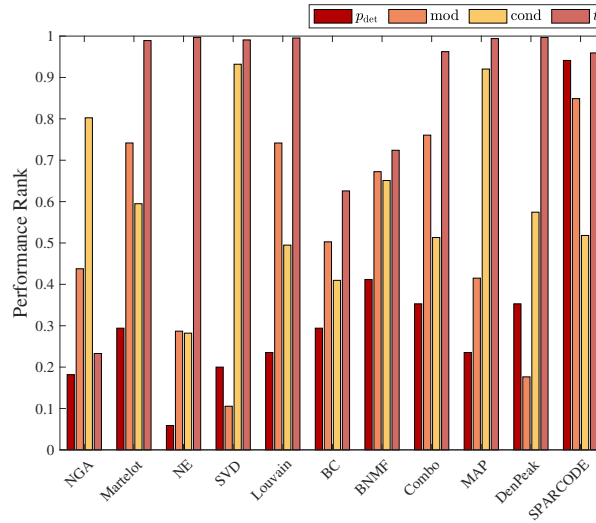


Figure 4.30: Empirical probability of detection and average normalized performance rank of each algorithm in terms of modularity, conductance and computation time.

modularity, the conductance and computation time in the same scale as the probability of detection, the performances of all algorithms are ranked and then averaged over all real data sets. The performance measures of the SVD approach are evaluated on the bipartite graphs, only. All results are summarized in Figure 4.30. As can be seen from the figure, SPARCODE achieves the best performance reaching a probability of detection $p_{\text{det}} = 0.94$ while the strongest competitor (BNMF) follows with $p_{\text{det}} = 0.41$.

Based on quality of partition, SPARCODE achieves the best performance with averaged modularity and conductance rank score of 0.68; Martelot, MAP and BNMF follow closely with 0.67, 0.67 and 0.66, respectively. Although SVD shows the best performance in terms of conductance, it has the worst performance with a considerable difference in terms of its modularity. NE, DenPeak and Louvain methods are the best algorithms with respect to computation time while SPARCODE is the seventh-fastest approach with 0.96 averaged execution time performance rank.

To summarize the performance of graph-based community detection approaches on real data sets, the detailed performance measures that are reported in Figure 4.30 are further aggregated by equally weighting all performance measures. The overall performance of the 11 competing methods is summarized in Figure 4.31. As can be seen, SPARCODE achieves the highest overall performance score of 0.82 whereas, Martelot is the best competitor with a score of 0.66. Combo, MAP and Louvain follow with an overall performance score of 0.65, 0.64 and 0.62, respectively.

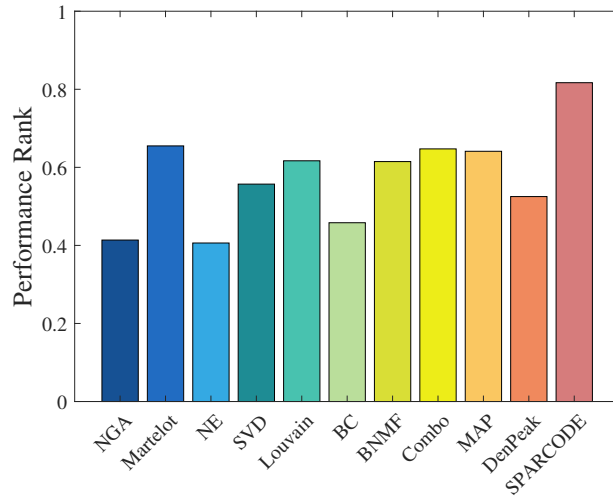


Figure 4.31: Performance of different algorithms based on equal weights on performance metrics.

Data Set	\hat{K} for Different Community Detection Methods							
	BCE	dip-means	K. dip-means	x-means	g-means	DBSCAN	SPARCODE	K
Gait [SAZ19]	7	4	3	176	9	6	5	5
Person Id. [TSM18]	4	4	3	22	10	3	4	4
Fisheriris [Fis36]	3	2	2	31	3	2	3	3
Ionosphere [SWH89]	2	31	1	32	2	2	2	2
Parkinson A. [NPC16]	1	1	1	47	1	1	2	2
D. Retinopathy [AH14]	1	2	2	199	16	1	2	2
Sonar [GS88]	1	1	1	29	1	1	2	2
QSAR Bioconcentration [GCV16]	5	4	6	174	55	1	3	3
Cardiotocography [ABG00]	8	4	4	435	51	2	3	3,10
Divorce Predictors [YAI19]	1	2	2	21	3	1	2	2

Table 4.6: Performance of cluster-based approaches on clustering data sets. The results whose computation take more than 12 hours and nontarget networks for SVD method denoted as "-".

Comparisons with Cluster-based Approaches

In this section, the SPARCODE algorithm is compared with six well-known clustering methods that estimate the number of communities K . The community detection results of different approaches are summarized in Tab. 4.6. As can be seen, none of the competitor algorithms performs well in the highly contaminated Gait data set. However, generally speaking, cluster-based approaches show better performance compared to competitor graph-based approaches on Person Id., Fisheriris and ionosphere data sets, except for x-means, which generally largely overestimates the number of communities.

The overall community detection performance of all algorithms is summarized in Figure 4.32. The SPARCODE method clearly outperforms existing cluster-based community detection

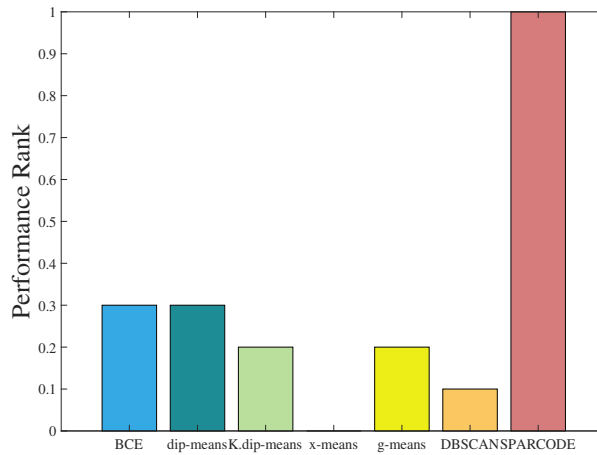


Figure 4.32: Performance of different algorithms based on rank in terms of probability of detection.

approaches in terms of probability of detection and outperforms the best competitors (BCE and dip-means), which have a performance score of 0.3, by a large margin.

4.4.1.1.6 CONCLUSION

We proposed SPARCODE, a community detection method that uses spectral partitioning based on estimating a robust and sparse graph. The level of sparseness is controlled by maximizing the modularity of the graph. SPARCODE includes a graph construction-based outlier detection method to increase robustness. Overall, when compared to both cluster-based and graph-based community detection algorithms on real and synthetic data sets, SPARCODE achieves the highest community detection performance providing a high quality of partition at a reasonable computation time compared to existing graph-based approaches. Judging from its applicability in a large variety of problems, compatibility with different graph-based methods and success in highly contaminated data sets, SPARCODE is a promising new algorithm for performing community detection in a robust manner.

4.4.1.2 FAST AND ROBUST SPARSITY-AWARE BLOCK DIAGONAL REPRESENTATION

4.4.1.2.1 INTRODUCTION

A block diagonally structured affinity matrix represents clusters of feature vectors by non-zero coefficients that are concentrated in blocks. Such a structure is an informative model to describe hidden relationships and it has numerous applications, e.g., denoising [KY19, DBE16], recognition [ZXS17], semi-supervised learning [QWZ21, LLZ15], subspace learning and clustering/classification [XWW21, LWS20, DPC19, WHG15, LMZ12].

Commonly used existing BDR methods impose structure on the affinity matrix using regularization with BD priors, e.g. based on a low-rank property [XTX15, LLY12, LY11, LLY10], sparsity [FLW21, WZW17, EV13] or a known number of blocks K [TMZ22, LFL18, XGL17, FLX14]. For example [FLX14], which is one of the current benchmark BDR methods, controls the number of connected components in the affinity matrix by imposing a rank constraint on the Laplacian matrix. An alternative popular approach [LFL18], proposes a K -block regularizer that is defined by the sum of the K smallest eigenvalues of the Laplacian matrix to compute a BD affinity matrix. A major challenge of these methods is the need to determine appropriate BD priors which play a crucial role in achieving accurate BDR results. Due to its key role in BDR methods, the determination of sparsity/low-rankness level has been intensively researched from different viewpoints, e.g. similarity coefficients' distribution [TMZ21], connectedness [NH11], geometric analysis [ARV09] and supervised learning [MDD18, GCC15]. Recently, in [TMZ22], an alternative unsupervised approach based on eigenvalues has been proposed to deduce the sparsity level in a BD matrix. The eigenvalue analysis is, however, is restricted to the setting of independent blocks.

A further significant challenge when working with real-world data is that heavy-tailed noise and outliers [ZKO18, RL05], might obscure the eigenvalue structure in corrupted data sets which results in a performance degradation for BDR approaches that rely on estimating eigenvalues to determine connectedness. To illustrate the necessity for robustness, we can recall the graph partitioning application that is shown in Figure 4.33 for a defined level of sparsity using the well-known handwritten digit samples from the MNIST data base [HS98]. In the exemplary graph model, the red edges represent connections to outliers while the remaining edges are the informative edges, where green, blue and yellow lines represent the within-cluster edges of digits 9, 4 and 3, respectively. The red ellipses indicate cluster assignments that are computed based on the general graph partitioning principle in which the number of edges that cross the cut is minimized [ARV08]. As can be seen, unconnected outlying digit samples (Type I outliers) are assigned into a

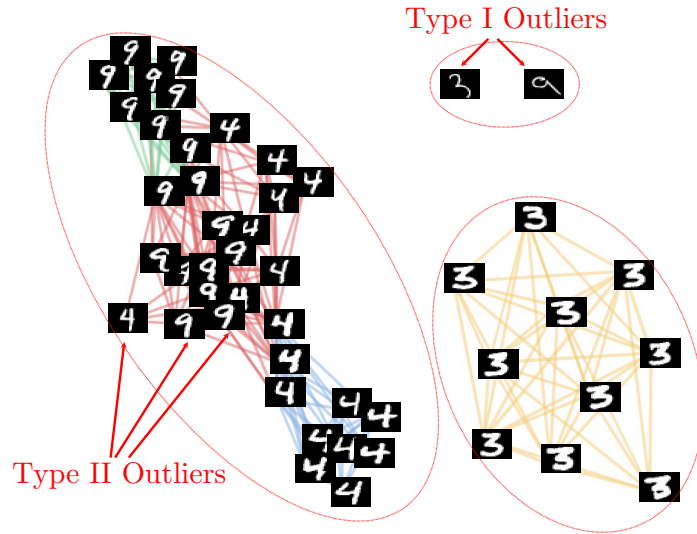


Figure 4.33: Exemplary graph partitioning digit samples from MNIST data base [HS98].

small cluster while a different type of outliers (Type II outliers) that create false positive connections between multiple clusters cause a merging of characters four and nine into one large cluster.

In [TMZ23], we have proposed the *Fast and Robust Sparsity-Aware Block Diagonal Representation (FRS-BDR)* method for robustly estimating an underlying BD structure, given an outlier-corrupted affinity matrix. FRS-BDR method is built upon the definition of a vector \mathbf{v} that has been given in Section 3.2.4 to represent the BD affinity matrix as a piece-wise linear function. Compared to existing popular BDR approaches, such as, [LFL18, XGL17, FLX14], the optimization is efficiently performed in vector space instead of matrix space. Additionally and in contrast to EBDR (for details, see Section 3.4.1) the method is robust against outliers. Our main contributions are summarized as follows:

1. A comprehensive robustness analysis that quantifies the effects of outliers. In particular, our theoretical analysis shows how the vector \mathbf{v} and the eigenvalues, which carry substantial information about the BD structure, are influenced by outliers.
2. Our analysis enabled the development of a BDR algorithm that is (i) robust against outliers, since it builds upon our robustness analysis and (ii) computationally efficient by re-formulating the problem as a piece-wise linear function optimization problem instead of a matrix-optimization problem. We show that our proposed method even provides mathematically interpretable results in challenging settings where deriving eigenvalue

information is no longer possible (i.e., in the extreme case when all blocks are connected because of corruption by outliers).

In the following section, problem statement and main ideas of the proposed FRS-BDR method are explained. FRS-BDR algorithm is detailed in Section 4.4.1.2.3. Then, a summary of computational analysis of FRS-BDR is introduced in Section 4.4.1.2.4 and experimental evaluations demonstrating the performance of FRS-BDR in comparison to popular BDR approaches are shown in Section 4.4.1.2.5. Finally, conclusions are drawn in Section 4.4.1.2.6.

4.4.1.2.2 PROBLEM STATEMENT AND MAIN IDEAS

Problem Statement

Let a given data set of feature vectors $\mathbf{X} \in \mathbb{R}^{M \times N}$ be represented as a weighted graph $G = \{V, E, \mathbf{W}\}$, i.e., $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and $\|\mathbf{x}_m\| = 1, m = 1, \dots, N$. Further, let \mathbf{D} and $\mathbf{L} \in \mathbb{R}^{N \times N}$ denote, respectively, the overall edge weight and the Laplacian matrices associated with \mathbf{W} . Then, the goal of this work is to robustly estimate a K block zero-diagonal symmetric affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ using the available information about the vector \mathbf{v} and an eigen-decomposition. The number of blocks K is assumed to be unknown and \mathbf{X} may be subject to heavy-tailed noise and outliers which results in undesired outliers effects, such as, group similarity. Computational efficiency is also of fundamental interest. *Thus, in brief, the overall aim is to develop a fast and sparsity aware BDR method that is robust against outliers and group similarity.*

Main Ideas and Outline of FRS-BDR Method

This section summarizes the main ideas of our proposed FRS-BDR method. The full details are of each step are given in Section 4.4.1.2.3 and a comprehensive visual summary is provided in Appendix B.2.1.

To provide a general understanding, a high-level flow diagram illustrating the key steps of FRS-BDR is provided in Figure 4.34. As shown in the figure, the method consists of two general steps, i.e., computing vector \mathbf{v} (Step 1) and estimating vector \mathbf{v} (Step 2). The computation step starts with a given Type I outlier-corrupted and non-sparse Laplacian matrix \mathbf{L} (Step 1.0: Initialization in Figure 4.34). According to the explicit Definition 4.1.1 of Type I outliers, the method first removes the similarity coefficients associated to Type I outliers, which are represented in red color, from \mathbf{L} (Step 1.1: Type I Outlier Removal in Figure 4.34). Then, the next step is to structure the resulting matrix $\check{\mathbf{L}}$ in a BD form $\check{\mathbf{L}}$ based on a similarity-based BD ordering that we present in the sequel (Step 1.2: Similarity-based Block Diagonal Ordering in Figure 4.34). The last part of Step 1 is, to

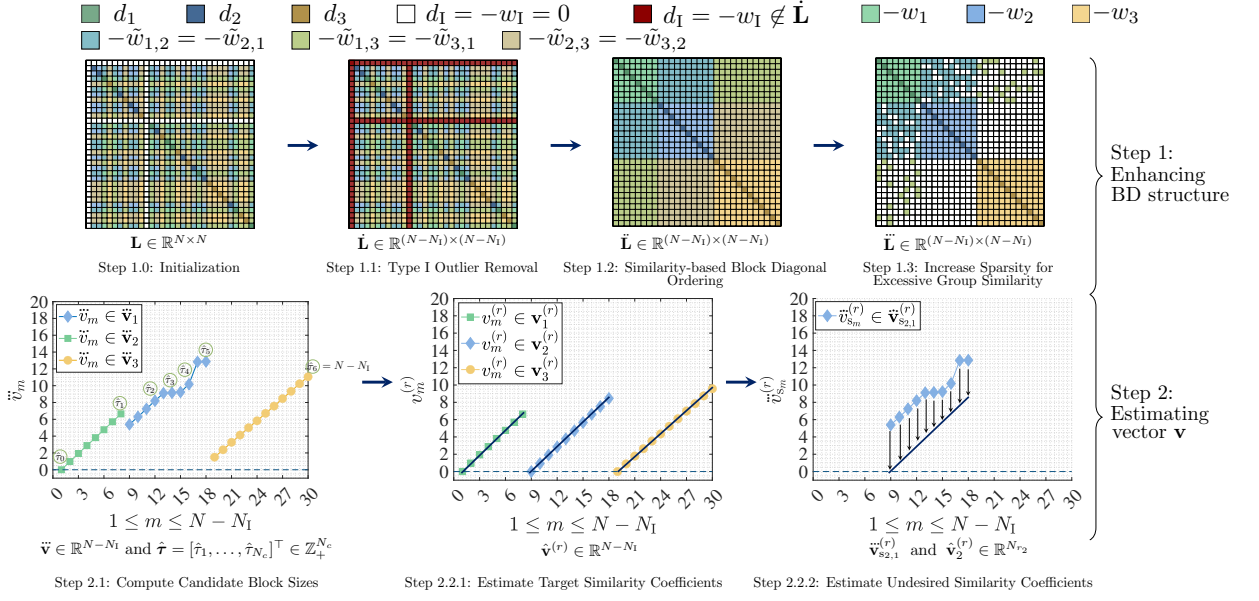


Figure 4.34: High-level flow diagram illustrating the key steps of FRS-BDR using a generic example with $K = 3$ clusters.

obtain vector \mathbf{v} in form of K discrete linear segments by computing an ordered sparse Laplacian matrix $\ddot{\mathbf{L}}$ (Step 1.3: Sparsity for Excessive Group Similarity in Figure 4.34). Then, the estimation step starts with a changepoint detection that we propose to compute the possible block sizes (Step 2.1: Compute Candidate Block Sizes in Figure 4.34). For each possible block size vector, i.e., $\mathbf{n}_r = [8, 10, 12]^T \in \mathbb{Z}_+^K$ in this illustrating example, the method computes a target vector $\mathbf{v}^{(r)}$ and corresponding estimate $\hat{\mathbf{v}}^{(r)}$ as a function of the target similarity coefficients (Step 2.2.1: Estimate Target Similarity Coefficients in Figure 4.34). Further, for every undesired similarity coefficient around which the blocks are concentrated, the shifted vectors (see Corollary 7.1) are computed separately and the undesired similarity coefficients are estimated (Step 2.2.2: Estimate Undesired Similarity Coefficients in Figure 4.34). Finally, the estimate $\hat{\mathbf{v}} \in \mathbb{R}^{N-N_1}$ is computed for the block size vector which provides the best fit to the computed vector $\tilde{\mathbf{v}}$.

4.4.1.2.3 FRS-BDR ALGORITHM

Step 1: Enhancing BD Structure

The key requirement for computing vector \mathbf{v} based on Eq. (3.2) is recovering an approximately BD structured Laplacian matrix. Assuming that \mathbf{W} (and the associated \mathbf{L}) are symmetric and sparse matrices, they can be ordered in a BD form [CM69] based on which vector \mathbf{v} can be directly computed. However, in general, similarity measures may not produce sparse affinity matrices. We therefore discuss the most challenging scenario, i.e., that \mathbf{W} is subject to Type I

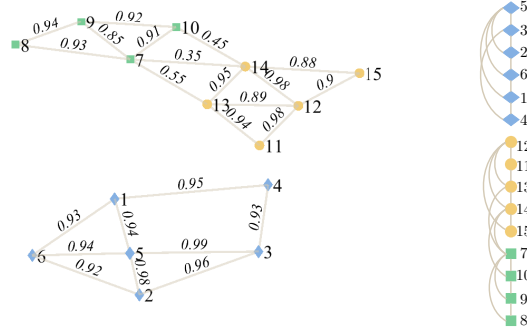


Figure 4.35: Exemplary plot of the sBDO algorithm.

outliers and all blocks exhibit similarity. Considering the Type I outliers' effect on the target vector \mathbf{v} (see, Section 4.2.5), the proposed vector \mathbf{v} computation starts with Type I outlier detection (Section 4.4.1.2.3). Then, a new BD ordering based on the similarity coefficients is proposed to generate a BD ordered Laplacian matrix (Section 4.4.1.2.3). Lastly, a sparse Laplacian matrix design is detailed for the case of excessive group similarity (Section 4.4.1.2.3).

Step 1.1 : Type I Outlier Removal

Type I outliers are detected according to

$$\mathbf{x}_m \in \mathbf{O}_I \text{ if } \forall w_{m,n} = 0 \text{ for } n = 1, \dots, N \text{ and } m \neq n, \quad (4.17)$$

where $\mathbf{O}_I \in \mathbb{R}^{M \times N_I}$ denotes the matrix of Type I outliers, $\mathbf{x}_m \in \mathbb{R}^M$ is the m th feature vector for $m = 1, \dots, N$, $w_{m,n}$ is the m, n th similarity coefficient corresponding to \mathbf{x}_m (due to the symmetry of \mathbf{W} , $w_{m,n} = w_{n,m}$).

Step 1.2 : Similarity-based BD Ordering (sBDO)

Let $\dot{\mathbf{X}} \in \mathbb{R}^{M \times (N-N_I)}$, $\dot{\mathbf{W}}$, $\dot{\mathbf{D}}$ and $\dot{\mathbf{L}} \in \mathbb{R}^{(N-N_I) \times (N-N_I)}$ be the resulting matrices after Step 1.1. The vector of the BD order, i.e., $\hat{\mathbf{b}} \in \mathbb{Z}_+^{N-N_I}$ is determined based on the following steps.

Step 1.2.1: Initialization: The BD order vector $\hat{\mathbf{b}}^{(1)}$ is comprised of the vertex index of maximum overall edge weight (i.e., \dot{d}_{\max}).

Step 1.2.2: Adding the most similar neighbor to $\hat{\mathbf{b}}^{(s)}$: Let $\hat{\mathbf{b}}^{(s)} = [\hat{b}_1, \dots, \hat{b}_{s-1}]^\top \in \mathbb{Z}_+^{s-1}$, with $s = 2, \dots, N - N_I$, denote the BD order vector at the s th stage. Assuming that the neighbors set is

Algorithm 5: sBDO

Input: $\dot{\mathbf{W}}, \dot{\mathbf{D}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$

Initialization:

Find the vertex of maximum overall edge weight \hat{d}_{\max}

for $s = 2, \dots, (N - N_1)$ **do**

Adding the most similar neighbor to $\hat{\mathbf{b}}^{(s)}$:

if *at least one neighbor exists* **then**

 Estimate \hat{b}_s using Eq. (4.18) and stack into $\hat{\mathbf{b}}^{(s)}$

else

 Find the vertex with maximum overall edge weight among unselected vertices and stack \hat{b}_s into $\hat{\mathbf{b}}^{(m)}$

end

end

Output: Estimated order vector $\hat{\mathbf{b}}^{(s)} \in \mathbb{Z}_+^{(N-N_1)}$

non-empty², the most similar neighbor to $\hat{\mathbf{b}}^{(s)}$ at the s th stage is determined by

$$\hat{b}_s = \arg \max_{m \in \mathbb{Z}_+^{N-N_1}} \left\{ \sum_{n=1}^{s-1} \dot{w}_{m, \hat{b}_n} \right\}, \quad (4.18)$$

where $m \in \mathbb{Z}_+^{N-N_1}$ denotes a neighbor vertex.

An example of the sBDO algorithm is illustrated in Figure 4.35 and technically summarized in Algorithm 5. As can be seen from Figure 4.35, starting from vertex five, whose overall edge weight is largest valued, the method selects the neighbors based on their edge weights that represent the similarity to previously selected vertices. After selecting all neighbors, the method jumps to the vertex that has the maximum overall edge weight among the remaining vertices and determines the ordering of the associated neighbors.

Step 1.3: Increase Sparsity for Excessive Group Similarity

Let $\ddot{\mathbf{W}}, \ddot{\mathbf{D}}$ and $\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$ be the matrices resulting from Step 1.2. A sparsity improved Laplacian matrix $\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$ is designed³ by increasing sparsity as long as, at least, the two smallest eigenvalues are close to zero⁴.

²If it is empty the method simply stacks the vertex index of maximum overall edge weight into $\hat{\mathbf{b}}^{(s)}$.

³For the exemplary sparse Laplacian matrix design algorithms, see Appendix B.2.4.

⁴For the definition of *close to zero*, see Appendix B.2.2.

Step 2: Estimating Vector \mathbf{v}

After computing $\ddot{\mathbf{L}}$, the vector $\ddot{\mathbf{v}} \in \mathbb{R}^{N-N_i}$ is obtained using Eq. (3.2)⁵. Then, this step models $\ddot{\mathbf{v}}$ as a K -piece linear function of similarity coefficients around which the blocks are assumed to be concentrated (for details, see Corollary 7.1.), i.e.,

$$\ddot{\mathbf{v}}_i = \mathbf{v}_i + 1 \sum_{\substack{j=1 \\ j \neq i}}^{i-1} N_j \tilde{w}_{i,j}, \quad i = 1, \dots, K, \quad (4.19)$$

where

$$\mathbf{v}_i = [0, w_i, \dots, (N_i - 1)w_i]^\top \in \mathbb{R}^{N_i} \quad (4.20)$$

denotes the i th linear segment of the target vector \mathbf{v}_i , w_i is the similarity coefficient around which the block i is concentrated and $\tilde{w}_{i,j}$ is the undesired similarity coefficient between blocks i and j around which they are concentrated, $\mathbf{1} \in \mathbb{R}^{N_i}$ is the column vector of ones, N_i and N_j are, respectively, the size of block i and j .

Step 2.1: Computing Candidate Block Sizes

Let $N_c \in \mathbb{Z}_+$ denote the number of changepoints, let $\boldsymbol{\tau} = [\tau_1, \tau_2, \dots, \tau_{N_c}]^\top \in \mathbb{Z}_+^{N_c}$ be the vector containing corresponding locations in $\ddot{\mathbf{v}}$, and let $\tau_0 = 0$ and $\tau_{N_c+1} = N$. Then, to estimate the model for vector $\ddot{\mathbf{v}}$ based on Eq. (4.19), our first step is to detect the the changepoints in $\ddot{\mathbf{v}}$ as in [KFE12]

$$\sum_{i=1}^{N_c+1} \sum_{m=\tau_{i-1}+1}^{\tau_i} (\ddot{v}_m - \hat{v}_m)^2 + \gamma N_c, \quad (4.21)$$

where γ denotes the penalty parameter⁶, \ddot{v}_m and \hat{v}_m are, respectively, the m th point in the i th linear segment of $\ddot{\mathbf{v}}$ and the corresponding least-squares linear fit. Then, for a candidate number of blocks from a given vector, i.e., $K_{\text{cand}} \in [K_{\text{min}}, \dots, K_{\text{max}}]^\top \in \mathbb{Z}_+^{N_K}$, the associated block-size matrix, i.e.,

$$\mathbf{N}^{(K_{\text{cand}})} = [\mathbf{n}_1, \mathbf{n}_2, \dots, \mathbf{n}_{K_{\text{cand}}}]^\top \in \mathbb{Z}_+^{K_{\text{cand}} \times K_{\text{cand}}}, \quad (4.22)$$

⁵The vector \mathbf{v} can alternatively be computed using $\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_i) \times (N-N_i)}$ by following Steps 1.1 and 1.2 if, at least, the two smallest eigenvalues of $\ddot{\mathbf{L}}$ are close to zero.

⁶To determine γ , its value is increased gradually as long as the function finds a lower number of changepoints than a predefined maximum number of changepoints $N_{c_{\text{max}}} \in \mathbb{Z}_+$ which is a reasonably small number satisfying $K - 1 \leq N_{c_{\text{max}}}$.

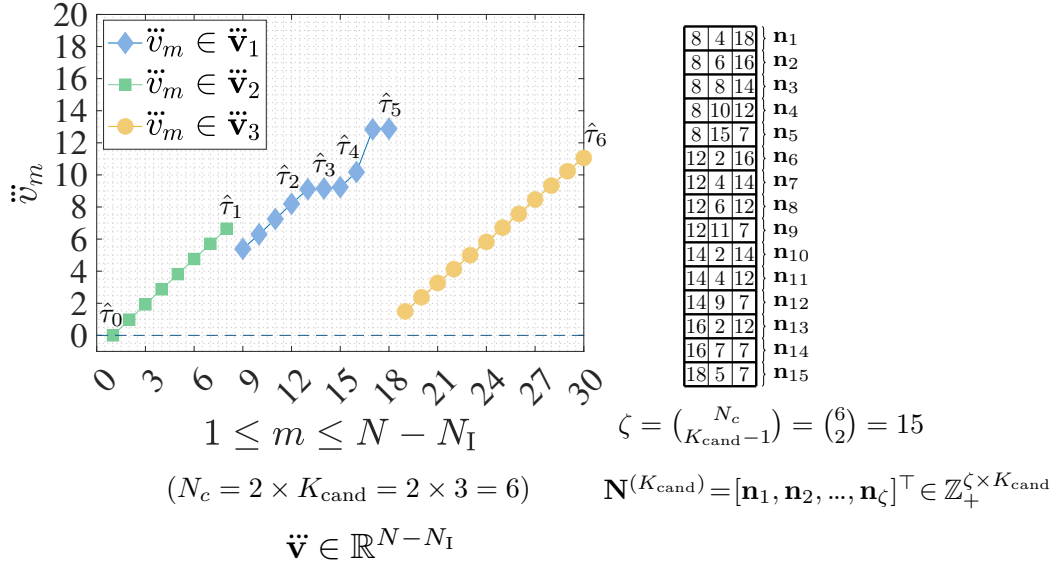


Figure 4.36: Exemplary plot of candidate block sizes.

is formed. The rows in Eq. (4.22), i.e., $\mathbf{n}_r = [N_{r_1}, N_{r_2}, \dots, N_{r_{K_{\text{cand}}}}]^\top \in \mathbb{Z}_+^{K_{\text{cand}}}$, $r = 1, \dots, \zeta$ denote the candidate size vectors that are designed by combination of all possible size vectors with $\zeta = \binom{N_c}{K_{\text{cand}}-1}$.⁷

The computation of candidate block sizes illustrated in Figure 4.36 for a candidate block number $K_{\text{cand}} = 3$. After estimating the changepoints using Eq. (4.21), a possible block size matrix, i.e. $\mathbf{N}^{(K_{\text{cand}})} \in \mathbb{Z}_+^{\zeta \times K_{\text{cand}}}$, with $\zeta = 15$ is computed for all possible block size combinations.

Step 2.2: Estimating Matrix of Similarity Coefficients

Step 2.2.1: Estimate Target Similarity Coefficients

Suppose that N_{r_i} denotes the size of i th linear segment from a candidate size vector \mathbf{n}_r , as defined in Eq. (4.22). Further, let $\mathbf{v}^{(r)} \in \mathbb{R}^{(N-N_I)}$ denote the target vector \mathbf{v} associated with \mathbf{n}_r defined by

$$v_m^{(r)} = \sum_{n=m}^{u_{r_i}} \ddot{l}_{m,n} \quad \text{s.t.} \quad \begin{matrix} \ell_{r_i} \leq m \leq u_{r_i} \\ i = 1, \dots, K_{\text{cand}} \end{matrix}, \quad (4.23)$$

where the m th and m, n th components of $\mathbf{v}^{(r)}$ and $\ddot{\mathbf{L}}$ are denoted, respectively, by $v_m^{(r)}$ and $\ddot{l}_{m,n}$, $\ell_{r_1} = 1, u_{r_1} = N_{r_1}$, $\ell_{r_i} = \sum_{k=1}^{i-1} N_{r_k} + 1$ and $u_{r_i} = \sum_{k=1}^i N_{r_k}$ for $i = 2, \dots, K_{\text{cand}}$.

⁷In practice, the candidate size vectors including the block sizes that are smaller than a predefined minimum number of vertices in the blocks N_{min} can be removed from $\mathbf{N}^{(K_{\text{cand}})}$.

After computing $\mathbf{v}^{(r)}$ using Eq. (4.23), with Theorem 3, we model it as a K -piece linear function of the target similarity coefficients. The model parameters are estimated in the FRS-BDR algorithm by applying the algorithm from [YYZ19] that determines a plane-based piece-wise linear fit. In more details, for every linear segment $i = 1, \dots, K_{\text{cand}}$ associated with K_{cand} , the method first estimates the parameters of the linear fit. Then, it estimates the target similarity coefficients $w_1, \dots, w_{K_{\text{cand}}}$ based on the slope of piece-wise linear fit estimate. A step-by-step detailed description of the plane-based piece-wise linear fit algorithm has been given in Section 3.4.1.3.1.

Step 2.2.2: Estimate Undesired Similarity Coefficients

In this step, the shifted vectors of $\mathbf{v}^{(r)}$ are computed as follows

$$\ddot{\mathbf{v}}_{s_{i,j}}^{(r)} = \mathbf{v}_i^{(r)} + \ddot{\mathbf{v}}_{i,j}^{(r)}, \quad \begin{array}{l} i = 2, \dots, K_{\text{cand}}, \\ j = 1, \dots, i - 1 \end{array} \quad (4.24)$$

where

$$\ddot{\mathbf{v}}_{i,j}^{(r)} = \left[\sum_{n=\ell_{r_j}}^{u_{r_j}} \ddot{l}_{\ell_{r_j},n}, \dots, - \sum_{n=\ell_{r_j}}^{u_{r_j}} \ddot{l}_{u_{r_j},n} \right]. \quad (4.25)$$

denotes the vector of increase associated with the undesired group similarity between block i and j , and $\ddot{\mathbf{v}}_{s_{i,j}}^{(r)}$ is the associated shifted target vector. Then, combining the results from Eq. (4.19), Eq. (4.23) and Eq. (4.24), the undesired similarity coefficients between different blocks can be estimated as

$$\hat{w}_{i,j}^{(r)} = \frac{\text{med}(\ddot{\mathbf{v}}_{s_{i,j}}^{(r)} - \hat{\mathbf{v}}_i^{(r)})}{N_{r_j}} \quad \begin{array}{l} i = 2, \dots, K_{\text{cand}} \\ j = 1, \dots, i - 1 \end{array}, \quad (4.26)$$

where $\text{med}(\cdot)$ denotes the median operator, N_{r_j} is defined in Eq. (4.22), and $\hat{w}_{i,j}^{(r)}$ is the undesired similarity coefficient estimate between i and j .

To clarify Steps 2.2.1 and 2.2.2, an example with $K_{\text{cand}} = K$ illustrating the computation of vector $\ddot{\mathbf{v}}$ and a matrix $\mathbf{W}_{\text{sim}} \in \mathbb{R}^{K \times K}$ is shown in Fig 4.37. As can be seen, the target similarity coefficients, which are the diagonal elements of \mathbf{W}_{sim} , i.e., $\text{diag}(\mathbf{W}_{\text{sim}}) = [w_1, w_2, \dots, w_K]^\top \in \mathbb{R}^K$, represent an estimate of the slopes of the $K_{\text{cand}} = K$ linear segments in $\ddot{\mathbf{v}}$. Further, off-diagonal elements of \mathbf{W}_{sim} represent undesired similarity coefficients between different blocks and are calculated by computing the undesired shifts that have been highlighted as shaded areas in Fig 4.37.

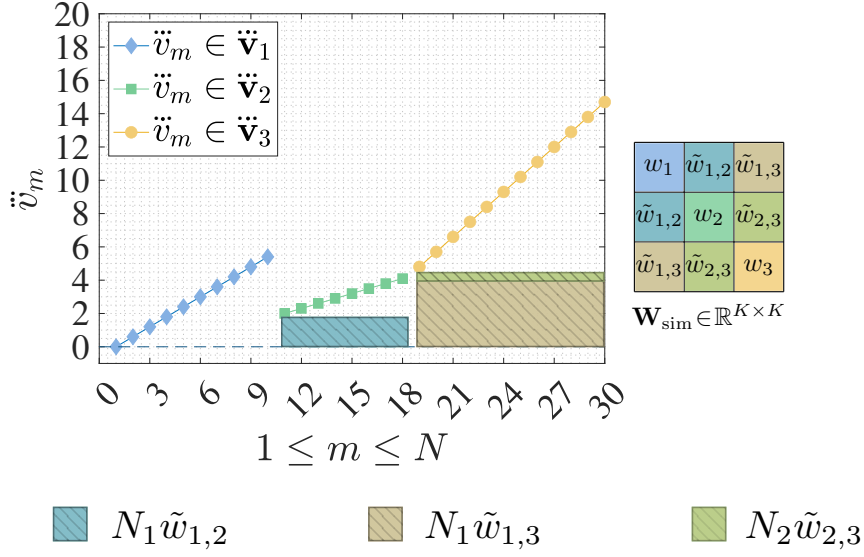


Figure 4.37: Exemplary illustration of $\ddot{\mathbf{v}}$ and \mathbf{W}_{sim} with $K_{\text{cand}} = K$, $\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $\text{diag}(\mathbf{W}_{\text{sim}}) = [0.6, 0.3, 0.9]^\top \in \mathbb{R}^K$, $\tilde{w}_{1,2} = 0.2$, $\tilde{w}_{1,3} = 0.4$, and $\tilde{w}_{2,3} = 0.1$.

Step 2.3: Estimating vector $\ddot{\mathbf{v}}$ and \mathbf{W}_{sim}

From the computed estimates $\hat{\mathbf{W}}_{\text{sim}}^{(r)} \in \mathbb{R}^{K_{\text{cand}} \times K_{\text{cand}}}$ and $\hat{\mathbf{v}}^{(r)} \in \mathbb{R}^{(N-N_i)}$, the vector $\hat{\mathbf{v}}_i^{(r)}$ is computed by plugging in the associated intermediate estimates for all $r = 1, \dots, \zeta$ and $K_{\text{cand}} = K_{\text{min}}, \dots, K_{\text{max}}$ into Eq. (4.19) and determining the final estimate as

$$\hat{\mathbf{v}} = \underset{\mathbf{n}_r \in \mathbf{N}^{(K_{\text{cand}})}}{\text{argmax}} \|\ddot{\mathbf{v}} - \hat{\mathbf{v}}^{(r)}\|_2 \quad (4.27)$$

where $\forall \hat{w}_i^{(r)} \in \text{diag}(\hat{\mathbf{W}}_{\text{sim}}^{(r)})$, $\hat{w}_i^{(r)} > \hat{w}_{i,j}^{(r)}$ holds for $i = 1, \dots, K_{\text{cand}}$, $j = 1, \dots, K_{\text{cand}}$ and $i \neq j$.

The proposed FRS-BDR is summarized in Algorithm 6.

4.4.1.2.4 COMPUTATIONAL ANALYSIS OF FRS-BDR

A comprehensive computational analysis is computed in Appendix B.2.3 by determining the number of fladd, flmlt, fldiv and flam. (For a detailed information, see [Ste01, Ste98]). The Landau's big O symbol is used for the cases when the complexity is not specified as above operations. Our analysis showed that the complexity of FRS-BDR strongly depends on the initial structure of the affinity matrix and the number of blocks K . In addition to the numeric analysis, the complexity is analyzed experimentally in the following sections.

Algorithm 6: FRS-BDR

Input: $\mathbf{X} \in \mathbb{R}^{M \times N}$, K_{\min} , K_{\max} , $N_{c_{\max}}$, N_{\min} (opt.)
 Compute $\mathbf{W} \in \mathbb{R}^{N \times N}$ i.e. $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ for $\forall \mathbf{x}_m \in \mathbf{X}$, $\|\mathbf{x}_m\|=1$
Step 1: Computing Vector \mathbf{v}
Step 1.1: Type I Outlier Removal
 Compute $\hat{\mathbf{W}}$, $\hat{\mathbf{D}}$ and $\hat{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$ via Eq. (4.17)
Step 1.2: Similarity-based Block Diagonal Ordering
 Perform Algorithm 5 to achieve $\hat{\mathbf{b}}^{(s)} \in \mathbb{Z}_+^{(N-N_1)}$
 Obtain $\hat{\mathbf{W}}$, $\hat{\mathbf{D}}$ and $\hat{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$ using $\hat{\mathbf{b}}^{(s)}$
Step 1.3 (opt.): Sparsity for Excessive Group Similarity
 Design $\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$ for the desired method,
 i.e. Algorithm 3 or 4 in Appendix B.2.4
 Compute $\ddot{\mathbf{v}} \in \mathbb{R}^{(N-N_1) \times 1}$ corresponds to $\ddot{\mathbf{L}}$ using Eq. (3.2)
 (or alternatively $\ddot{\mathbf{v}} \in \mathbb{R}^{(N-N_1) \times 1}$ corresponds to $\hat{\mathbf{L}}$)
Step 2: Estimating Vector \mathbf{v}
for $K_{\text{cand}} = K_{\min}, \dots, K_{\max}$ **do**
Step 2.1: Computing Candidate Block Sizes
 Compute $\mathbf{N}^{(K_{\text{cand}})} \in \mathbb{Z}_+^{K_{\text{cand}} \times K_{\text{cand}}}$ using Eqs. (4.21)-(4.22)
Step 2.2: Estimating \mathbf{W}_{sim}
for $\mathbf{n}_r = \mathbf{n}_1, \dots, \mathbf{n}_\zeta$ **do**
Step 2.2.1: Estimating Target Similarity Coefficients
 Compute $\mathbf{v}^{(r)} \in \mathbb{R}^{(N-N_1)}$ using Eq. (4.23)
for $i = 1, \dots, K_{\text{cand}}$ **do**
 Calculate $\boldsymbol{\Sigma}_i^{(r)} \in \mathbb{R}^{2 \times 2}$ and $\boldsymbol{\mu}_i^{(r)} \in \mathbb{R}^2$ for $\Upsilon_i^{(r)}$
 Find $\hat{\boldsymbol{\vartheta}}_i^{(r)} \in \mathbb{R}^2$ and $\hat{b}_i^{(r)} \in \mathbb{R}$
 Find $\hat{\mathbf{v}}_i^{(r)} \in \mathbb{R}^{N_{r_i}}$ and compute \hat{w}_i
end
 Form $\text{diag}(\hat{\mathbf{W}}_{\text{sim}}^{(r)}) = [\hat{w}_1^{(r)}, \hat{w}_2^{(r)}, \dots, \hat{w}_{K_{\text{cand}}}^{(r)}]^\top \in \mathbb{R}^{K_{\text{cand}}}$
 and $\hat{\mathbf{v}}^{(r)} = [(\hat{\mathbf{v}}_1^{(r)})^\top, (\hat{\mathbf{v}}_2^{(r)})^\top, \dots, (\hat{\mathbf{v}}_{K_{\text{cand}}}^{(r)})^\top]^\top \in \mathbb{R}^{(N-N_1)}$
Step 2.2.2: Estimating Undesired Similarity Coefficients
for $i = 2, \dots, K_{\text{cand}}$ **do**
for $j = 1, \dots, i-1$ **do**
 Compute $\ddot{\mathbf{v}}_{s_{i,j}}^{(r)} \in \mathbb{R}^{(N-N_1)}$ using Eqs. (4.24)
 Compute $\hat{w}_{i,j}^{(r)}$ using Eq. (4.26) and stack $\hat{\mathbf{W}}_{\text{sim}}^{(r)}$
end
end
 Estimate $\ddot{\mathbf{v}}^{(r)}$ using Eq. (4.19)
 Update $\hat{\mathbf{v}}$ based on Eq. (4.27)
end
end
Output: $\hat{\mathbf{v}}, \hat{\mathbf{W}}_{\text{sim}}, \hat{\mathbf{n}}$

4.4.1.2.5 EXPERIMENTAL EVALUATION

This section benchmarks the proposed FRS-BDR method in a broad range of real data experiments including cluster enumeration and handwritten digit, object and face clustering.

Data Sets: The performance is analyzed using the well-known MNIST [HS98] and USPS [Hul94] data sets for handwritten digit clustering, COIL20 [NNM95] for object clustering and ORL [SH94], JAFFE [LAK98] and Yale [BHK97] for face clustering. For cluster enumeration, the methods are benchmarked on the Breast Cancer Wisconsin (Breast Cancer) [WM89], Chemical Composition of Ceramic (Ceramic) [HZZ16], Vertebral Column [RSB11], Fisheriris [Fis36], Gait [SAZ19], O. Cancer [CFR04], Person Id. [TSM18] and Parkinson A. [NPC16] data sets.

Baselines: For the task of subspace clustering, FRS-BDR is benchmarked against seven state-of-the-art BDR approaches, i.e. BDSSC [FLX14], BDLRR [FLX14], BDR-B [LFL18], BDR-Z [LFL18], IBDLR [XGL17], EBDR [TMZ22], LSR [LMZ12], two low-rank representation methods LRR [LLY12], RKLRR [XTX15], a sparse representation method SSC [EV13] and the initial affinity matrix that is defined by $\mathbf{W}^{N-1} = \mathbf{X}^\top \mathbf{X}$. For cluster enumeration⁸, the method is benchmarked against seven popular community detection methods including Louvain [BGL08], Martelot [MH11], BNMF [PRE11], DenPeak [BYS17], Combo [SCB14], MAP [BEL14] and SPARCODE [TMZ21].

Parameter Setting: In all experiments, the parameters are optimally tuned for the competitor approaches, while FRS-BDR is computed with the default parameters that are detailed in Appendix B.2.5.1.

Evaluation Metrics: The computation time (t) and average clustering accuracy \bar{p}_{acc} are used for the subspace clustering performance analysis. In cluster enumeration, the empirical probability of detection (p_{det}), modularity (mod) and conductance (cond) are used in addition to t .

Handwritten Digit Clustering

The effectiveness of FRS-BDR in handwritten digit clustering is shown based on the following popular real-world data sets:

MNIST Data Set: The data base includes 60000 training and 10000 test images correspond to 10 digits. For different number of subjects $K = \{2, 3, 5, 8, 10\}$, the data matrix \mathbf{X} is generated by using 100 randomly selected images from the test set for every subject where the images are used as feature vectors and normalized. As in [LFL18], \mathbf{X} of size $784 \times 100K$ is produced for the images

⁸For the numerical cluster enumeration results, see Appendix B.2.5.2.8

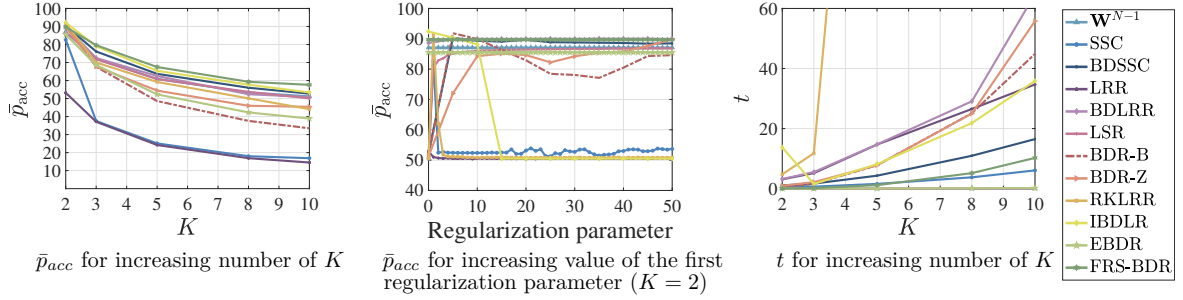


Figure 4.38: Numerical results for MNIST data set. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.

of size 28×28 .

USPS Data Set: 7291 training and 2007 test images of size 16×16 are contained in the data set. The data matrix \mathbf{X} is computed by following the same procedure, except for using 50 randomly selected images from the test set for every subject. As a result, for the images of size 16×16 , the data matrix \mathbf{X} of size $256 \times 50K$ corresponding to $K = \{2, 3, 5, 8, 10\}$ number of subjects, is obtained.

In contrast to object and face applications that will be detailed in the following steps, the data matrix \mathbf{X} of high dimensional feature vectors is directly used in affinity matrix design. Then, SC (For the details about SC, see Section 2.5.1.) is applied to the resulting affinity matrices of different methods.

An example of digit clustering results is shown in Figure 4.38 for the MNIST data base. (A broad set of analysis including MNIST and USPS data bases is provided in Appendix B.2.5.2.) Even though the performance of SSC [EV13], BDSSC [FLX14], BDLRR [FLX14][FLX14], BDR-B [LFL18], BDR-Z [LFL18], IBDLR [XGL17], LSR [LMZ12], LRR [LLY12] and RKLRR [XTX15] is reported for an optimal tuning of the parameters, which is not feasible in practice, the FRS-BDR achieves the highest clustering accuracy results in almost all cases. Further, the regularization parameter effect analysis in Figure 4.38 showed that BDR-B and BDR-Z performances are sensitive to the choice of the first regularization parameter, even when tuning the second one optimally. Based on the computation time analysis, the main drawback of competitor approaches is that they are sensitive to the dimension of feature space whereas FRS-BDR is an efficient algorithm for the data sets including high dimensional feature vectors.

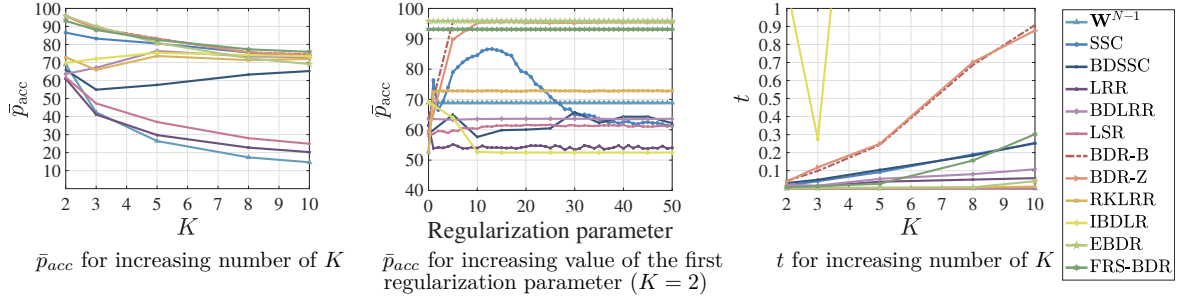


Figure 4.39: Numerical results for COIL20 data set. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.

Object Clustering

This section introduces a set of experiments that are performed on COIL20 [NNM95] data base of 20 objects. In COIL20, each object has 72 images where images are taken by rotating the object on a turntable for five degrees intervals. In our experiments, the processed COIL20 data set in [CHH10] containing images of size 32×32 pixels is used. Then, the data set \mathbf{X} of size 1024×400 is generated by selecting 20 images randomly for every object. The feature space is reduced to 10 based on PCA performance analysis which is provided in Appendix B.2.5.2.3.

As in [LFL18], performance analysis of every application is conducted for an increasing value of K , i.e., $K = \{2, 3, 5, 8, 10\}$ using 100 randomly selected subject combinations. To obtain the affinity matrices for the competing methods, the regularization parameters are manually tuned on a grid of 50 values. Finally, SC [Lux07] is applied and the results in Figure 4.39, for increasing values of blocks K , are obtained analogously to [LFL18] (see Appendix B.2.5.2.3 for all further details). The average clustering accuracy \bar{p}_{acc} results show that FRS-BDR performs best while EBDR is an efficient method for small values of K . In terms of t , the main competitors BDR-B and BDR-Z show poor performance whereas FRS-BDR performs relatively good even for large values of K .

BDR-B and BDR-Z methods show poor performance for small-valued regularization parameters even though the second regularization parameter is optimally tuned. An important point is that these approaches reach their best results lately in comparison to experiments on face clustering data sets that will be explained in the following section.

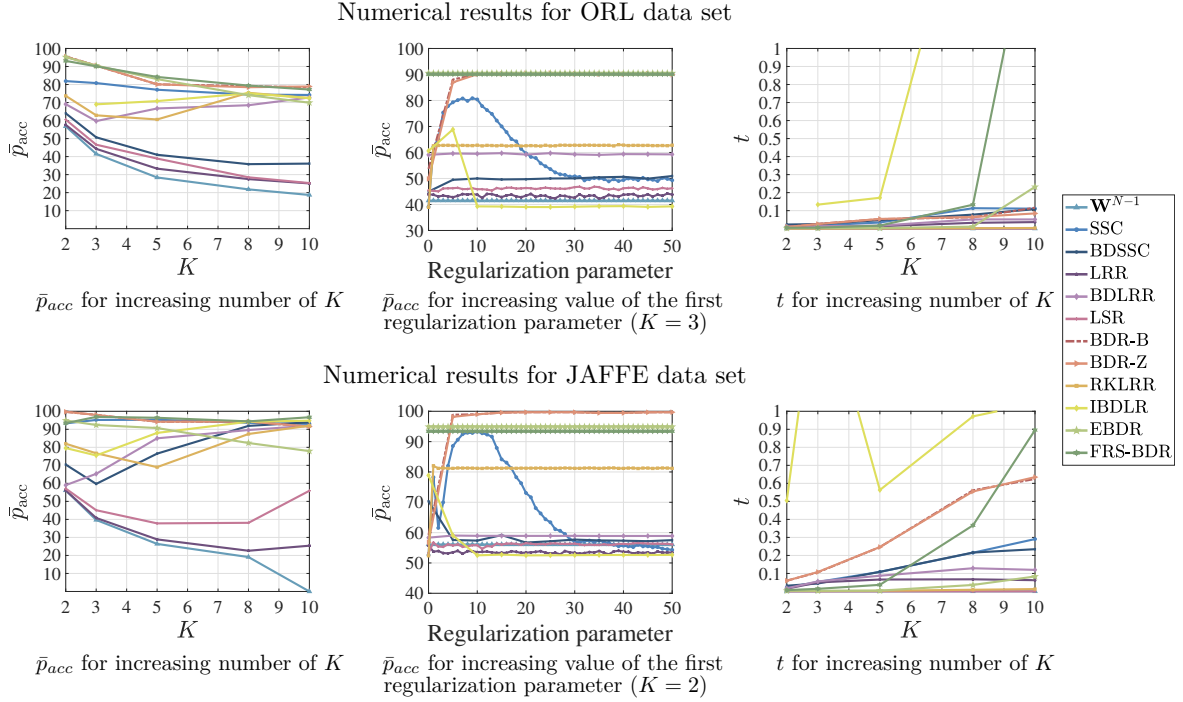


Figure 4.40: Numerical results for ORL and JAFFE data sets. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.

Face Clustering

In this section, the subspace clustering performances of different methods are benchmarked in terms of their \bar{p}_{acc} and t by using the following application details:

ORL Data Set: The data set includes 10 images of 40 different subjects that are taken at different times by varying the lighting, facial expressions and details. As in [XGL17], we resize all images to 32×32 to obtain a data matrix \mathbf{X} of size 1024×400 using normalized features. The feature space dimension is reduced to nine using PCA in order to reduce the computation time (For the PCA analysis of ORL data set, see Appendix B.2.5.2.4.).

JAFFE Data Set: 213 images of seven facial expressions from 10 Japanese female models comprise the JAFFE data set. As in [XGL17], the images are resized to 64×64 pixels and the data set \mathbf{X} of size 4096×213 is computed using resized images as normalized feature vectors before applying PCA to reduce the dimensionality to 14 features (For the PCA analysis of JAFFE data set, see Appendix B.2.5.2.5.).

Yale Data Set: 165 grayscale images of 15 different individuals. For every subject, the data set

Average Clustering Accuracy (\bar{p}_{acc}) for Different BDR Methods												
Data Set	\mathbf{W}^{N-1}	Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters										
		SSC	BDSSC	LRR	BD-LRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
Breast Cancer[WM89],	88.2	51.0-74.7	50.3-88.2	54.3-90.3	88.0-90.0	73.5-88.2	62.4-90.0	52.9-90.2	62.6-91.7	60.3-90.0	85.2	90.1
Ceramic [HZZ16],	98.9	51.1-98.9	51.1-100	95.5-98.9	95.5-98.9	54.5-98.9	51.1-100	51.1-98.9	51.1-95.5	51.1-98.9	98.9	98.9
Vertebral Column [RSB11],	73.2	50.0-77.7	50.3-74.8	53.9-72.6	72.6-72.6	62.6-75.8	67.4-76.8	71.9-76.8	67.4-71.3	67.4-76.1	74.8	75.8
Fisheriris [Fis36],	78.0	34.7-82.7	34.0-83.3	38.7-80.7	80.0-98.0	78.0-82.7	34.0-96.7	65.3-96.7	34.0-80.0	34.7-84.0	98.0	96.7
Gait [SAZ19],	77.3	20.3-77.4	20.1-77.5	26.1-83.9	78.9-83.5	55.4-75.9	20.3-84.8	26.4-84.5	20.5-85.5	20.4-81.6	81.1	77.1
O. Cancer [CFR04],	61.7	51.4-73.6	50.9-71.3	52.3-76.4	54.2-76.4	51.9-66.2	53.7-75.9	51.9-74.1	55.6-88.4	55.6-75.5	77.8	77.3
Person Id. [TSM18],	x	33.7-96.8	31.6-95.7	49.7-94.7	71.1-94.7	33.2-64.2	31.6-96.3	59.4-95.7	34.2-94.1	33.7-95.7	97.3	96.8
Parkinson A. [NPC16],	61.3	50.4-58.8	50.0-61.3	50.4-54.2	50.4-61.3	57.9-61.3	50.4-61.3	50.0-61.3	50.4-61.7	50.4-61.3	56.7	58.2
Average	76.9	42.8-80.1	42.3-81.5	52.6-81.4	73.8-84.4	58.4-76.6	46.4-85.2	53.6-84.8	47.0-83.5	46.7-82.9	83.7	83.9

Table 4.7: Subspace clustering performance of different BDR approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. ‘x’ denotes the failed results due to the complex-valued eigenvectors.

contains 11 images that capture different facial expressions. The data matrix \mathbf{X} of size 1024×165 is constructed as in the ORL Data Set (For the PCA analysis of Yale data set, see Appendix B.2.5.2.6.).

After determining the number of PCA features, the same procedure as in object clustering is performed and the performance is reported for a different number of subjects $K = \{2, 3, 5, 8, 10\}$ in Figure 4.40. (For a detailed performance analysis, see Appendix B.2.5.2.)

The average clustering accuracy \bar{p}_{acc} and computation time t for the ORL and JAFFE data sets are provided in Figure 4.40. Consistent with the previous experiments, FRS-BDR shows the best clustering accuracy performance among all approaches in almost all cases. In terms of t , FRS-BDR shows a reasonably good performance until number of subjects reaches $K = 8$. A reduction for large-valued K can be obtained by adjusting N_{cmax} . Extensive further numerical experiments are reported in Appendices B.2.5.2.4, B.2.5.2.5 and B.2.5.2.6.

Subspace Clustering on Well-Known Clustering Data Sets

This section investigates the subspace clustering performance of different approaches in terms of their average clustering accuracy using the following popular clustering data sets: Breast Cancer [WM89], Ceramic [HZZ16], Vertebral Column [RSB11], Fisheriris [Fis36], Gait [SAZ19], O. Cancer [CFR04], Person Id. [TSM18] and Parkinson A. [NPC16]. Starting from the initial affinity matrix that is defined by $\mathbf{W}^{N-1} = \mathbf{X}^\top \mathbf{X}$, the affinity matrices are estimated for different approaches. For competing methods, the affinity matrix construction methods are optimally tuned and FRS-BDR is computed with the default parameters. Then, SC as in is applied, as detailed in Section 2.5.1.

The clustering accuracy performances of different block-diagonal representation approaches are detailed in terms of their average clustering accuracy in Table 4.7. As can be seen from Table 4.7, FRS-BDR provides similar performance to the maximum clustering accuracy results

of its strongest competitors (BDR-B, BDR-Z, BD-LRR) while it outperforms all other BDR approaches. The method is also computationally efficient in comparison to most of the competitor methods based on the additional experiments that are given in Appendix B.2.5.2.7.

4.4.1.2.6 CONCLUSION

A robust method to recover a BD affinity matrix in challenging has been presented. The proposed Fast and Robust Sparsity-Aware Block Diagonal Representation (FRS-BDR) method jointly estimates cluster memberships and the number of blocks. It builds upon our presented theoretical results that describe the effect of different fundamental outlier types in cluster analysis, allowing to reformulate the problem as a robust piece-wise linear fitting problem. Comprehensive experiments including a variety of real-world applications demonstrate the effectiveness of FRS-BDR compared to optimally tuned benchmark methods in terms of clustering accuracy, computation time and cluster enumeration performance.

4.4.2 ROBUST EIGENVECTOR ESTIMATION METHODS

This section incorporate robustness in the embedding space based on the two different embedding strategies that are detailed in the following.

4.4.2.1 ROBUST REGULARIZED LOCALITY PRESERVING INDEXING FOR FIEDLER VECTOR ESTIMATION

4.4.2.1.1 INTRODUCTION

As discussed in Section 2.3.3, the Fiedler vector of a connected graph is the eigenvector associated with the second smallest eigenvalue, the so called Fiedler value, of the graph Laplacian matrix. The Fiedler vector and the Fiedler value provide important information for estimating [BM13, ASD12, Fie73] and controlling [TD20, LB14, YFG10] the algebraic connectivity of a graph, finding densely connected groups of vertices that are hidden in the graph structure [Sch07, ST96], and representing the implicit relationships between variables in a low-dimensional space [DS20, Hen07]. More generally, eigenvector decomposition is used in a variety of applications to achieve tasks, such as dimension reduction [CHZ07, CHH05, HCL04, HN04, DDF90], recognition [ZZL18, LT09, YTL06], clustering/classification [HRG18, OFK18, CC14, XG08, GKR05, NJW01] and localization [SVB01]. Due to its central role in graph analysis, the computation of the Fiedler vector has been a fundamental research area for decades [DS20, CHZ07, CHH05, HCL04, HN04, DDF90].

A popular method to embed data points into a low-dimensional space is Laplacian eigenmaps (LE) [BN01] method for which the embedding is performed based on the eigen-decomposition of Laplacian matrix. It can therefore also be used for Fiedler vector computation. The LE method performs a nonlinear dimensionality reduction while preserving the local neighborhood information in a certain sense, and it explicitly reveals the manifold structure [BN01]. An alternative approach is that of locality preserving indexing (LPI) [HCL04], which transforms the nonlinear dimensionality reduction in the Laplace Beltrami operator into a linear system of equations. LPI requires a complete singular value decomposition (SVD), resulting in a considerable computational complexity which is why computationally more attractive alternative approaches have been proposed in [HRG18, CHZ07]. However, the performance of [CHZ07] strongly depends on the penalty parameter selection. Further, in real-world scenarios, outliers and heavy-tailed noise may obscure the graph structure that represents the clean data. Consequently, the computed Fiedler vectors are corrupted, and embeddings based on these vectors no longer provide useful information about the majority of the data set, as they are dominated by outliers. Therefore, robust Fiedler vector estimation methods are needed. One popular strategy is to mitigate the effects of outliers in the *representation space* via a restructuring of the affinity matrix based on a prior information, e.g., the number of clusters [LNC18, FLX14] and the level of sparsity [ZCS19, AGR19, LNC18, ZZL18, FLX14] that plays a crucial role in the structure of eigenvectors. However, a major challenge is to determine this prior information, especially in the presence of outliers and heavy-tailed noise that may obscure the underlying structure. Alternatively, outliers can be suppressed in the *projection space*, such as in [TMZ21, YCL20, ZCS19, CNW15, PYT15]. However, most of these approaches, again, require prior knowledge, e.g., the label information of a data set [YCL20, CNW15] or data dependent parameter tuning to determine the descriptive features [ZCS19]. Moreover, the robust projection operation in [PYT15], uses the ℓ_1 norm that creates a different eigenbasis and requires prior information about the data, i.e. representative samples. The robust locality preserving feature mapping (RLPFM) approach in [TMZ21] preserves the ℓ_2 norm and builds upon M-estimation to suppress the outliers. However, it performs M-estimation of the eigenvectors by iteratively reweighting the residuals of LE-based prediction which results in a large computation cost.

To address these issues, we proposed a new Robust Regularized Locality Preserving Indexing (RRLPI) method for Fiedler vector estimation in [TMZ22]. Our key idea was to provide robustness in the *embedding space* by transforming the Fiedler vector estimation problem into a linear system of equations that reveals the hidden group structure in a given graph without

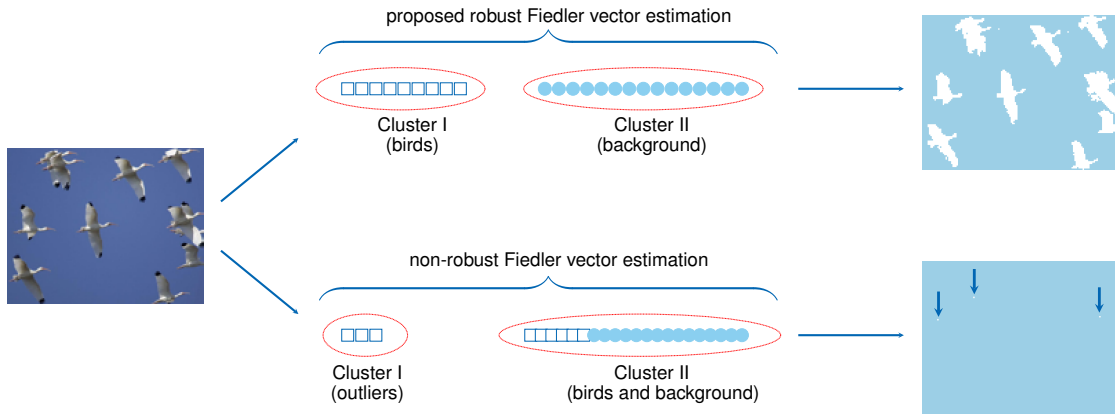


Figure 4.41: Exemplary image segmentation result comparing the popular LE [BN01] and the proposed RRLPI methods for Fiedler vector estimation.

assuming any prior knowledge. Motivated by the importance of the sparsity level in the Fiedler vector structure, we began by distinguishing fundamental outlier types and investigated how their occurrence depends on the determined sparsity level. This analysis of the effects of outliers on the eigen-decomposition enabled us to understand how to best integrate robustness into the Fiedler vector computation. Based on our analysis, we showed that the overall edge weight attached to a vertex is a valuable information to identify an outlier. Therefore, an error model was formulated based on the typical overall edge weight of a graph. Unlike other embedding approaches whose performance strongly depends on correctly setting parameters with the help of prior knowledge, e.g., [YCL20, ZCS19, CNW15, CHZ07], our penalty parameter determination was formulated as part of the optimization (similar to [TMZ21]) based on Δ -separated sets [ARV09] which are defined based on geometric analysis of well-spread ℓ_2^2 representations. However, in contrast to RLPFM [TMZ21], the proposed RRLPI robustly estimates the Fiedler vector based on the typical overall edge weight of the graph, which incorporates the weighting operation into a single step and makes the proposed method computationally efficient in comparison to [TMZ21] that has been discussed in Section 4.4.2.2.

An image segmentation application illustrating the need for robust Fiedler vector estimation is provided in Figure 4.41. Starting from an original image including birds and background (sky), the aim clearly is to assign birds and the background into different segments. To this end, the image is represented as graph and the Fiedler vectors are computed using RRLPI (top) and LE (bottom). The resulting Fiedler vectors are then clustered into two groups. As can be seen, the LE based Fiedler vector computation results in assigning outlying entries of the Fiedler vector as one small cluster while merging birds and background in a second big cluster. By contrast, the robustly

estimated Fiedler vector using the proposed RRLPI method provides the correct structure in the Fiedler vector estimate to enable the desired segmentation into birds and background. Robustness is obtained by assigning weights to all data-points based on the typical overall edge weight in the associated graph. Consistent with the ideas of M-estimation in robust statistics [ZKO18, ZKC12], a large degree of outlyingness of a data-point corresponds to a small weight.

This section is organized as follows. First, the basic concepts and Fiedler vector estimation using LPI are briefly discussed. Further, the ideas underlying the proposed algorithm and the problem formulation are given. Then, the proposed RRLPI method is detailed. After introducing the theoretical analysis, penalty parameter selection, computational complexity analysis and possible applications, conclusions are drawn. An implementation of RRLPI is available at:

<https://github.com/A-Tastan/RRLPI>

4.4.2.1.2 LPI FOR COMPUTING THE FIEDLER VECTOR

Suppose that a data set $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ with m denoting the data dimension and n being the number of data-points, can be represented as a graph $G = \{V, E, \mathbf{W}\}$, where V denotes the vertices, E represents the edges, and $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the symmetric affinity matrix that is computed using a similarity measure, e.g. cosine similarity as defined in Eq. (2.1). Let $\mathbf{L} \in \mathbb{R}^{N \times N}$ denote the nonnegative definite Laplacian matrix with associated eigenvalues $0 \leq \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}$ sorted in ascending order as defined in Section 2.3.2. Then, it follows that the Fiedler vector $\mathbf{y}_F \in \mathbb{R}^n$ is the eigenvector associated with the second smallest eigenvalue λ_1 of the eigen-problem Eq. (2.5) or in a generalized eigenvalue problem form Eq. (2.6). Here, the Laplacian matrix \mathbf{L} is defined analogously to the Laplace Beltrami operator on the manifold by $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} \in \mathbb{R}^{N \times N}$ is a diagonal weight matrix with overall edge weights $d_{m,m} = \sum_n w_{m,n}$ on the diagonal and $\mathbf{y}_m \in \mathbb{R}^N$ is the eigenvector associated with λ_m .

The LPI method determines linear approximations to the eigenfunctions of the Laplace Beltrami operator [HCL04, HN04] by representing the Fiedler vector as the response of a linear regression with input variables \mathbf{X} , i.e., $\mathbf{y}_F = \mathbf{X}^\top \boldsymbol{\beta}_F$. Hence, the LPI finds a transformation vector $\boldsymbol{\beta}_F \in \mathbb{R}^M$ that is the eigenvector associated with the second smallest eigenvalue of the generalized eigen-problem

$$\mathbf{X}\mathbf{L}\mathbf{X}^\top \boldsymbol{\beta}_m = \lambda_m \mathbf{X}\mathbf{D}\mathbf{X}^\top \boldsymbol{\beta}_m, \quad m = 0, \dots, N-1 \quad (4.28)$$

or, equivalently, it is associated with the second largest eigenvalue of the generalized eigen-problem

$$\mathbf{X}\mathbf{W}\mathbf{X}^\top \boldsymbol{\beta}_n = \lambda_n \mathbf{X}\mathbf{D}\mathbf{X}^\top \boldsymbol{\beta}_n, \quad (4.29)$$

which has the same eigenvalue λ_m as in Eq. (2.6) for λ_n with $n = N - (m + 1)$. In the following theorem, the link between the well-known Fiedler vector computation and LPI [HCL04] is shown.

Theorem 8. *Let \mathbf{y}_F be the Fiedler vector associated with the second largest eigenvalue λ_F such that $F = N - 2$ of the eigen-problem*

$$\mathbf{W}\mathbf{y}_F = \lambda_F \mathbf{D}\mathbf{y}_F. \quad (4.30)$$

If $\mathbf{X}^\top \boldsymbol{\beta}_F = \mathbf{y}_F$, then $\boldsymbol{\beta}_F$ is the eigenvector of the eigen-problem in Eq. (4.29) with the same eigenvalue λ_m such that $m = 1$.

Proof. Replacing $\mathbf{X}^\top \boldsymbol{\beta}_F$ by the Fiedler vector \mathbf{y}_F on the left side of Eq. (4.29) yields

$$\mathbf{X}\mathbf{W}\mathbf{X}^\top \boldsymbol{\beta}_F = \mathbf{X}\mathbf{W}\mathbf{y}_F = \mathbf{X}\lambda_F \mathbf{D}\mathbf{y}_F = \lambda_F \mathbf{X}\mathbf{D}\mathbf{y}_F = \lambda_F \mathbf{X}\mathbf{D}\mathbf{X}^\top \boldsymbol{\beta}_F \text{ s.t. } F = N - 2$$

and shows that for $F = n = N - 2$, $\boldsymbol{\beta}_F$ is the eigenvector of the eigen-problem of Eq. (4.29) which concludes the proof. \square

Therefore, building upon [CHZ07], the projective functions of LPI can be determined in two consecutive steps for Fiedler vector estimation. First, the Fiedler vector \mathbf{y}_F associated with the second smallest eigenvalue of Eq. (2.6) must be computed. Then, for the Fiedler vector \mathbf{y}_F , the LPI method estimates a transformation vector $\boldsymbol{\beta}_F \in \mathbb{R}^M$ that satisfies $\mathbf{y}_F = \mathbf{X}^\top \boldsymbol{\beta}_F$ by solving the following least squares problem

$$\hat{\boldsymbol{\beta}}_F = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{m=1}^N (\boldsymbol{\beta}_F^\top \mathbf{x}_m - y_{m,F})^2, \quad (4.31)$$

where $y_{m,F}$ is the m th embedding point in \mathbf{y}_F and $\hat{\boldsymbol{\beta}}_F$ is the estimated transformation vector.

4.4.2.1.3 MOTIVATION AND PROBLEM STATEMENT

The previous section discussed the applicability of LPI for Fiedler vector computation. In particular, LPI may discover the hidden nonlinear structure by finding linear approximations to the nonlinear LE (for details, see [HCL04] and [HN04]). However, when using the least-squares objective function, outliers and heavy-tailed noise may have a large impact on the estimation of the transformation vector $\boldsymbol{\beta}_F$. This leads to errors in the computed Fiedler vector and, consequently, an information loss about the representation of the underlying graph structure using such a corrupted Fiedler vector computation. This section analyzes the effects of outliers and noise on

the eigen-decomposition of the Laplacian matrix. The analysis provides the theoretical basis and an understanding of the ideas underlying the proposed robust Fiedler vector estimation approach.

Outlier Effects on the Fiedler Vector

The effect of outliers on the eigen-decomposition has been analyzed in terms of two fundamental types of outliers and group similarity in Section 4.2. This section extends our theory by analysing outliers' effect on the Fiedler vector.

Based on the Type I outlier effect analysis in Section 4.2.4.2, if the affinity matrix has distinct blocks and the Type I outliers are disconnected, the Fiedler vector can be easily determined after removing these outliers. However, in real-world scenarios the true blocks are generally not distinct and/or outliers do have a few non-zero similarities which result in non-zero eigenvalues [CC15, ZP04]. Since the number of blocks K and the number of outliers N_I are unknown, directly using eigenvalues for outlier detection may be impossible in practice.

The following preposition provides a numerical understanding of Type I outliers' effect on the Fiedler vector.

Proposition 4.4.1. *For a definite nonnegative K block zero-diagonal symmetric affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ and the associated Laplacian $\mathbf{L} \in \mathbb{R}^{N \times N}$, let the eigenvectors have a norm such that $\|\mathbf{y}_m\|_2^2 = 1$ holds, where \mathbf{y}_m denotes the eigenvector associated with the m th eigenvalue of \mathbf{L} . Further, let $\tilde{\mathbf{y}}_F$ be the Fiedler vector associated with the eigenvalue that corresponds to an additive Type I outlier and let $\tilde{y}_{o_1, F}$ denote the embedding result of a Type I outlier in $\tilde{\mathbf{y}}_F$. Then, it follows that the Euclidean distance between embeddings of different blocks decreases to zero when the absolute value of the embedding result of the outlier increases to one, i.e. when $|\tilde{y}_{o_1, F}| \rightarrow 1$.*

Proof. See Appendix A.3.0.1. □

Motivated by the eigenvectors' crucial role in cluster analysis, the results of Proposition 4.4.1 can be extended to multiple eigenvectors. If a distance-based clustering approach, such as SC is applied on the eigenvectors that are the indicator vectors of N_I outliers, all non-outlying points are mapped to the same cluster as a result of Proposition 4.4.1, as $|\tilde{y}_{o_1, F}| \rightarrow 1$. This explains why SC breaks down in the presence of Type I outliers. Only if the number of outliers is known or can be deduced from the data set (e.g. because they perfectly match Definition 4.1.1), ignoring the indicator vectors of outliers can overcome this problem. However, in practice, Definition 4.1.1 may only hold approximately, and data points may vary in the degree of their outlyingness, making a binary detection challenging or inappropriate. In the following sections, we will therefore present

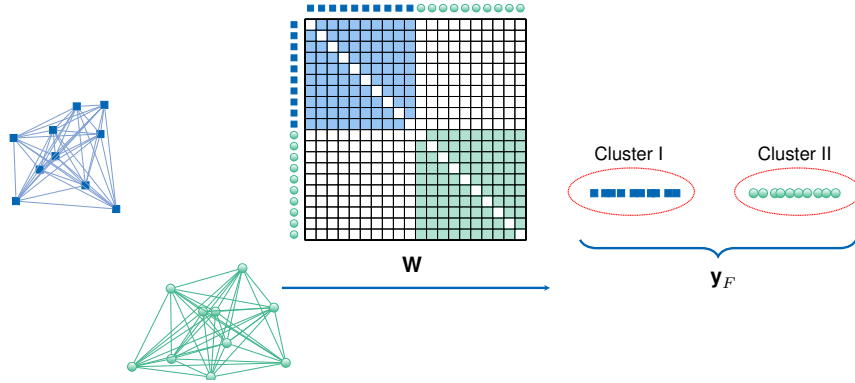


Figure 4.42: Fiedler vector computation for an ideal $K = 2$ blocks affinity matrix.

a robust M-estimation-based approach to suppress the impact of outliers, when such simple outlier detection and removal strategies do not apply.

Next, it is important to analyze the effect of Type II outliers on the eigenvectors to understand their particular effect on the Fiedler vector. Hence, the following proposition examines the extreme case of Type II outliers from a general perspective in terms of their effects on the eigenvectors.

Proposition 4.4.2. *For a K block zero-diagonal symmetric nonnegative affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$, let $w_i \in \{w_1, w_2, \dots, w_K\}$ denote a constant around which the correlation coefficients of the i th block are assumed to be concentrated with negligibly small variations. Further, let \tilde{w}_u denote a constant around which the correlations between blocks are concentrated. Let $\tilde{\mathbf{W}}$ define an affinity matrix, which is equal to \mathbf{W} , except that we impose $\tilde{w}_u > 0$, such that the vertices associated with i th and j th block become connected. Then, it follows that the connections between vertices corresponding to different blocks result in embedding all data-points onto the same location on the eigenvector $\tilde{\mathbf{y}}_0$ that is associated with the smallest eigenvalue $\tilde{\lambda}_0$ of the Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ corresponding to $\tilde{\mathbf{W}}$.*

Proof. See Appendix A.3.o.2. □

Even though Proposition 4.4.2 shows the loss of group structure in the eigenvector associated with the smallest eigenvalue, in real applications, this eigenvector might be the Fiedler vector when the data includes Type I outliers with negligibly small similarity coefficients.

To illustrate the different outlier effects, examples of computed Fiedler vectors are shown for ideal and corrupted affinity matrices in Figure 4.42 and Figure 4.43, respectively. In the ideal case, the vertices of different clusters do not have edges between each other while vertices of the same cluster are connected with strong edges. If such an ideally clustered graph is embedded on the

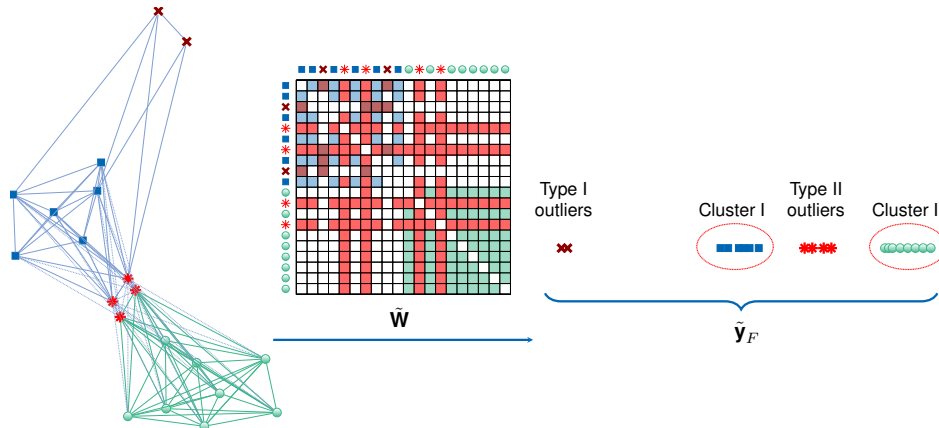


Figure 4.43: Fiedler vector computation for a corrupted $K = 2$ blocks affinity matrix. The corruptions of the affinity matrix by Type I and Type II outliers are highlighted by coloring the corresponding affected elements in light red and dark red, respectively. In the Fiedler vector, outliers are positioned as shown in the right illustration.

real line using the Fiedler vector, the vertices of the same cluster are concentrated while they are far away from the vertices of a different cluster, see Figure 4.42. Therefore, the embedding results of different clusters are easily separable, which is crucial for subsequent graph partitioning. On the other hand, the corrupted graph in Figure 4.43 includes two typical outlier effects. Based on the theoretical analysis, Type I outliers are embedded far from the clusters while Type II outliers that correlate with more than one cluster are embedded between different clusters making their separation difficult. In such scenarios, the outliers result in a performance degradation because of the computed Fiedler vector that would lead to losing the group structure information of the graph.

Outlyingness Measure: Overall Edge Weights

The overall edge weight is an informative measure for the determination of outliers which has been shown in Section 4.2.2 based on the real data examples. However, since the number of outliers is unknown in real-world scenarios a binary outlier detection based on overall edge weights may result in an information loss. Therefore, in the method that we proposed in [TMZ22], robustness is introduced by suppressing outliers' negative impact on the Fiedler vector rather than detecting and eliminating them. In this way, we allow for some uncertainty in our decision, giving moderate but non-zero weight to the points that we are not sure about.

Problem Statement

Given a data set of data-points $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{M \times N}$, the aim of this work is to estimate the Fiedler vector $\mathbf{y}_F \in \mathbb{R}^N$ such that it embeds each data-point on a real line, providing robustness at a reasonable computation cost. In the following section, the main ideas of the proposed robust Fiedler vector estimator are explained including an unsupervised penalty parameter selection procedure, an analysis of the computational cost, and possible applications of practical interest.

4.4.2.1.4 ROBUST FIEDLER VECTOR ESTIMATION

Let data matrix \mathbf{X} be subject to heavy tailed noise and outliers that obscure the underlying group structure in the graph $G = \{V, E, \mathbf{W}\}$ that represents \mathbf{X} . In Section 4.2.2, it was shown that the overall edge weight attached to a vertex is a valuable characteristic of an outlier because it significantly differs from the typical overall edge weight. Thus, the overall edge weight of attached to vertex m is modeled as

$$d_m = d_{\text{typ}} + \varepsilon_m. \quad (4.32)$$

Here, $d_m = \sum_n w_{m,n}$ and ε_m , respectively, denote the overall edge weight and the error term for the m th vertex, d_{typ} is the typical overall edge weight of the graph G . In practice, a robust estimator, such as the median is used, i.e. $\hat{d}_{\text{typ}} = \text{med}(\mathbf{d})$ for a vector of overall edge weights $\mathbf{d} = [d_1, \dots, d_N]$. For Fiedler vector estimation, an error vector $\boldsymbol{\varepsilon} \in \mathbb{R}^N$ is constructed using the error terms associated with each overall edge weight in \mathbf{d} . Then, the transformation vector $\boldsymbol{\beta}_F$ associated with \mathbf{y}_F is computed using penalized ridge regression M-estimation [ROK12] by solving the following zero gradient equation

$$-\sum_{m=1}^N \psi\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right) \left(\frac{\mathbf{x}_m^\top}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right) + \gamma \boldsymbol{\beta}_F = 0, \quad (4.33)$$

where γ denotes the penalty parameter, $\hat{\sigma}_{\boldsymbol{\varepsilon}}$ is a robust scale estimate of $\boldsymbol{\varepsilon}$ and ψ is a bounded and continuous odd function called the score-function. A popular M-estimator is defined by Huber's function

$$\psi\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right) = \begin{cases} \frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}, & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| \leq c_{\text{Hub}} \\ c_{\text{Hub}} \cdot \text{sign}\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right), & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| > c_{\text{Hub}} \end{cases}. \quad (4.34)$$

where c_{Hub} , commonly set to a default value of $c_{\text{Hub}} = 1.345$ for 95% asymptotic relative efficiency (ARE), is the tuning parameter that trades off robustness against outliers and ARE under a Gaussian distribution model for $\boldsymbol{\varepsilon}$ (see [ZKO18] for a detailed discussion). A frequently used

robust scale estimate is the normalized median absolute deviation [ZKO18] that is defined by

$$\hat{\sigma}_{\boldsymbol{\varepsilon}} = \text{madn}(\boldsymbol{\varepsilon}) = 1.4826 \cdot \text{med}|\boldsymbol{\varepsilon} - \text{med}(\boldsymbol{\varepsilon})|. \quad (4.35)$$

The motivation for adopting M-estimation for Fiedler vector estimation is that a bounded score function, such as Huber's, ensures that vertices with atypical edge weights are down-weighted in Eq. (4.33). RRLPI, therefore, softly suppresses the negative impact of outliers on the Fiedler vector estimate based on Huber's function. This becomes intuitively clear when considering Huber's weight function $\omega_m = \omega\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right)$:

$$\omega\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right) = \begin{cases} 1, & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| \leq c_{\text{Hub}} \\ c_{\text{Hub}} / \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right|, & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| > c_{\text{Hub}}, \end{cases} \quad (4.36)$$

that gives constant weight up to c_{Hub} and then increasingly down-weights outliers by smoothly descending towards zero. Under some conditions, e.g. when outliers are extremely large valued, a different weighting function instead of Huber's may be used to completely down-weight extreme outliers. For example, in robust statistics, Tukey's weight function

$$\omega\left(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right) = \begin{cases} \left(1 - \left(\left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| / c_{\text{Tuk}}\right)^2\right)^2, & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| \leq c_{\text{Tuk}} \\ 0, & \text{for } \left|\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}}\right| > c_{\text{Tuk}} \end{cases}, \quad (4.37)$$

is a popular choice, which gives zero-weight to extreme outliers. However, such a function leads to non-convex optimization problems, which is why, in many cases, Huber's weights are preferred (see [ZKO18] and [HR09] for a detailed discussion).

To provide an intuitive understanding, an exemplary plot is provided in Figure 4.44 that compares RRLPI with LE. Consistent with the outlier effect analysis that has been detailed in Section 4.4.2.1.3, the data-points which are mapped far away from any other cluster in the Fiedler vector \mathbf{y}_F are Type I outliers while the embeddings between different clusters are outliers of Type II. As can be seen, these outlying embeddings result in a performance degradation for clustering algorithms that are based upon a non-robust computation of \mathbf{y}_F . An important property of these outliers is that their occurrence depends on the determined level of sparsity. In more details, a non-sparse graph model results in over-connected vertices (Type II) whereas increasing sparsity redundantly results in less-connected or disconnected vertices (Type I). A robust sparsity level determination is therefore essential. Since the number of outliers is unknown, an outlier detection

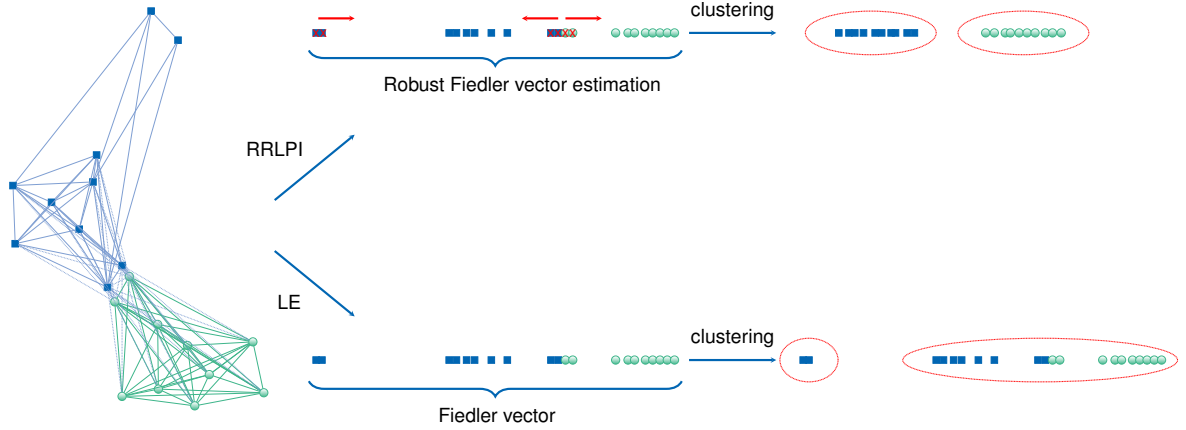


Figure 4.44: Exemplary plot of the Fiedler vector computation based on LE and RRLPI methods. The weighting operation in RRLPI on Type I and Type II outliers results in two clusters of concentrated mappings that include the outliers. In this way, the true structure of the non-outlying data becomes visible, even in the presence of outliers. By contrast, for LE, the outliers deteriorate the underlying two-cluster structure. Further, the weights provide a robust measure of outlyingness, which may be used to detect and analyze outliers, which is of high interest in some applications.

based on the available graph structure may result in misdetection or losing information. Therefore, instead of outlier detection, RRLPI down-weights the deviating embedding points based on their overall edge weights to achieve a robust Fiedler vector estimate in which the embeddings associated with the same cluster are concentrated while being separated from embeddings of different clusters.

Theoretical Analysis

The RLPI [CHZ07] method represents the Fiedler vector \mathbf{y}_F as the response of a linear regression with input variables \mathbf{X} , i.e. $\mathbf{y}_F = \mathbf{X}^\top \boldsymbol{\beta}_F$. Then, it determines the transformation vector $\boldsymbol{\beta}_F$ that minimizes a penalized residual sum of squares problem $\sum_{m=1}^N (\boldsymbol{\beta}_F^\top \mathbf{x}_m - y_{m,F})^2 + \gamma \|\boldsymbol{\beta}_F\|^2$. In RRLPI, this approach is generalized by defining $\boldsymbol{\beta}_F$ as the solution to Eq. (4.33) whose matrix form leads to

$$\hat{\boldsymbol{\beta}}_F = (\mathbf{X}\boldsymbol{\Omega}\mathbf{X}^\top + \gamma\sigma_\epsilon^2\mathbf{I})^{-1}\mathbf{X}\boldsymbol{\Omega}\mathbf{y}_F, \quad (4.38)$$

where $\boldsymbol{\Omega} \in \mathbb{R}^{N \times N}$ is diagonal matrix of weights defined by Eq. (4.36).

As discussed before, the LPI method uses a linearization of the embedding operation for the Fiedler vector computation. To understand the relationship between LPI and RRLPI, we first clarify the relation between RLPI and RRLPI.

Theorem 9. *RRLPI is a robustly weighted RLPI [CHZ07], and for $\boldsymbol{\Omega} = \mathbf{I}$, it gives identical solutions to RLPI based Fiedler vector computation.*

Proof. See Appendix A.4.o.1. □

From Theorem 9, it follows that for $\gamma > 0$ and/or $\mathbf{\Omega} \neq \mathbf{I}$, the estimated transformation vector $\hat{\boldsymbol{\beta}}_F$ is not the eigenvector of the eigen-problem in Eq. (4.29) which means that it is not associated with the Fiedler value. However, the following theorem shows in which cases $\boldsymbol{\beta}_F$ gives exactly the eigenvector of the eigen-problem in Eq. (4.29).

Theorem 10. *Suppose \mathbf{y}_F is the Fiedler vector associated with the second largest eigenvalue of the eigen-problem in Eq. (4.29). Further, let $\mathbf{\Omega} \in \mathbb{R}^{N \times N}$ and $\mathbf{\Psi} \in \mathbb{R}^{M \times M}$ be two weighting matrices such that $\mathbf{U}^\top \mathbf{\Psi} \mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top \mathbf{\Omega} \mathbf{V} = \mathbf{I}$. If \mathbf{y}_F is in the space spanned by row vectors of the weighted data matrix \mathbf{X}^* , for $\mathbf{X}^* = \mathbf{X} \mathbf{\Omega}$, the corresponding transformation vector $\hat{\boldsymbol{\beta}}_F$ estimated with RRLPI is the eigenvector of the eigen-problem in Eq. (4.29) as γ decreases to zero.*

Proof. See Appendix A.4.0.2. □

Based on Theorem 10, the estimated transformation vector $\hat{\boldsymbol{\beta}}_F$ is the solution of Eq. (4.29) for $\gamma \rightarrow 0$, and $\mathbf{U}^\top \mathbf{\Psi} \mathbf{U} = \mathbf{I}$, $\mathbf{V}^\top \mathbf{\Omega} \mathbf{V} = \mathbf{I}$. To understand the relationship between RRLPI and LPI, the results of this theorem are extended for all transformation vectors $\hat{\boldsymbol{\beta}}_m \in [\hat{\boldsymbol{\beta}}_0, \dots, \hat{\boldsymbol{\beta}}_{N-1}]$ for the case that the data space m is greater than the number of data-points n and the data-points are linearly independent, i.e. $\text{rank}(\mathbf{X}) = N$.

Corollary 10.1. *If the data-points are linearly independent, i.e. $\text{rank}(\mathbf{X}) = N$, all transformation vectors are solutions of Eq. (4.29) for $\gamma \rightarrow 0$, and $\mathbf{U}^\top \mathbf{\Psi} \mathbf{U} = \mathbf{I}$, $\mathbf{V}^\top \mathbf{\Omega} \mathbf{V} = \mathbf{I}$ which means that RRLPI is identical to LPI.*

Proof. See Appendix A.4.0.3. □

Δ -Separated Sets for Penalty Parameter Selection

The geometric structure of well-spread ℓ_2^2 -representations shows that the two sets \mathbf{s} and \mathbf{t} are well separated if every pair of points $s_m \in \mathbf{s}$ and $t_n \in \mathbf{t}$ are mapped at least $\Delta = \varphi(1/\log^{-2/3} N)$ apart in ℓ_2^2 distance [ARV09]. Inspired by well-spread ℓ_2^2 -representations, we propose a penalty parameter selection algorithm by projecting graph vertices onto the real line using RRLPI-based Fiedler vector estimation, such that every pair of two sets $s_m \in \mathbf{s}$ and $t_n \in \mathbf{t}$ is at least $\Delta = \varphi(1/\log^{-2/3} N)$ apart in ℓ_2^2 distance for the estimated penalty parameter.

Let $\gamma_r \in \boldsymbol{\gamma}$ be the r th candidate penalty parameter in Eq. (4.38) from a given vector of candidate penalty parameters $\boldsymbol{\gamma} = [\gamma_{\min}, \dots, \gamma_{\max}] \in \mathbb{R}^{N_\gamma}$. Further, suppose that for each candidate penalty parameter γ_r , there exists an associated Fiedler vector estimate $\hat{\mathbf{y}}_F^{(\gamma_r)}$ that projects the graph vertices onto the real line. The geometric structure of well-spread ℓ_2^2 -representations allows for designing the sets \mathbf{s} and \mathbf{t} by projecting the points on a random line such that, for a suitable constant κ , the

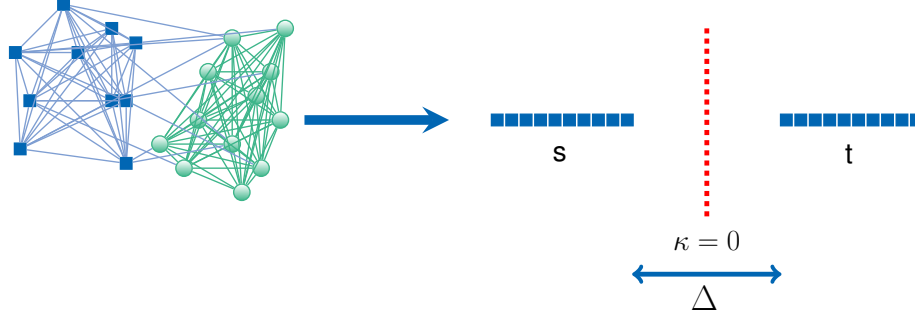


Figure 4.45: Example of Δ -separated sets \mathbf{s} and \mathbf{t}

points that are located on the left and right hand sides of κ are the initial candidates for the sets \mathbf{s} and \mathbf{t} , respectively [ARV09].

It has been shown (see, e.g. [ST07]) that it is possible to split a candidate Fiedler vector $\hat{\mathbf{y}}_F^{(\gamma_r)}$ into the two subsets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ for $\kappa = 0$. Another possible option for κ is the median of embeddings such that $\kappa = \text{med}(\hat{\mathbf{y}}_F^{(\gamma_r)})$. From the definition of the Δ -separated sets, the projection subsets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ associated with γ_r take values between zero and one. Therefore, after selecting the members of the two sets $\mathbf{s}^{(\gamma_r)} \in \mathbb{R}^{N_s}$ and $\mathbf{t}^{(\gamma_r)} \in \mathbb{R}^{N_t}$ associated with γ_r , the final design of the sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ is performed using the rescaled estimated Fiedler vector $\bar{\mathbf{y}}^{(\gamma_r)}$ as

$$\begin{aligned} \mathbf{s}^{(\gamma_r)} &= \{\bar{y}_{n,F}^{(\gamma_r)} : \hat{y}_{n,F}^{(\gamma_r)} > \kappa\} \\ \mathbf{t}^{(\gamma_r)} &= \{\bar{y}_{n,F}^{(\gamma_r)} : \hat{y}_{n,F}^{(\gamma_r)} \leq \kappa\}. \end{aligned} \quad (4.39)$$

Here, $\hat{y}_{n,F}^{(\gamma_r)}$ denotes the n th element of the estimated Fiedler vector $\hat{\mathbf{y}}_F^{(\gamma_r)}$ and $\bar{y}_{n,F}^{(\gamma_r)}$ is the n th element of the rescaled estimated Fiedler vector $\bar{\mathbf{y}}_F^{(\gamma_r)}$. If the rescaled Fiedler vector $\bar{\mathbf{y}}_F^{(\gamma_r)}$ is not sufficiently sparse, it contains pairs of points $\bar{y}_{m,F}^{(\gamma_r)} \in \mathbf{s}$ and $\bar{y}_{n,F}^{(\gamma_r)} \in \mathbf{t}$ whose squared Euclidean distance is less than Δ . Thus, for a set of pairs of projections $\bar{y}_{m,F}^{(\gamma_r)} \in \mathbf{s}$ and $\bar{y}_{n,F}^{(\gamma_r)} \in \mathbf{t}$, a vector of discarded projections $\mathbf{p}_{\text{disc}}^{(\gamma_r)} \in \mathbb{R}^{N_{\text{disc}}^{(\gamma_r)}}$ is designed as

$$\mathbf{p}_{\text{disc}}^{(\gamma_r)} = \left\{ \bar{y}_{m,F}^{(\gamma_r)}, \bar{y}_{n,F}^{(\gamma_r)} : \|\bar{y}_{m,F}^{(\gamma_r)} - \bar{y}_{n,F}^{(\gamma_r)}\|_2^2 \leq \Delta \right\}, \quad (4.40)$$

as long as the two sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ have a reasonable number N_{min} of projections. The proposed strategy to estimate penalty parameter γ is, therefore, to minimize the number of discarded points i.e.,

$$\hat{\gamma} = \arg \min_{\gamma_r = \gamma_{\text{min}}, \dots, \gamma_{\text{max}}} \{N_{\text{disc}}^{(\gamma_r)}\}, \quad (4.41)$$

where $N_{\text{disc}}^{(\gamma_r)}$ denotes the number of discarded projections for candidate penalty parameter γ_r , and $\hat{\gamma}$ is the estimated penalty parameter. In practice, there might not exist Δ -separated sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ for any candidate penalty parameter such that $\gamma_r \in \{\gamma_{\min}, \dots, \gamma_{\max}\}$. For example, the sets might not be Δ -separated, although $N_{\text{disc}}^{(\gamma_r)}$ has reached its maximum value. Additionally, the initial sets may not satisfy $N_{\mathbf{s}} < N_{\min}$ or $N_{\mathbf{t}} < N_{\min}$. In such cases, the penalty parameter can alternatively be estimated by maximizing the squared Euclidean distance between the closely valued projections from the two sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$,

$$\hat{\gamma} = \arg \max_{\gamma_r = \gamma_{\min}, \dots, \gamma_{\max}} \{ \|\bar{\mathbf{y}}_{\min, F}^{(\gamma_r)} - \bar{\mathbf{y}}_{\max, F}^{(\gamma_r)}\|_2^2 \}, \quad (4.42)$$

where $\bar{\mathbf{y}}_{\min, F}^{(\gamma_r)} \in \mathbf{s}$ and $\bar{\mathbf{y}}_{\max, F}^{(\gamma_r)} \in \mathbf{t}$ are the minimum and the maximum valued projections from the sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$, respectively.

In terms of robustness, approaches based on Δ -separated sets in Eq. (4.41) are advantageous compared to directly using Eq. (4.42). In particular, Eq. (4.42) may maximize the distance between Type I outliers and true samples while the necessity of a reasonable number N_{\min} of projections in different sets makes the Δ -separated sets robust against Type I outliers. Moreover, the Fiedler vector estimate $\hat{\mathbf{y}}_F^{(\gamma_r)}$ may contain Type II outliers which are embedded between true samples of sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$. In such cases, usage of Eq. (4.42) may result in losing a good penalty parameter due to the Type II outliers that obscure the real distance between sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$. In contrast, Δ -separated sets discard these Type II outliers up to a certain point and provide a robust measure of separation between the sets.

The main steps of the proposed Fiedler vector estimation are summarized in Algorithm 7.

Algorithm 7: Robust Fiedler vector estimation

Input: A data matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$ and an associated affinity matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$, N_{\min}

for $\gamma_r = \gamma_{\min}, \dots, \gamma_{\max}$ **do**

Initialization:

 Evaluate the Fiedler vector $\mathbf{y}_F \in \mathbb{R}^N$ via Eq. (2.6)

 Compute $\boldsymbol{\beta}_F \in \mathbb{R}^M$ for $\mathbf{y}_F = \mathbf{X}^\top \boldsymbol{\beta}_F$

RRLPI

 Update the error vector $\boldsymbol{\varepsilon} \in \mathbb{R}^N$ using Eq. (4.32)

 Compute $\hat{\sigma}_\varepsilon$ via Eq. (4.35)

 Calculate the weights $\omega_m = \omega(\frac{\varepsilon_m}{\hat{\sigma}_\varepsilon})$, $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\omega})$

 Solve Eq. (4.38) and estimate $\hat{\boldsymbol{\beta}}_F^{(\gamma_r)}$

 Estimate the Fiedler vector for $\hat{\mathbf{y}}_F^{(\gamma_r)} = \mathbf{X}^\top \hat{\boldsymbol{\beta}}_F^{(\gamma_r)}$

Δ -separated sets

 Generate sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ via Eq. (4.39)

 Calculate $\|\bar{\mathbf{y}}_{\min, F}^{(\gamma_r)} - \bar{\mathbf{y}}_{\max, F}^{(\gamma_r)}\|_2^2$ s.t. $\bar{\mathbf{y}}_{\min, F}^{(\gamma_r)} \in \mathbf{s}$ and $\bar{\mathbf{y}}_{\max, F}^{(\gamma_r)} \in \mathbf{t}$ and collect in a vector $\mathbf{z} \in \mathbb{R}^{N_\gamma}$

while $N_s \geq N_{\min}$ and $N_t \geq N_{\min}$ **do**

 Create $\mathbf{p}_{\text{disc}} \in \mathbb{R}^{N_{\text{disc}}^{(\gamma_r)}}$ using Eq. (4.40)

 Update $N_{\text{disc}}^{(\gamma_r)}$

if $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ are Δ -separated **then**

 | break

end

end

 Collect $N_{\text{disc}}^{(\gamma_r)}$ into a vector $\mathbf{h} \in \mathbb{R}^{N_\gamma}$

end

if at least one pair of Δ -separated sets exist **then**

 | Estimate $\hat{\gamma}$ using Eq. (4.41)

else

 | Estimate $\hat{\gamma}$ using Eq. (4.42)

end

Estimate transformation vector $\hat{\boldsymbol{\beta}}_F^{\hat{\gamma}}$ in Eq. (4.38) for $\hat{\gamma}$

Estimate the Fiedler vector for $\hat{\mathbf{y}}_F^{(\hat{\gamma})} = \mathbf{X}^\top \hat{\boldsymbol{\beta}}_F^{(\hat{\gamma})}$

Output: A robust estimate of the Fiedler vector $\hat{\mathbf{y}}_F^{(\hat{\gamma})}$

Computational Complexity

As computational complexity is essential for the scalability of graph embedding techniques, the computational complexity of the proposed approach is analyzed in terms of its main operations. The computational complexity of operations is detailed using the term flam [Ste98], which is a compound operation that includes one addition and one multiplication. For the cases when the complexity is not specified as flam, the Landau's big O symbol is used. The computational complexity of the proposed approach is given as follows:

Graph Construction: The pairwise cosine similarity which takes $\frac{1}{2}N^2M + 2NM$ as in [CHZ07] can be used for constructing graph G .

Initialization: For large eigen-problems, e.g. MATLAB© uses a Krylov Schur decomposition [Ste02]. The algorithm includes two main phases that are known as expansion and contraction. When N is larger than N_{Lan} , where N_{Lan} denotes the number of Lanczos basis vectors (preferably chosen as $N_{\text{Lan}} \geq 2N_{\text{eig}}$ for N_{eig} eigenvectors), the computational complexity of the algorithm can mainly be attributed to expansion and contraction phases. The expansion phase requires between $N(N_{\text{Lan}}^2 - N_{\text{eig}}^2)$ flam and $2N(N_{\text{Lan}}^2 - N_{\text{eig}}^2)$ flam. The contraction phase requires $NN_{\text{Lan}}N_{\text{eig}}$ flam [Ste01].

Robust Regularized Locality Preserving Indexing (RRLPI): The proposed projection algorithm requires an estimate of scale that uses repetitive medians. The complexity of repetitive medians is $O(N)$ [RB90]. Further, for a densely connected matrix, the complexity is mainly attributed to the Cholesky decomposition which is of complexity $O(N^3)$ or, more specifically, $\frac{1}{6}N^3$ flam [Ste98]. This complexity can be reduced to $O(N)$ using [Cou08] if the matrix is rank deficient. If the matrix is sparse, the computation cost of decomposition can be reduced to $t(2NN_{\text{fea}} + 3N + 5M)$ flam using a least squares algorithm such as [PS82] where N_{fea} denotes the average number of nonzero features and t is the number of iterations.

Δ -Separated Sets: To split the projection into two sets as s and t , the vector y must be sorted which is of complexity $O(N \log N)$ and there are computationally efficient alternatives such as [Han20] for which the complexity is reduced to $O(N\sqrt{\log N})$. To compute Δ -separated sets, a maximum of N projections can be subtracted which means that this operation maximally takes N flam.

Summing up the terms with respect to flam yields minimally

$$t(2NN_{\text{fea}} + 3N + 5M) + \frac{1}{2}N^2M + N(2M + N_{\text{Lan}}^2 - N_{\text{eig}}^2 + N_{\text{Lan}}N_{\text{eig}} + 1)$$

flam. Hence, the complexity is of order $O(N^2)$. Based on the information that both $O(N)$

and $O(N \log N)$ are considerably smaller than $O(N^2)$, the minimum computational cost can be summarized as $O(N^2)$ for each candidate penalty parameter. Overall, the algorithm is, at least, of complexity $O(N_\gamma N^2)$ for a number N_γ of candidate penalty parameters.

Example Applications

Eigenvector decomposition has a large variety of applications, such as, dimension reduction [CHZ07, CHH05, HCL04, HN04, DDF90], recognition [ZZL18, LT09, YTL06] and localization [SVB01]. Considering images as high-dimensional data sets, it is not surprising that eigen-decomposition is a fundamental research area also in image segmentation, e.g. [CHH05, HCL04, HN04, DDF90]. A frequently encountered problem is that the image is subject to noise, which may result in embedding noisy pixels far from the neighboring group of pixels in the embedding space and, consequently, losing the underlying structure. This problem may also occur in cluster enumeration approaches that attempt to find densely connected groups of embeddings in the projection space, which necessitates the application of a robust embedding technique. In the following, the example of robust graph-based cluster enumeration is discussed.

Cluster Enumeration: Assume that for each candidate number of clusters $K_{\text{cand}} \in \{K_{\min}, \dots, K_{\max}\}$ there is a clustering algorithm, e.g. [XG08, NJW01], that partitions $\hat{\mathbf{y}}^{(\hat{\gamma})}$ into K_{cand} number of clusters and provides an estimated label vector $\hat{\mathbf{c}}_{K_{\text{cand}}}$. After estimating label vectors for each candidate number of clusters K_{cand} , the cluster number K can be estimated by comparing quality of partitions using modularity as [CNM04]

$$\hat{K} = \arg \max_{\hat{K}_{\min}, \dots, \hat{K}_{\max}} \{ \text{mod}_{\hat{K}_{\text{cand}}} \}, \quad (4.43)$$

where $\text{mod}_{\hat{K}_{\text{cand}}}$ denotes the modularity score for a candidate number of clusters \hat{K}_{cand} that is computed using Eq. (2.7).

4.4.2.1.5 EXPERIMENTAL EVALUATION

This section contains the numerical experimental evaluation of the proposed RRLPI method on a broad range of simulated and real-world data sets with applications to robust cluster enumeration and image segmentation. In the following, a detailed information about experimental setting is provided. A MATLAB implementation of RRLPI is available at: <https://github.com/A-Tastan/RRLPI>

Benchmark Methods: The effects of Type I and Type II outliers on the Fiedler vector computation

are studied for the LE [BN01], LPI [HCL04], RLPI [CHZ07], RLPFM [TMZ21] and RRLPI embedding-based approaches by designing synthetic data Monte Carlo experiments. Then, in addition to the above mentioned embedding approaches, the proposed RRLPI is benchmarked against three state-of-the-art graph-based cluster enumeration approaches, i.e., Martelot [MH11], Combo [SCB14] and SPARCODE [TMZ21] and two state-of-the-art spectral partitioning approaches, i.e., fast large-scale spectral clustering via explicit feature mapping (FastEFM) [HRG18] and LSC [CC14] in terms of image segmentation capabilities.

Parameter Settings: Some of the competitors are parameter free approaches, i.e. LE [BN01], LPI [HCL04], Martelot [MH11], Combo [SCB14], SPARCODE [TMZ21] and LSC [CC14]. For the FastEFM approach [HRG18], the Gaussian scale parameter is the mean distance among all data points as the authors suggested. In terms of accuracy, the authors suggested to increase the desired dimension of explicit features. Therefore, the desired dimension is $D = 500$ as suggested by the authors. Further, to analyze the performance of proposed penalty parameter selection and to provide fair comparisons, the RLPI [CHZ07] and RLPFM [TMZ21] approaches are all run using the proposed penalty parameter selection algorithm. The remaining parameters of RLPI, RLPFM and RRLPI are defined using the default setting: $\gamma_{\min} = 10^{-8}$, $\gamma_{\max} = 1000$, $K_{\min} = 1$, $K_{\max} = 10$ and $N_{\min} = \frac{N}{K_{\max}}$.

Affinity Matrix Construction: To analyze the robustness of RRLPI, cosine similarity is used as the affinity matrix construction method in all experiments, unless otherwise specified.

Performance Measures: The average partition accuracy \bar{p}_{acc} is measured by evaluating

$$\bar{p}_{\text{acc}} = \frac{1}{NN_E} \sum_{m=1}^{N_E} \sum_{n=1}^N \mathbb{1}_{\{\hat{c}_n = c_n\}}, \quad (4.44)$$

where

$$\mathbb{1}_{\{\hat{c}_n = c_n\}} = \begin{cases} 1, & \text{if } \hat{c}_n = c_n \\ 0, & \text{otherwise} \end{cases}, \quad (4.45)$$

N is the number of observations, N_E is the total number of experiments, and \hat{c}_n and c_n are the estimated and ground truth labels for the n th observation, respectively.

The empirical probability of detection p_{det} , as defined in Eq. (4.15), is used to assess cluster enumeration performance.

The contour matching score F_{score} for boundaries and the Jaccard index J are used for the numerical performance analysis in the case of image segmentation [CLP13]. The F_{score} quantifies

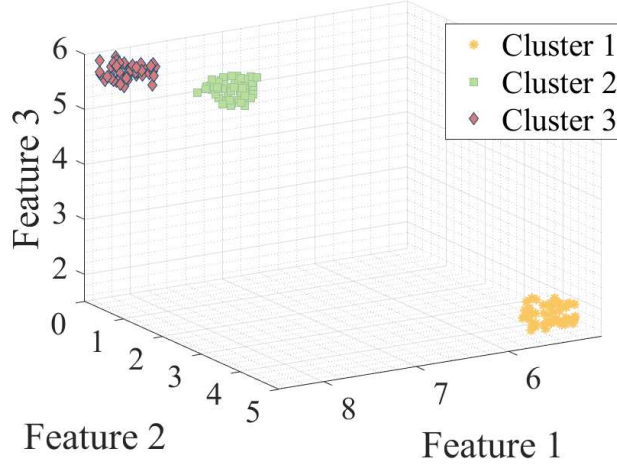


Figure 4.46: Exemplary plot of the first three features of the uncorrupted synthetic data set.

whether a boundary has a match on the ground truth boundary as follows

$$F_{\text{score}} = 2 \frac{P \cdot R}{R + P}, \quad (4.46)$$

where P and R denote precision and recall values, respectively. The Jaccard index evaluates similarity between estimated and ground truth segmentations according to

$$J(\hat{\mathbf{I}}_{\text{seg}}, \mathbf{I}_{\text{seg}}) = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}, \quad (4.47)$$

where $\hat{\mathbf{I}}_{\text{seg}}$ and \mathbf{I}_{seg} denote estimated and ground truth segmentations for image \mathbf{I} and TP, FP, and FN are true positives, false positives and false negatives, respectively.

Outlier Effects and Robustness

To visualize outlier effects on the eigen-decomposition, a synthetic data set is generated for $K = 3$ easily separable clusters, see Figure 4.46. The M -dimensional data-points of each cluster c_i , with $i = 1, \dots, K$, and $M = 6$ are generated as $\mathbf{x}_{m,i} = \boldsymbol{\mu}_i + \vartheta_i \mathbf{r}$, where $\mathbf{x}_{m,i}$ is the m th data-point associated with the i th cluster, $\boldsymbol{\mu}_i$ is the i th cluster centroid, ϑ_i is the i th scaling constant, and \mathbf{r} is a vector of independently and identically distributed random variables from a uniform distribution on the interval $[-0.5, 0.5]$. All details and parameter values to generate the data are provided in Appendix B.3.1.

Representative examples of the computed eigenvectors are shown for LE and for the proposed

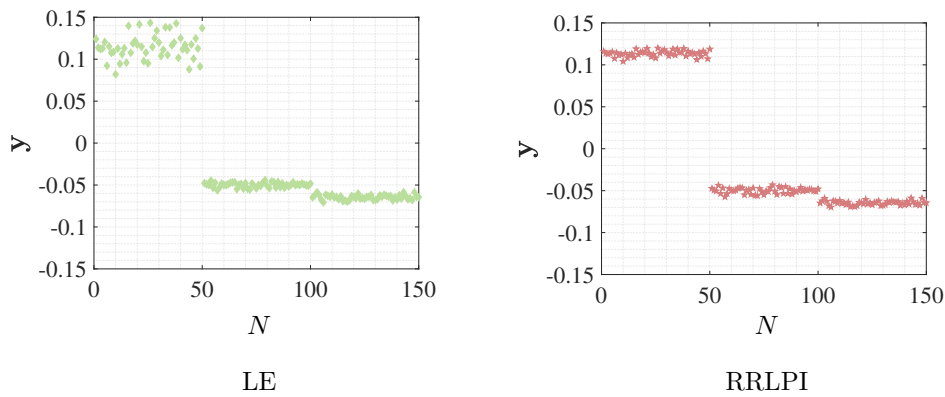


Figure 4.47: Computed eigenvectors for the uncorrupted data set.

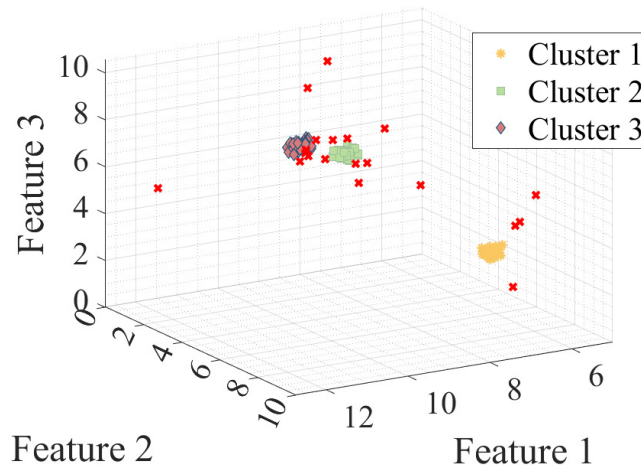


Figure 4.48: Exemplary plot of the first three features of the synthetic data set after corruption with Type I and Type II outliers (red crosses). See Section 4.4.2.1.3, for a discussion.

RRLPI in Figure 4.47. In the absence of outliers, both algorithms provide embeddings where the embedding points that are associated with the same cluster are concentrated, and the different clusters are separated. To study robustness, the data set is contaminated with two outlier types, i.e., outliers that do not correlate with any cluster (Type I) and outliers correlate with more than one cluster (Type II); see Sec. 4.4.2.1.3 for a definition and a discussion. An example showing the first three features of the contaminated data set is shown in Figure 4.48, where both Type I and Type II outliers are highlighted as red crosses.

The Type I and Type II outliers are, respectively, generated as $\tilde{\mathbf{x}}_m^{(I)} = \mathbf{x}_{m,i} + \vartheta_{o1}\mathbf{r}$ and

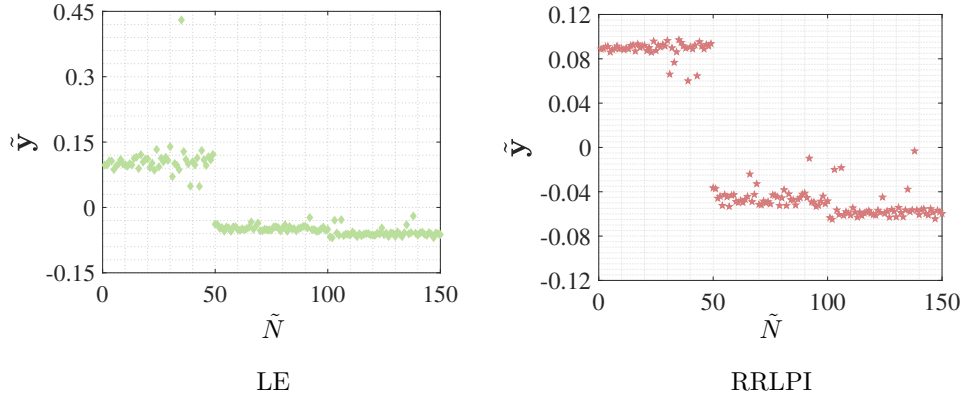
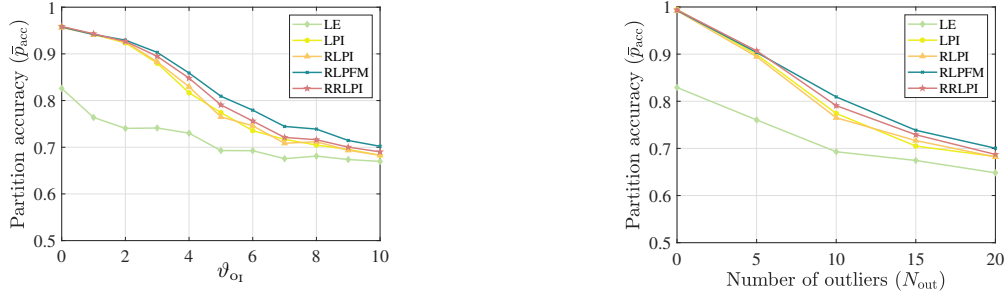


Figure 4.49: Computed eigenvectors for the corrupted data set.



\bar{p}_{acc} for increasing ϑ_{OI} associated with Type I outliers
($N = 300, N_{\text{out}} = 10, \vartheta_{\text{OI}} = 1.5, \vartheta_{e_K} = 0.5$ s.t. $i = 1, \dots, K$).

\bar{p}_{acc} for each of outlier type with increasing N_{out}
($N = 300, \vartheta_{\text{OI}} = 5, \vartheta_{\text{OI}} = 1.5, \vartheta_{e_K} = 0.5$ s.t. $i = 1, \dots, K$).

Figure 4.50: Average partition accuracy as a function of ϑ_{OI} and N_{out} for each outlier type

$\tilde{\mathbf{x}}_m^{(\text{II})} = \boldsymbol{\mu}_{\text{II}} + \vartheta_{\text{OI}} \mathbf{r}$ where $\tilde{\mathbf{x}}_m^{(n)}$, $n = \text{I, II}$ denotes the type of the outlier, ϑ_{OI} , $n = \text{I, II}$ is a scaling constant associated with the outlier type and $\boldsymbol{\mu}_{\text{II}}$ is a vector associated with the location of Type II outliers. A detailed explanation including all parameter values, is provided in the Appendix B.3.1. Examples of the eigenvector computations based on the corrupted data set are shown for LE and RRLPI in Figure 4.49, respectively. As can be seen, for the LE method, Type I outliers in the data produce outliers in the embedding results that obscure the underlying structure of $K = 3$ clusters. In contrast, the proposed RRLPI provides an embedding that is less influenced by the outliers.

Figure 4.50 reports the average partition accuracy as a function of the constant ϑ_{OI} associated with Type I outliers and the number of outliers N_{out} for each outlier type, respectively. The value of ϑ_{OI} is kept constant to generate points that lie between clusters two and three. The robust methods show best performance while the performance of LE quickly decreases in the presence of outliers.

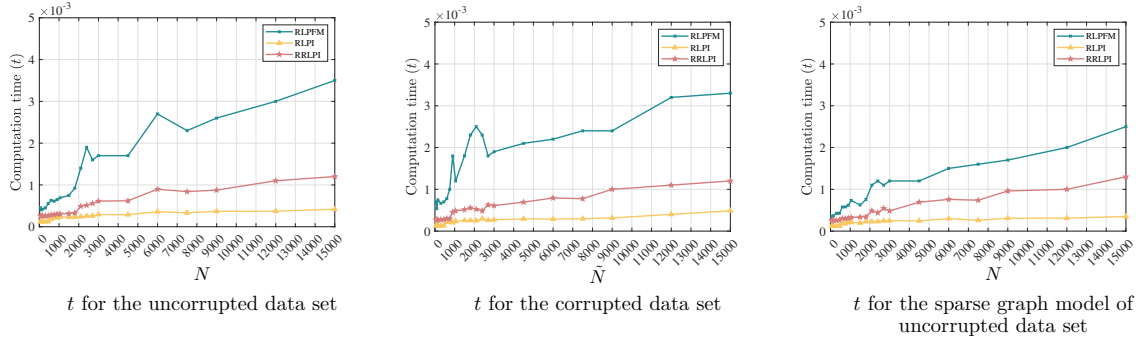


Figure 4.51: Computation time performance analysis. The results are reported in seconds.

Computation Time

The computation time (t) is reported as a function of increasing number of data points in the synthetic data set of $K = 3$ clusters. The experiments are performed based on three different scenarios. First, computation time is analyzed for an uncorrupted data set that has been explained in the previous section. Then, the data set is contaminated with two outlier types (Type I and II) where 10% of the data set are outliers for every type. The graphical models of these two data sets are generated based on the cosine similarity measure. Finally, to analyze the effect of sparsity, a sparse graph model of an uncorrupted data set is computed by using nearest neighbor graphs where the number of neighbors is set according to the cluster sizes. In all experiments, the penalty parameter is set to one and t is averaged over 100 Monte Carlo runs.

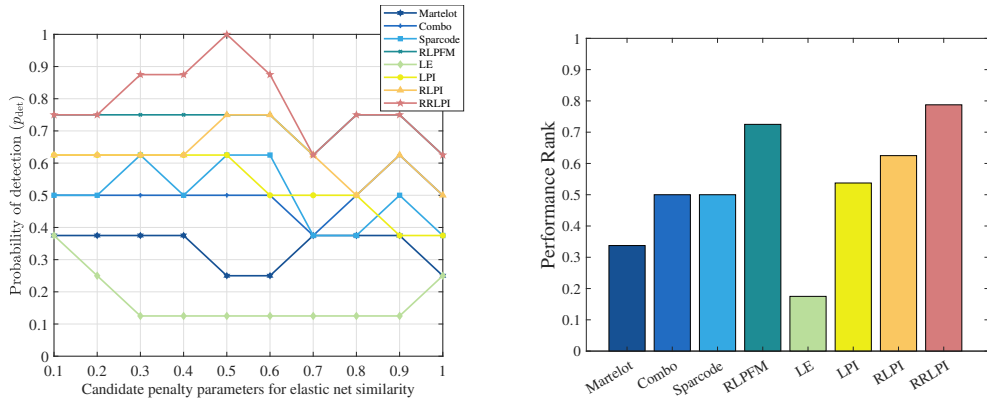
The performance of RRLPI is benchmarked against its main competitors RLPI and RLPFM in Figure 4.51. Even though robustness results in an increased computation cost, the single-step weighting procedure of RRLPI is considerably more efficient than the iterative weighting in RLPFM. The LPI method [HCL04] is excluded in the computation time analysis due to its different operational procedure. However, our own run-time analysis confirmed the theoretical analysis that RRLPI has quadratic complexity with respect to n . This means that the proposed method is computationally more efficient than LPI which has cubic complexity in unsupervised settings as it has been stated in [CHZ07].

Cluster Enumeration

In this section, the cluster enumeration performance of different approaches is benchmarked in terms of their empirical probability of detection using the following data sets: Gait [SAZ19], Breast Cancer [WM89], Fisheriris [Fis36], Person Id. [TSM18], Sonar [GS88], Ionosphere [SWH89],

Data Set	\hat{K} for Different Cluster Enumeration Methods									Similarity
	Martelot	Combo	SPARCODE	RLPFM	LE	LPI	RLPI	RRLPI	K	
Gait [SAZ19],	4	6	5	4	4	4	4	5	5	enet
Breast Cancer [WM89],	1	2	2	2	4	2	2	2	2	cos
Fisheriris [Fis36],	2	3	2	3	5	3	3	3	3	enet
Person Id. [TSM18],	6	7	4	5	10	4	4	4	4	enet
Sonar [GS88],	2	2	2	2	6	2	2	2	2	cos
Ionosphere [SWH89],	3	3	4	2	7	3	2	2	2	cos
D. Retinopathy [AH14],	2	2	2	2	2	2	2	2	2	cos
Gesture Phase S. [WPM14],	2	3	3	5	10	2	6	5	5	cos

Table 4.8: Performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\text{pen} = 0.5$.



(a) \bar{p}_{det} with respect to different penalty parameters for K -medoids partitioning with Tukey's distance function for the initialization ($T_{\text{Tukey}} = 3$).

(b) Performance rank for K -medoids partitioning with Tukey's distance function for the initialization ($T_{\text{Tukey}} = 3$).

Figure 4.52: Numerical results for cluster enumeration based on RRLPI.

D. Retinopathy [AH14] and Gesture Phase Segmentation (Gesture Phase S.) [WPM14].

If none of the cluster enumeration approaches estimates the cluster number correctly with the default cosine similarity, the elastic net similarity measure as in [ZKO18], is used with ten candidate penalty parameters ρ on an equidistant grid ranging from 0.1 to one. Results are reported for $\rho = 0.5$, which gave the best average overall detection performance for all methods. Tukey's distance function [ZKO18] where the threshold defined as $T_{\text{Tukey}} = 3$ is used as an initialization for K -medoids partitioning in the proposed algorithm. For a detailed discussion about different similarity measures and partitioning results, see Appendix B.3.3. The estimated cluster numbers are reported for the different cluster enumeration approaches in Tab. 4.8. As can be seen from the table, the gait and gesture phase data sets include a considerable number of outliers that result in

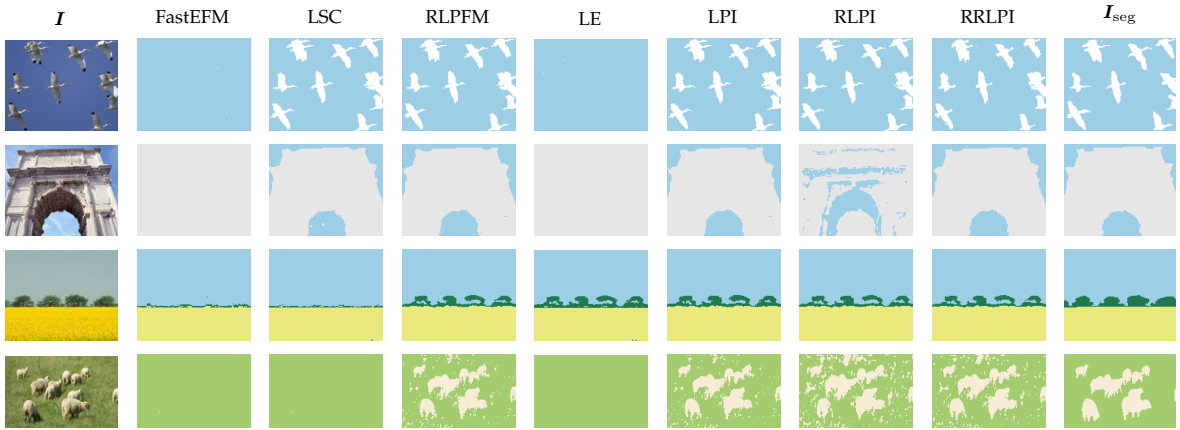


Figure 4.53: Image segmentation results for the original images.

mis-detection of the cluster number for almost all competitors. The proposed method is the only one that consistently estimates the correct cluster numbers for all data sets.

The empirical probability of detection with respect to different penalty parameters is detailed in Figure 4.52a. Then, the performance is summarized in Figure 4.52b by averaging the results over all penalty parameters. The results for cluster enumeration demonstrate that the proposed RRLPI shows the best probability of detection performance for all candidate penalty parameters with an average score of 79 %, whereas the best competitors (RLPFM and RLPI) have scores of 73 % and 63 %, respectively.

Image Segmentation

ADE_{20K} [ZZP17], is a large-scale dataset that includes high quality pixel-level annotations of 25210 images (20210, 2000, and 3000 for the training, validation, and test sets, respectively.). In our experiments, 10 images from the ADE_{20K} data set containing different objects, where each object has a different color, have been selected for color-based image segmentation. The selected and corresponding annotated images are denoted as \mathbf{I} and \mathbf{I}_{seg} , respectively. The images are down-sampled, where the dimension of data set \mathbf{X} is $\mathcal{M} = 3$ and $N \cong 15000$ using RGB color codes associated to down-sampled image pixels as features. To analyze robustness, the images are corrupted by adding multiplicative noise using the equation $\tilde{\mathbf{I}} = \mathbf{I} + \xi \times \mathbf{I}$, where $\tilde{\mathbf{I}}$ denotes the corrupted image and ξ is uniformly distributed random noise with zero mean and variance $\sigma^{(\xi)}$.

The down-sampled images are segmented for a pre-defined number of segments K using the default setting which performs K -means partitioning for the data sets that have more than $N = 3000$ samples. In Figure 4.53, examples of the original image \mathbf{I} and associated segmented images

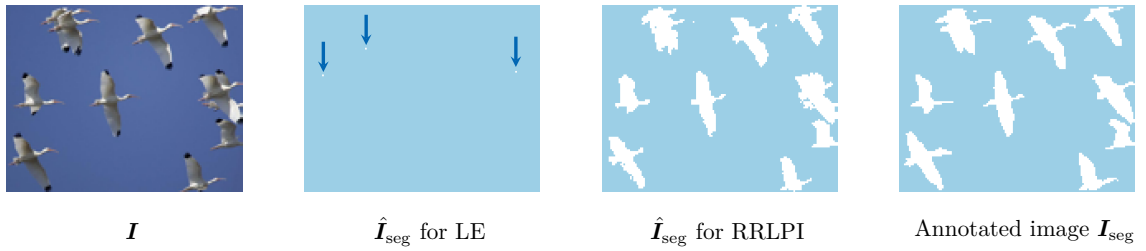


Figure 4.54: Example segmentations for LE and RRLPI methods. The embeddings that are mapped far away from the group of pixels are pointed out using arrows.

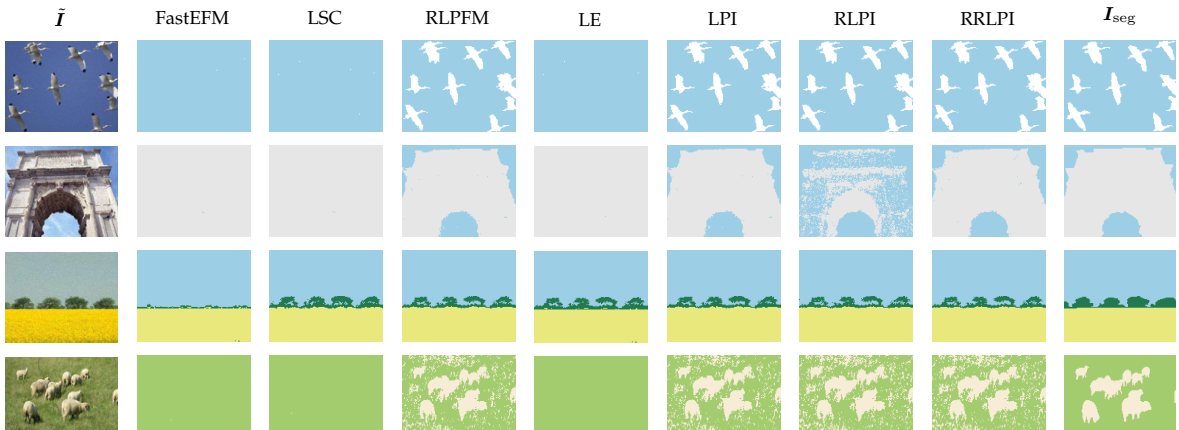


Figure 4.55: Image segmentation results for the corrupted images. ($\sigma^{(\xi)} = 10^{-3}$)

using the computed Fiedler vectors for seven different embedding approaches are shown along with the ground truth segmented image \mathbf{I}_{seg} . The uncorrupted images \mathbf{I} may also contain outlying pixels and/or noisy pixels. The effect of outliers is that a small number of pixels are mapped far away from the group of pixels and, thus, the remaining group of pixels assigned to a single large segment based on the distance-based partitioning methods.

A typical example of a segmentation result illustrating the outlier effects is provided in Figure 4.54. As can be seen, the described outlier effect is observed even for the embeddings of the uncorrupted (original) image when using LE. To exemplify the robust Fiedler vector estimation, the segmentation result of RRLPI is also shown. The segmentation result demonstrates that the proposed robust Fiedler vector estimation suppress outlier effects on the eigen-decomposition and provides segmentation results that are more consistent with the annotated image \mathbf{I}_{seg} . Further, in Figure 4.55, examples of segmented images are presented for the corrupted images where $\sigma^{(\xi)} = 10^{-3}$. The results show that the outlier effect on eigen-decomposition causes a breakdown of the FastEFM, LSC, and LE approaches. For further examples and detailed numerical results, see

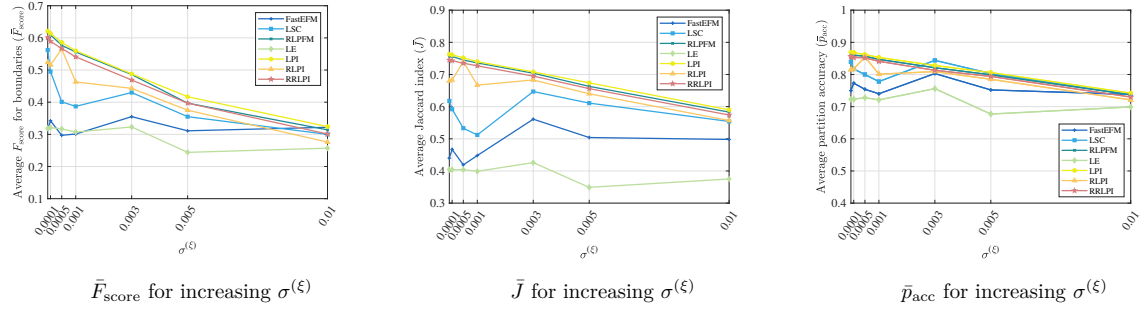


Figure 4.56: Numerical results for the image segmentation.

Appendix B.3.4.

The experiments are evaluated quantitatively using \bar{F}_{score} , \bar{J} and \bar{p}_{acc} , and the results are summarized in Figure 4.56. All performance measures are evaluated by comparing each estimated segmented image \hat{I}_{seg} with the annotated image I_{seg} . The LE and FastEFM show poor performance, even for the original images. Although LSC shows a reasonably good performance for the original images, its performance reduces drastically in the outlier-corrupted case in terms of \bar{F}_{score} and \bar{J} . The LPI, RLPFM and RRLPI are the top three methods in all performance measures and RRLPI follows them with a reasonably good performance, which indicates that the proposed penalty parameter selection algorithm is a promising approach, even when using non-robust methods.

4.4.2.1.6 CONCLUSION

Based on the derived theoretical results in Section 4.2, we proposed RRLPI, a method to robustly estimate the Fiedler vector that down-weights embeddings, for which the overall edge weight deviates from the typical overall edge weight of a given graph. The objective function to estimate the Fiedler vector is penalized using the proposed unsupervised penalty parameter selection algorithm that builds upon Δ -separated sets. The performance of RRLPI is benchmarked for different applications on a variety of real-world data sets. The numerical results for cluster analysis and image segmentation showed that the RRLPI is a promising approach for Fiedler vector estimation in situations where robustly determining the group structure in a data set is essential.

4.4.2.2 ROBUST SPECTRAL CLUSTERING: A LOCALITY PRESERVING FEATURE MAPPING BASED ON M-ESTIMATION

4.4.2.2.1 INTRODUCTION

Dimension reduction and feature extraction are fundamental in many clustering algorithms that have been intensively researched for decades [HRG18, CHZ07, CHH05, BN01]. As discussed in Section 2.5, SC is a simple and effective tool that relies on the eigenfunctions of the Laplace-Beltrami operator on a manifold to discover the intrinsic structure hidden in the data. It has various applications such as in face recognition and image segmentation [WQD14].

A popular way of estimating eigenvectors of a Laplacian is the method of LE [BN01], which is a manifold learning technique motivated by the correspondence between the graph Laplacian and the Laplace-Beltrami operator on a manifold. The term LE refers to a nonlinear method that embeds high-dimensional feature vectors into a low-dimensional vector space while preserving certain local properties. LPI is motivated by determining the optimal linear approximations to the eigenfunctions of the Laplace Beltrami operator in an attempt at discovering the inherent nonlinear structure. The computational complexity of LPI can mainly be attributed to computing a complete SVD and it has been reduced in [HRG18, CHZ07], making such approaches attractive in practice. However, in real-world scenarios the data may be corrupted by outliers and noise [ZKO18], leading to a performance degradation. Existing robust algorithms for SC have been proposed to minimize the effect of outliers in representation space, e.g. [LNC18] or in the projection operation [PYT15]. The robust projection operation as in [PYT15], uses the ℓ_1 norm that creates a different eigenbasis and it requires prior information about the data, such as, representative samples. To the best of our knowledge, an unsupervised robust projection algorithm that uses the ℓ_2 norm as in the eigen-decomposition of the original SC has not been proposed in the literature.

To integrate robustness in SC, in [TMZ21], we proposed a *robust locality preserving feature mapping (RLPFM)* and an *unsupervised penalty parameter selection algorithm* using the geometric structure of well-spread embeddings. Building upon regularized locality preserving indexing (RLPI), which is a computationally efficient extension of the LPI framework that regularizes the eigenvectors, we developed a robust M-estimation approach to feature embedding to mitigate the effect of outliers on the determination of the group structure. The penalty parameter, which is a key factor for the performance of RLPFM, was selected, such that the estimated Fiedler vector is Δ -separated with minimum information loss.

The following sections are organized as follows. Section 4.4.2.2.2 briefly revisits LPI while

Section 4.4.2.2.3 contains the motivation and problem formulation. Then, the developed robust SC method is detailed in Section 4.4.2.2.4. The performance of the proposed method in comparison to popular embedding and SC approaches is demonstrated in Section 4.4.2.2.5 and, finally, conclusions are drawn in Section 4.4.2.2.6.

4.4.2.2.2 LPI FOR SPECTRAL CLUSTERING

Suppose that a data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{M \times N}$ with M denoting the dimension and N the number of feature vectors, can be represented as a graph $G = \{V, E, \mathbf{W}\}$, where V denotes the vertices, E represents the edges, and $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the nonnegative definite affinity matrix that is computed from a similarity measure, e.g. cosine similarity as defined in Eq. (2.1). SC [NJW01] maps the original M dimensional feature vectors onto a smaller K dimensional vector space by finding the eigenvectors associated with the K smallest eigenvalues of the eigen-problem in Eq. (2.6).

According to the Theorem 8, for an eigenvector $\mathbf{y} \in \mathbb{R}^N$ with $\mathbf{y} = \mathbf{X}^\top \boldsymbol{\beta}$, the LPI method finds a transformation vector $\boldsymbol{\beta} \in \mathbb{R}^M$ that is the eigenvector associated with the smallest eigenvalue of the generalized eigen-problem in Eq. (4.29). This fundamental property of LPI, gives identical solutions to SC if the data matrix \mathbf{X} is a full rank square matrix. Thus, building upon [CHZ07], the LPI basis functions can be determined in two consecutive steps for SC. First, the K eigenvectors $\mathbf{y}_1, \dots, \mathbf{y}_K$ associated with the K smallest eigenvalues $\lambda_1 < \dots < \lambda_K$ in Eq. (2.6) is computed. Then, for each eigenvector $\mathbf{y}_i \in \mathbb{R}^N$, where $i = 1, \dots, K$, LPI estimates a transformation vector $\boldsymbol{\beta}_i \in \mathbb{R}^M$ that satisfies $\mathbf{y}_i = \mathbf{X}^\top \boldsymbol{\beta}_i$ by solving the following least squares problem

$$\hat{\boldsymbol{\beta}}_i = \operatorname{argmin}_{\boldsymbol{\beta}_i} \sum_m^N (\boldsymbol{\beta}_i^\top \mathbf{x}_m - y_{m,i})^2, \quad (4.48)$$

where $y_{m,i}$ is the m th mapping point in the i th eigenvector \mathbf{y}_i and $\hat{\boldsymbol{\beta}}_i$ is the estimated i th transformation vector.

4.4.2.2.3 MOTIVATION AND PROBLEM FORMULATION

Motivation

To motivate the use of robust methods, this section provides an illustrative discussion of possible outlier effects on SC. Figure 4.57a shows an example, where the data that consists of $N = 30$ feature vectors can be separated into $K = 3$ disjoint clusters by the popular LE method [BN01], which analyzes the eigenvectors corresponding to the three smallest eigenvalues. The ellipsoids around the yellow, blue, and green vectors highlight the discovered clusters. Figure 4.57b

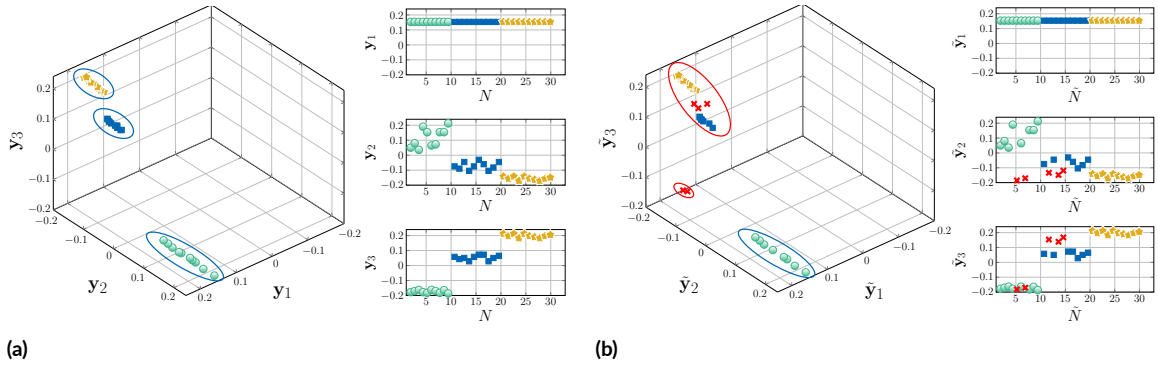


Figure 4.57: The eigenvectors associated with $K = 3$ smallest eigenvalues for a data matrix.

uses the same dataset, except that six blue and green points have been replaced by outliers that are marked as red crosses. In the context of clustering, outliers are, generally speaking, defined as data points that do not follow the cluster structure that is inherent to the large majority of the data. Consistent with the definitions in Section 4.1, we can distinguish two different types of outliers: On the one hand, an outlier may be a point that does not have any similarity with any of the clusters (Type I Outlier). On the other hand, an outlier may also be defined as a point that has considerable similarity with multiple clusters (Type II Outlier). In both cases, as illustrated in Figure 4.57b, the outliers obscure the cluster structure inherent to the eigenvectors. In this example, the popular LE method is not able to correctly split the data into the yellow, blue and green clusters. Instead, it opens up a cluster for the outliers that are not associated with any of the clusters, and it fuses the yellow and blue data points into a single cluster. Robust SC methods should be designed to be less sensitive to outliers. M-estimation is a widely used robust alternative to least-squares estimation when the data is subject to heavy-tailed noise and outliers [ZKO18]. Building upon the concepts of robust statistics [ZKO18], we propose an M-estimation approach, that down-weights outlying data points in the objective function, as will be detailed in the next section.

Problem Formulation

Given a dataset of feature vectors $\mathbf{X} \in \mathbb{R}^{M \times N}$, the goal of this method is to embed each feature vector into a K dimensional space where K denotes the specified number of clusters. Robustness implies that the method is not heavily affected by a few outliers in the data set.

4.4.2.2.4 ROBUST SPECTRAL CLUSTERING

This section is dedicated to robust SC that use penalized M-estimation for RLPFM. In the sequel, M-estimation for Locality Preserving Feature Mapping and a computational complexity analysis are provided.

M-estimation for Locality Preserving Feature Mapping

Assume that the dataset \mathbf{X} is corrupted by outliers and noise. The mappings in dimension-reduced space can then be written as

$$y_{m,i} = \boldsymbol{\beta}_i^\top \mathbf{x}_m + \varepsilon_{m,i}, \quad (4.49)$$

where $y_{m,i} \in \mathbb{R}$ denotes the mapping point for the m th feature vector \mathbf{x}_m and i th transformation vector $\boldsymbol{\beta}_i$, and $\varepsilon_{m,i} \in \mathbb{R}$ represents noise and additive outliers. For an embedding operation from the M dimensional space to the K dimensional space, the error vector $\boldsymbol{\varepsilon} \in \mathbb{R}^{N \times 1}$ is constructed by using embedding errors of all feature vectors as

$$\boldsymbol{\varepsilon}_m = \sum_i^K \varepsilon_{m,i}, \quad (4.50)$$

where $\varepsilon_i \in \boldsymbol{\varepsilon}$ denotes the embedding error of the m th feature vector. Similar to RRLPI [TMZ22] (for details, see Section 4.4.2.1), RLPFM softly suppresses the negative impact of outliers on the eigenvectors estimate based on Huber's function and performs Δ -separated sets for penalty parameter determination. However, there is an important difference in *how* the outliers are down-weighted. While RRLPI [TMZ22] compute the error vector $\boldsymbol{\varepsilon} \in \mathbb{R}^{N \times 1}$ based on the overall edge weights and incorporates the weighting operation into a single step for Fiedler vector estimation, RLPFM performs M-estimation of the multiple eigenvectors by iteratively reweighting the residuals.

After computing the error vector using Eq. (4.50), the RLPFM method adapts the remaining steps in Section 4.4.2.1.4 to multiple eigenvectors. Therefore, a summary of the RLPFM method including all steps is given in Algorithm 8.

Computational Complexity

The computational cost of operations is measured in flam [Ste98], and if the computational complexity is not specified using flam, the well-known Landau notation is used. The RLPFM requires $N(N_{\text{Lan}}^2 - K^2)$ to $2N(N_{\text{Lan}}^2 - K^2)$ flam for the expansion and $NN_{\text{Lan}}K$ flam for the contraction phases for the initialization of eigenvectors, where N_{Lan} is the number of Lanczos basis

Algorithm 8: Robust Spectral Clustering

Input: A data \mathbf{X} and affinity matrix \mathbf{W} , K, N_{\min}

Eigenvector Estimation using RLPFM

for $\gamma_r = \gamma_{\min}, \dots, \gamma_{\max}$ do

Initialization:

 Evaluate the eigenvectors $\mathbf{y}_1, \dots, \mathbf{y}_K$ as in Eq. (2.6)

 Get $\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_K$ for $\mathbf{y}_K = \mathbf{X}^\top \boldsymbol{\beta}_K$

RLPFM

 Compute the error vector $\boldsymbol{\varepsilon} \in \mathbb{R}^N$ using Eq. (4.50)

 Compute $\hat{\sigma}_{\boldsymbol{\varepsilon}}$ via Eq. (4.35)

 Calculate $\omega_m = \omega(\frac{\varepsilon_m}{\hat{\sigma}_{\boldsymbol{\varepsilon}}})$, $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\omega})$, via Eq. (4.36)

 Solve Eq. (4.38) and estimate $\hat{\boldsymbol{\beta}}_1^{(\gamma_r)}, \dots, \hat{\boldsymbol{\beta}}_K^{(\gamma_r)}$

 Estimate $\hat{\mathbf{y}}_1^{(\gamma_r)}, \dots, \hat{\mathbf{y}}_K^{(\gamma_r)}$ for $\hat{\mathbf{y}}_K^{(\gamma_r)} = \mathbf{X}^\top \hat{\boldsymbol{\beta}}_K^{(\gamma_r)}$

Δ -separated sets

 Generate sets $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ via Eq. (4.39)

 while $N_s > N_{\min}$ and $N_t > N_{\min}$ do

 Create $\mathbf{p}_{\text{disc}} \in \mathbb{R}^{N_{\text{disc}}^{(\gamma_r)}}$ using Eq. (4.40) and update $N_{\text{disc}}^{(\gamma_r)}$

 if $\mathbf{s}^{(\gamma_r)}$ and $\mathbf{t}^{(\gamma_r)}$ are Δ -separated then

 | break

 end

 end

 Collect $N_{\text{disc}}^{(\gamma_r)}$ into a vector $\mathbf{h} \in \mathbb{R}^{N_\gamma}$

end

Minimize the $N_{\text{disc}}^{(\gamma_r)}$ and estimate $\hat{\gamma}$ using Eq. (4.41)

Estimate $\hat{\boldsymbol{\beta}}_1^{\hat{\gamma}}, \dots, \hat{\boldsymbol{\beta}}_K^{\hat{\gamma}}$ for $\hat{\gamma}$

Estimate $\hat{\mathbf{y}}_1^{(\hat{\gamma})}, \dots, \hat{\mathbf{y}}_K^{(\hat{\gamma})}$ where $\hat{\mathbf{y}}_K^{(\hat{\gamma})} = \mathbf{X}^\top \hat{\boldsymbol{\beta}}_K^{(\hat{\gamma})}$

Partitioning

Get $\hat{\mathbf{c}}_K$ by applying the K -means on $\hat{\mathbf{y}}_1^{(\hat{\gamma})}, \dots, \hat{\mathbf{y}}_K^{(\hat{\gamma})}$

Output: An estimated label vector $\hat{\mathbf{c}}_K$ for K clusters

Dataset	SC	LPI	RLPI	FastEFM	LSC	RLPFM
Fisheriris [Fis36]	66.0	98.0	98.0	96.6	92.9	98.0
Breast Cancer [WM89]	62.9	88.2	87.4	72.1	85.4	87.0
Ionosphere [SWH89]	64.4	51.9	71.2	68.4	71.5	70.4
Parkinson A. [NPC16]	50.4	53.2	60.4	61.0	52.1	60.0
Sonar [GS88]	54.3	55.3	56.3	54.6	51.1	60.6

Table 4.9: K -means partitioning performance for real-world datasets. The average probability of detection shown in %.

vectors and K is the number of eigenvectors. The weighting operation of M -estimation requires repetitive medians that is of complexity $O(N)$. For a sparse matrix, the least squares algorithm, such as in [PS82] requires $t(2NN_{\text{fea}} + 3N + 5M)$ where t is the number of iterations and N_{fea} is the average number of nonzero features. However, if the matrix is dense, Cholesky decomposition requires $O(N^3)$ and in particular $\frac{1}{6}N^3$ flam [Ste98]. Lastly, the Δ -separated sets step requires $O(N \log N)$ time for sorting and a maximum of N flam for discarding for each candidate γ . In summary, for a sparse matrix the RLPFM step requires from

$$N_{\gamma}t(2NN_{\text{fea}} + 3N + 5M) + N(N_{\text{Lan}}^2 - K^2 + N_{\text{Lan}}K + N_{\gamma})$$

to

$$N_{\gamma}t(2NN_{\text{fea}} + 3N + 5M) + N(2N_{\text{Lan}}^2 - 2K^2 + N_{\text{Lan}}K + N_{\gamma})$$

flam in addition to $O(N_{\gamma}N)$, $O(N_{\gamma}N \log N)$ for repetitive medians and sorting where N_{γ} is the number of candidate penalty parameters.

4.4.2.2.5 EXPERIMENTAL RESULTS

In this section, the proposed RLPFM is compared with five state-of-the-art methods including embedding approaches LPI [CHH05] and RLPI [CHZ07] and SC approaches [BN01], FastEFM [HRG18], large scale spectral clustering with landmark-based sparse representation (LSC) [CC14]. The numerical experiments are performed with real-world databases Fisheriris [Fis36], Breast Cancer [WM89], Ionosphere [SWH89], Parkinson A. [NPC16], and Sonar [GS88] from the UCI machine learning repository. The parameter N_{min} for Δ -separated sets is defined as $N_{\text{min}} = \frac{N}{10}$ where different values of N_{min} do not have a huge impact as long as N_{min} is a reasonably small value. To analyze performance numerically, average clustering accuracy \bar{p}_{acc} is calculated by averaging clustering results for $N_E = 100$ repetitions and RLPI is performed with the proposed penalty parameter selection method to provide a fair comparison.

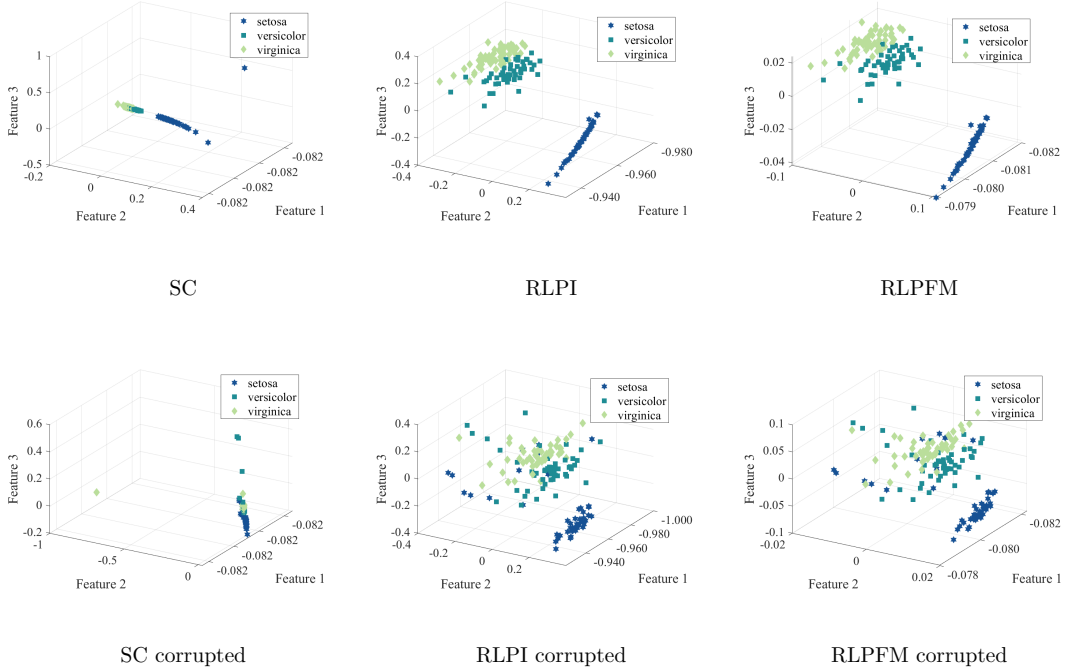


Figure 4.58: Examples of estimated feature spaces for corrupted and non-corrupted versions of the Fisheriris data set.

The clustering accuracy results are summarized for six different methods on five real-world datasets using K -means partitioning in Tab. 4.9. As can be seen, the SC method shows poor performance in terms of average clustering accuracy of 59.6% whereas almost all other clustering approaches have an average accuracy greater than 70%. The proposed RLPFM method outperforms all its competitors with 75.2% and RLPI follows it by a narrow margin reaching 74.7% which indicates that the proposed penalty parameter selection algorithm is a promising approach that can be used in other regularized feature mapping algorithms. We also implemented a simple plug-in robustification that replaces K -means by K -medoids, however, it did not improve the partitioning results, and is therefore not reported in detail.

Robustness

To evaluate robustness against outliers, we contaminated the Fisheriris dataset as follows. The outliers were generated as $\tilde{\mathbf{x}}_m = \mathbf{x}_m + \vartheta_c \mathbf{r}$, where \mathbf{r} denotes a vector of uniformly distributed random numbers in the interval $U(0,1)$, ϑ_c is a constant, \mathbf{x}_m and $\tilde{\mathbf{x}}_m$ are the original and corrupted m th feature vector for a randomly selected m , respectively. The examples of estimated eigenvectors for $K = 3$ clusters are shown in Figure 4.58 for the original and corrupted cases. For the corrupted case, the examples shown for $\vartheta_c = 5$ and the number of outliers in per cluster

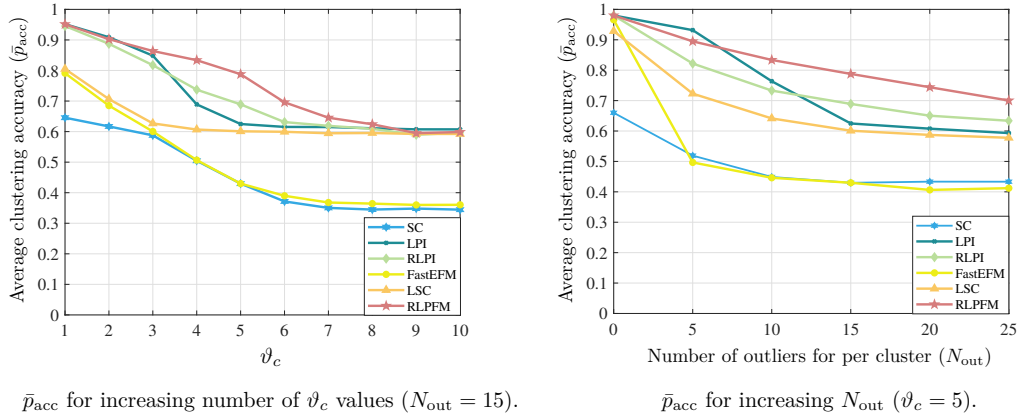


Figure 4.59: \bar{p}_{acc} for increasing ϑ_c and N_{out} values.

$N_{out} = 15$. The results of SC shows that, even the original Fisheriris dataset results in an outlier in the SC mappings that causes the method to break down. Figure 4.58 shows that both the RLPI and the proposed RLPFM produce similar and accurate mapping results for the original data. For corrupted data, RLPFM and RLPI approximately preserve the cluster structure, and RLPFM reduces the effect of outliers by mapping them closer to the cluster centers.

The clustering accuracy is detailed according to different ϑ_c and N_{out} values in Figure 4.59. Even though most of the algorithms have a clustering accuracy of more than 90% in the beginning, the performance of the competitors drops significantly after $\vartheta_c = 3$. The proposed method is also more robust for an increasing number of outliers while its main competitor RLPI follows it by approximately margin of 10%.

4.4.2.2.6 CONCLUSION

We proposed an unsupervised RLPFM including a penalty parameter selection approach for SC in [TMZ21]. The eigenvectors of the Laplacian matrix were reweighted and penalized by optimizing the penalty parameter, such that, the corresponding Fiedler vector is Δ -separated with minimum information loss. The method was benchmarked on different real-world datasets and it showed promising performance compared to five popular competitors, especially in terms of robustness against outliers and noise.

4.4.3 OUTLIER DETECTION BASED ON VERTEX DEGREE AND APPLICATION TO GAIT ANALYSIS

4.4.3.1 INTRODUCTION

As walking is the most practiced physical activity, it is not surprising that gait analysis has been the subject of intense scientific research. Doppler radar provides an efficient and privacy preserving way of analyzing human gait signatures that is independent of effects of clothing [LG02], or lighting condition [Ote05]. Doppler radar is widely used for the detection of falls and activity [AZA16, SHR14, WSR14], the identification of a person [Ote05], and for distinguishing human gait signatures from animal ones [ZPW07]. In addition to safety and security applications, the examination gait provides an ability to define gait abnormalities which plays a crucial role in medical diagnosis [HGM14]. Previous works on classification of gait abnormalities, e.g. [SAZ19, PWA15, WBH15] mainly focused on supervised learning algorithms. Obtaining labelled data requires a detailed examination of the available data which is inefficient, or even infeasible in real-world settings with large data sets containing considerable amounts of outliers. Similarity graphs are a powerful tool for unsupervised clustering [LSW16, CYY09] as they allow for representing clusters as communities. Yet, in graph model-based clustering, outlying entries have a negative impact on the connectivity of a graph, which severely affects clustering performance and makes community detection [TMZ21, SSS19, TMZ18] challenging.

Popular outlier detection methods proposed in the literature use distance or angle as a metric for outlyingness, e.g. [KSZ08, RL05, BS03]. Such methods are not well-suited for radar-based human gait data because outliers show a grouping effect and outliers of one cluster may overlap with typical data points of another cluster. As an illustration, a scatter plot of three important features of radar-based human gait data is shown in Figure 4.60. As it can be seen in the figure, the clusters corresponding to the 'Cane' and 'Limping two' clusters are grouped into two clusters due to outliers, and outliers of 'Cane' cluster overlap with the true samples of the normal walk cluster. Moreover, the outliers of 'Limping two' have a considerable sample size. The example shows that detection of outliers is challenging. Therefore, in such scenarios, an outlier detection procedure requires a different perspective on outlyingness beyond conventional outlier detection metrics such as distance or angle.

In [TMZ20], our main contribution was to propose *a new graph-based clustering algorithm* and to apply it to *label human gait signatures in a robust and unsupervised manner*. To this end, we first extracted a set of features and represented them as a weighted graph. Then, we identified outliers by

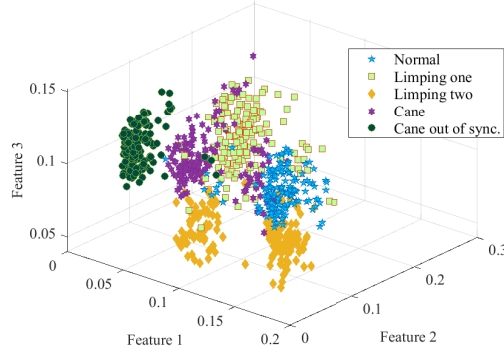


Figure 4.60: Scatter plot for three important features of radar-based human gait data belonging to five object communities.

robustly learning the typical degree of a vertex based on a dictionary that was formed from feature vectors that lie close to robustly estimated cluster centroids. This required estimating the number of clusters, which was done based on evaluating the modularity index after rejecting some vertices with very atypical degrees in a preprocessing step. The method outperformed existing robust and SC methods on a real-world data set (24 GHz radar system, 800 observations from five gait clusters of ten subjects).

The following sections are organized as follows. Section 4.4.3.2 comprises the problem formulation. The proposed cluster enumeration and the outlier detection methods are detailed in Section 4.4.3.3. Section 4.4.3.4 demonstrates the clustering performance for human gait radar data in comparison to four competing clustering algorithms. Finally, conclusions are drawn in Section 4.4.3.5.

4.4.3.2 PROBLEM FORMULATION

Given a data set consisting of M -dimensional feature vectors $\mathbf{X} \in \mathbb{R}^{M \times N}$, the aim of this method is to find a label vector $\mathbf{c}_K \in \mathbb{R}^N$ that partitions \mathbf{X} into K independent and mutually exclusive clusters. The true number of clusters K is unknown and $K \in \{K_{\min}, \dots, K_{\max}\}$, where K_{\min} and K_{\max} are, respectively, the prespecified minimum and maximum cluster number of clusters. Let $G = \{V, E, \mathbf{W}\}$, again, define a weighted graph, with V denoting the vertices, E represents the edges and $\mathbf{W} \in \mathbb{R}^{N \times N}$ being the affinity matrix. As discussed earlier in Section 2.5, SC [CYY09] is a popular unsupervised learning technique that allows for decomposing a data set into clusters based on the spectrum of the affinity matrix. However, as illustrated in Figure 4.60, radar-based human gait signatures include a considerable amount of outliers that may show a grouping effect, and outliers of one cluster may overlap with the data points of any other cluster. Hence, outliers

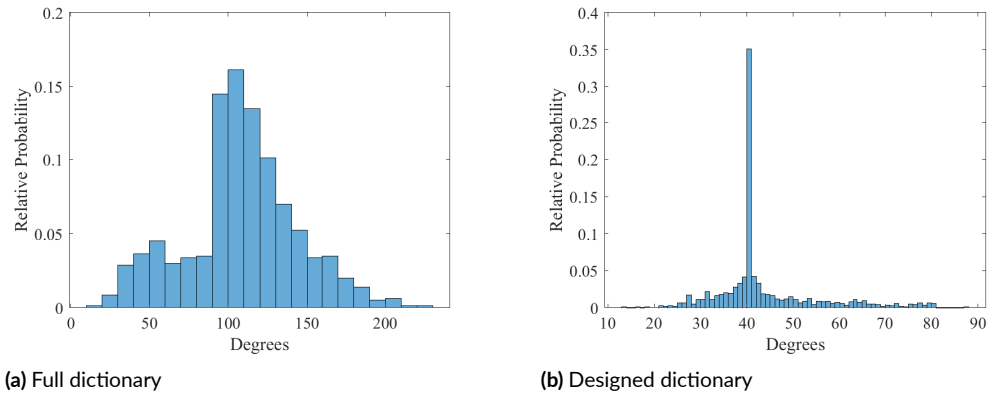


Figure 4.61: Normalized histogram of degrees for graph of radar-based gait signatures.

obscure the underlying data structure and cause a performance degradation in SC algorithms. In the following section, we therefore propose a graph-based SC approach that is robust against such outliers.

4.4.3.3 PROPOSED ALGORITHM

The main ideas of our algorithm are summarized as follows: we use sparse subspace representation to identify some of the outliers based on their atypical number of nonzero coefficients. Then, we estimate the number of clusters using modularity of graph partitioning (as defined in Eq. (2.7)). Next, robustly estimated centroids are used to identify a subset of feature vectors that are associated to the clusters with high confidence. Based on these, we build a dictionary and obtain sparse coefficient vectors. Repeating this procedure with many subsets allows for robustly learning the typical degree of a vertex. After learning the number of clusters and the typical degree, outliers can be rejected and a SC algorithm can be used to assign labels to the feature vectors. These steps are detailed in the following sections. A summary of the proposed method is given in Algorithm 9.

4.4.3.3.1 CLUSTER ENUMERATION

The algorithm is based on constructing a pairwise similarity graph which transforms the clustering into a graph partition problem [CYY09, WYG08]. Extensive data analysis on human gait Doppler radar data showed that outliers obscure the graph construction and severely influence the number estimated connections of the vertices. Figure 4.61 shows the normalized histograms for the degrees of a graph that is constructed on micro-Doppler radar data set of 800 observations with five different gait clusters of ten subjects (for details, see Sec. 4.4.3.4). Figure 4.61a illustrates

the empirical distribution of the degrees of the graph when dictionary learning is realized over all feature vectors in the data matrix \mathbf{X} . As can be seen in the figure, it is difficult to identify a typical degree over all vertices of the graph even though this data set contains an equal number of observations for each cluster, which implies a typical degree of 160. Clearly, the outliers, which have a low degree shift the largest mode to the left, create an additional mode at around 50. Figure 4.61b shows the empirical distribution of the degrees of the graph when the dictionary learning is applied N_t times over a selected subset of $N_s = 40$ samples that have been identified as typical for each cluster. Now, the empirical degree distribution is strongly focused around the true value, i.e., the graph structure has been correctly identified. However, the identification of these typical samples for each cluster requires a prior knowledge about the number of clusters. Considering the above observations, the estimation of the number of clusters is performed based on robustly estimated typical degrees for a full dictionary matrix by using the data matrix \mathbf{X} as a dictionary matrix. More precisely, we design an improved graph model using initially estimated feature vectors from \mathbf{X} that have typical degrees. The estimation of initial typical degrees is comprehensively explained in the following step.

Preprocessing

Based on the intuition that sparse outlying entries and noise have fewer non-zero correlation coefficients in coefficient vector \mathbf{a} [EV13], the outcome of this preprocessing step is a matrix $\check{\mathbf{X}}^{(0)}$ which consists of the initially estimated outlier-free columns of \mathbf{X} . To find these vectors, the empirical distribution of the number of nonzero coefficients for each coefficient vector \mathbf{a} must be determined. Assume that $\hat{\mathbf{A}}^{(0)} = [\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \dots, \hat{\mathbf{a}}_N] \in \mathbb{R}^{N \times N}$ is the estimated initial coefficient matrix over all coefficient vectors of $\mathbf{X} \in \mathbb{R}^{M \times N}$, where $\hat{\mathbf{a}}_m$ represents m th coefficient vector of \mathbf{X} . The initial coefficient matrix $\hat{\mathbf{A}}^{(0)}$ can be obtained solving the sparse subspace representation problem [EV13] if the solution is sparse enough [Don06]

$$\hat{\mathbf{A}}^{(0)} = \arg \min \|\mathbf{A}^{(0)}\|_1 \text{ s.t. } \mathbf{X} = \mathbf{X}\mathbf{A}^{(0)}, \text{diag}(\mathbf{A}^{(0)}) = 0, \quad (4.51)$$

where $\text{diag}(\mathbf{A}^{(0)}) \in \mathbb{R}^N$ is the vector of diagonal elements of $\mathbf{A}^{(0)}$. The initial degrees vector $\mathfrak{d}^{(0)} \in \mathbb{R}^N$ is formed by stacking the number of nonzero elements for each $\hat{\mathbf{a}}$. Based on $\mathfrak{d}^{(0)}$, we analyze the empirical distribution of the degrees. In particular, the normalized median absolute deviation of the degrees vector can be computed as [ZKO18]

$$\text{madn}(\mathfrak{d}^{(0)}) = 1.4826 \cdot \text{med}|\mathfrak{d}^{(0)} - \text{med}(\mathfrak{d}^{(0)})|, \quad (4.52)$$

where $\text{med}(\mathfrak{d}^{(0)})$ represents the median of the initial degrees vector $\mathfrak{d}^{(0)}$. Then, the initially estimated outlying samples are detected via the $2\hat{\sigma}$ rejection rule as the feature vectors that differ in their degree by more than two times the robustly estimated standard deviation. The preprocessed matrix $\check{\mathbf{X}}^{(0)}$ is designed using the remaining $\check{N}^{(0)}$ number of feature vectors from \mathbf{X} .

Graph Construction

Let $\check{G}^{(0)} = \{\check{V}^{(0)}, \check{E}^{(0)}, \check{\mathbf{W}}^{(0)}\}$ denote the weighted graph representation for the initially estimated outlier-free matrix $\check{\mathbf{X}}^{(0)}$. The weight matrix $\check{\mathbf{W}}^{(0)} \in \mathbb{R}^{\check{N}^{(0)} \times \check{N}^{(0)}}$ can be formed based on the estimated set of coefficient vectors from $\hat{\mathbf{A}}^{(0)}$, using Pearson's linear correlation coefficients, as

$$w_{m,n}^{(0)} = \frac{(\hat{\mathbf{a}}_m^{(0)} - \hat{\boldsymbol{\mu}}_m)^\top (\hat{\mathbf{a}}_n^{(0)} - \hat{\boldsymbol{\mu}}_n)}{\hat{\sigma}_m \hat{\sigma}_n} \quad (4.53)$$

with associated sample means $\hat{\boldsymbol{\mu}}_m, \hat{\boldsymbol{\mu}}_n$, and sample standard deviations $\hat{\sigma}_m, \hat{\sigma}_n$, respectively, for $m, n = 1, \dots, \check{N}^{(0)}$.

Graph Partition and Modularity

Assuming that for each candidate number of clusters $K_{\text{cand}} \in \{K_{\text{min}}, \dots, K_{\text{max}}\}$ there is a graph partitioning algorithm, such as [Hes04], that partitions $\check{\mathbf{X}}^{(0)}$ into K_{cand} clusters and provides the estimated label vector $\hat{\mathbf{c}}_c$, the cluster number K can be estimated by comparing quality of partitions as in Eq. (4.43). Herein, the only difference is averaging modularity score of candidate cluster number over N_t runs of a graph partitioning algorithm, where N_t is a reasonably large value.

Graph Construction-based Outlier Detection

Let $\hat{\mathbf{M}} \in \mathbb{R}^{M \times \hat{K}}$

$$\hat{\mathbf{M}} = [\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\mu}}_2, \dots, \hat{\boldsymbol{\mu}}_{\hat{K}}], \hat{\boldsymbol{\mu}}_i \in \mathbb{R}^M, \quad (4.54)$$

where $\hat{\boldsymbol{\mu}}_i$ represents the robustly estimated centroid vector for the i th cluster based on a \hat{K} -medoids partitioning of $\mathbf{X} \in \mathbb{R}^{M \times N}$. The dictionary matrix $\mathbf{H} \in \mathbb{R}^{M \times N_{\text{test}}}$ is formed by picking columns from \mathbf{X} , such that,

$$\mathbf{H} = [\mathbf{x}_{1,1}, \dots, \mathbf{x}_{1,N_s}, \dots, \mathbf{x}_{\hat{K},1}, \dots, \mathbf{x}_{\hat{K},N_s}], \mathbf{x}_{i,n} \in \mathbb{R}^M, \quad (4.55)$$

where $\mathbf{x}_{i,n}$ represents the feature vector from the i th cluster that has the n th Euclidean distance to the cluster centroid estimate $\hat{\boldsymbol{\mu}}_i$, N_s is a reasonably small value of the number of typical feature

vectors for each cluster where $N_{\text{test}} \ll N$ and $N_{\text{test}} = N_s \hat{K}$. Considering N_{test} test sample vectors from data matrix \mathbf{X} , the coefficient matrix $\hat{\mathbf{A}} \in \mathbb{R}^{N_{\text{test}} \times N_{\text{test}}}$ can be obtained solving the sparse representation problem

$$\hat{\boldsymbol{\alpha}}^{(t)} = \arg \min \|\boldsymbol{\alpha}^{(t)}\|_1 \text{ s.t. } \mathbf{x}_{\text{test}}^{(t)} = \mathbf{H}\boldsymbol{\alpha}^{(t)} \quad (4.56)$$

where $\hat{\boldsymbol{\alpha}}^{(t)}$ represents the estimated coefficient vector, $\mathbf{x}_{\text{test}}^{(t)} \in \mathbb{R}^M$ is a randomly selected sample test vector from data matrix \mathbf{X} for the t th dictionary learning iteration, and \mathbf{H} is designed dictionary matrix. To recover the degree information of each sample test vector in data matrix \mathbf{X} , the dictionary learning that is defined in Eq. (4.56) must be performed $t = \{1, \dots, N_t\}$ times, where N_t is a reasonably large number. The full degree vector can be obtained by stacking each degree vector $\mathfrak{d}^{(t)} \in \mathbb{R}^{N_{\text{test}}}$ into degree vector $\text{vec}(\mathfrak{D}) \in \mathbb{R}^{N_{\text{test}}N_t}$ for each dictionary learning iteration routine. The following step is to analyze the empirical distribution of all degrees. A $2\hat{\sigma}$ outlier rejection can be computed based on $\text{madn}(\text{vec}(\mathfrak{D}))$.

The proposed framework for the unsupervised graph-based robust clustering is summarized in Algorithm 9.

4.4.3.4 EXPERIMENTAL RESULTS

Experimental Radar Data

The experimental data, as in [SAZ19], has been collected in an office environment at Technische Universität Darmstadt using a 24 GHz radar system. The recordings include 16 observations for ten subjects and five different object clusters that consist of normal walk (NW), limping with one leg (L1), limping with two legs (L2), walking with a cane (CW) and walking with a cane out of synchronization (CW_{oos}). The data was recorded when the subject was walking towards and away from the radar. The observation number for each direction is equal to eight and the duration of data measurement is six seconds. In total, 800 observations for five different gait clusters of ten subjects have been used in our experiments.

Feature Extraction

The human gait features include physical features as defined in [SAZ19], and additional features that are extracted from the spectrogram and its envelope. In total, 89 gait features are used for clustering of gait signatures.

Physical Features [SAZ19]: The physical features of five gait classes were obtained by using the

Algorithm 9: Unsupervised Graph-based Clustering

Input: A normalized data set $\mathbf{X} \in \mathbb{R}^{M \times N}$, i.e. $\|\mathbf{x}_m\|_2 = 1$, for $m = 1, \dots, N$

Step 1: Cluster Enumeration

Step 1.1: Preprocessing

Create $\hat{\mathbf{A}}^{(0)} = [\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \dots, \hat{\mathbf{a}}_N]$ using Eq. (4.51)

Obtain $\mathfrak{d}^{(0)} \in \mathbb{R}^N$ for a set of $\hat{\mathbf{a}}$ vectors

Compute $\text{madn}(\mathfrak{d}^{(0)})$ of $\mathfrak{d}^{(0)}$ via Eq. (4.52)

Reject initially estimated outliers based on $2\hat{\sigma}$ rule

Reconstruct $\hat{\mathbf{A}}^{(0)}$ for initially estimated outlier-free samples as $\hat{\mathbf{A}}^{(0)} \in \mathbb{R}^{\check{N}^{(0)} \times \check{N}^{(0)}}$

Step 1.2: Graph Construction

Construct graph for $\check{\mathbf{W}}^{(0)} \in \mathbb{R}^{\check{N}^{(0)} \times \check{N}^{(0)}}$ via Eq. (4.53)

Step 1.3: Graph Partition and Modularity

for $t = 1, \dots, N_t$ **do**

for $K_{\text{cand}} = K_{\text{min}}, \dots, K_{\text{max}}$ **do**

 Apply a graph partitioning such as [Hes04]

 Evaluate $\text{mod}_{K_{\text{cand}}}$ of partition via Eq. (2.7)

end

end

Compute average of modularity scores for each K_{cand}

Obtain \hat{K} as in Eq. (4.43)

Step 2: Outlier Detection

Get set of $\hat{\boldsymbol{\mu}}$ for \hat{K} clusters by using K -medoids

Get matrix $\hat{\mathbf{M}} \in \mathbb{R}^{M \times \hat{K}}$ via Eq. (4.54)

Form $\mathbf{H} \in \mathbb{R}^{M \times N_{\text{test}}}$ via Eq. (4.55)

for $t = 1, \dots, N_t$ **do**

 Estimate $\hat{\mathbf{a}}^{(t)} \in \mathbb{R}^{N_{\text{test}}}$ of each test vector $\mathbf{x}_{\text{test}}^{(t)}$ via Eq. (4.56)

 Create $\hat{\mathbf{A}}^{(t)} = [\hat{\mathbf{a}}_1^{(t)}, \hat{\mathbf{a}}_2^{(t)}, \dots, \hat{\mathbf{a}}_{N_{\text{test}}}^{(t)}] \in \mathbb{R}^{N_{\text{test}} \times N_{\text{test}}}$

 Obtain $\mathfrak{d}^{(t)} \in \mathbb{R}^{N_{\text{test}}}$ for a set of $\hat{\mathbf{a}}$ vectors and stack in $\text{vec}(\mathfrak{D}) \in \mathbb{R}^{N_{\text{test}} t}$

end

Obtain full degree vector $\text{vec}(\mathfrak{D}) \in \mathbb{R}^{N_{\text{test}} N_t}$

Compute $\text{madn}(\text{vec}(\mathfrak{D}))$

for $n = 1, \dots, \check{N}^{(0)}$ **do**

 Reject outliers based on the $2\hat{\sigma}$ rejection rule

 Stack estimated outlier-free vectors into matrix $\check{\mathbf{X}} \in \mathbb{R}^{M \times \check{N}}$

end

Step 3: Spectral Clustering

Obtain $\check{\mathbf{W}} \in \mathbb{R}^{\check{N} \times \check{N}}$ via Eq. (4.51), Eq. (4.53)

Apply the SC algorithm as in [NJW01] replacing the K -means by K -medoids

Output: A vector $\hat{\mathbf{c}}_{\hat{K}}$ for \hat{K}

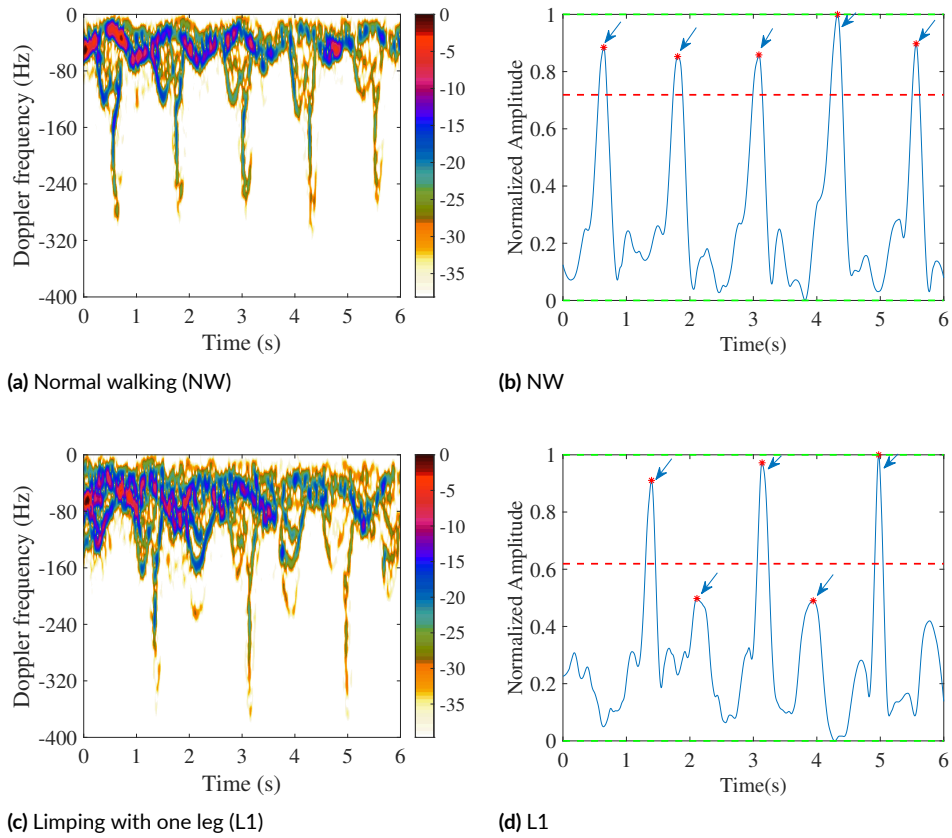


Figure 4.62: Example of (a), (c) spectrograms and (b), (d) corresponding micro-Doppler envelopes for two object clusters.

sum-of-harmonics model and the spectrogram of human gait signatures. The physical features include the fundamental frequency f_o , micro-Doppler frequency f_{mD} , the maximum Doppler frequency shift f_{max}^D and the gait harmonic frequency ratio β .

Additional Features: As in sparse representation of images, the features are taken as samples from the spectrogram and its envelope. In the context of human gait signatures, features must be carefully designed based on data analysis.

Considering different Doppler radar representations, see examples shown in Figure 4.62, our aim is to find descriptive features of object clusters. The maximum Doppler frequency shift and its neighbouring samples, obtained from the spectrogram, provides a representative feature for the L2 cluster. Moreover, samples from the maximum peak of the spectrogram envelope are used to distinguish the L2 cluster. To demonstrate rhythm of strides, samples from consecutive peaks are extracted which are represented with arrows in Figure 4.62b and 4.62d. Furthermore, the duration of time samples between peaks which are higher than 80% of the average of peaks that

Cluster Enumeration of Radar-based Human Gait Signatures		
Method	\hat{K}	K
Gaussian EM [TMZ18]	9	5
t Distribution-based EM [TMZ21]	10	5
Louvain [BGL08]	4	5
First Neighbor Relations [SSS19]	8	5
Proposed Unsupervised Graph-based Robust Clustering	5	5

Table 4.10: Cluster enumeration results of different parameter-free clustering algorithms for five object clusters.

is shown with a red dotted line in Figure 4.62b and Figure 4.62d, is taken to capture stride time and rhythm characteristics simultaneously. Stride time and rhythm are distinctive features for the CW_{00s} and the L_1 clusters, respectively. Finally, the Doppler shift samples between maximum and minimum points of an envelope are extracted to show peaks without micro-Doppler effect. All time-frequency representations are examined in a window that includes the samples from 0.3 to 5.7s.

Human Gait Data Clustering and Labelling

In this section, clustering results of five different parameter-free clustering algorithms are shown which provide a proof-of-concept that graph-based robust clustering is a useful tool for radar based gait analysis. The cluster enumeration results of the proposed unsupervised graph-based robust clustering algorithm are compared with state-of-the-art parameter-free clustering methods that include both statistical and graph-based approaches. In particular, we compare to the Gaussian expectation maximization (EM) [TMZ18], t distribution-based EM algorithm [TMZ21], graph modularity scoring with gain [BGL08] and first neighbor relations [SSS19]. The proposed outlier detection approach is compared with three different outlier detection methods that use angle [KSZ08], distance [BS03] and correlations between the variables [RB18]. In order to compare the success of partitioning the data set, the unsupervised clustering results of the proposed method are compared with four different clustering algorithms for which the number of clusters K was correctly provided.

For the EM algorithms, the dimension of the data is reduced to 10 by using PCA because the performance was poor on the original 89 dimensional data set. The estimated number of clusters for five different parameter-free cluster enumeration methods are summarized in Tab. 4.10. As can be seen, the three competitor cluster enumeration methods take outliers as groups and overestimate

Gaussian EM [TMZ18]					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	93.8	2.3	-	3.9	-
L1	-	99.4	-	0.6	-
L2	1.2	1.8	58.7	38.3	-
CW	21.5	15.3	-	63.2	-
CW _{oos}	1.9	0.6	-	-	97.5
t Distribution-based EM [TMZ21]					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	91.6	-	0.6	6.6	1.2
L1	-	95.6	-	4.3	0.1
L2	1.2	-	58.7	38.9	1.2
CW	21.3	-	0.6	78.1	-
CW _{oos}	-	-	-	8.7	91.3
Louvain [BGL08]					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	79.1	0.6	-	6.6	13.7
L1	3.1	91.9	-	1.9	3.1
L2	-	0.6	79	6.6	13.8
CW	20.6	3.1	-	75	1.3
CW _{oos}	1.2	-	-	1.3	97.5
ℓ_1 -Graph [CYY09]					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	89.5	-	4.6	5.9	-
L1	0.8	95.3	1.4	2.3	0.2
L2	6.1	0.8	91	1.9	0.2
CW	17.9	2.6	4.5	71.0	4.0
CW _{oos}	-	-	-	4.6	95.4
ℓ_1 -Graph [CYY09] with Outlier Detection [RB18]					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	41.8	9.1	9.1	6.4	33.6
L1	11.3	46.4	7.6	5.1	29.6
L2	11.8	8.4	42.4	6.2	31.2
CW	7.8	9.3	8.2	41.7	33
CW _{oos}	9.7	9.4	8	30.6	42.3
Proposed Unsupervised Graph-based Robust Clustering					
True / Predicted	NW	L1	L2	CW	CW _{oos}
NW	96.7	-	1.3	2	-
L1	-	98.3	1.6	0.1	-
L2	3.7	0.3	94.8	0.2	1.0
CW	12.2	-	3.1	81.2	3.5
CW _{oos}	-	-	-	6.6	93.4

Table 4.11: Confusion matrices of different gait clustering algorithms for five object clusters. Numbers are shown in % and best performance results are indicated in bold font.

the true cluster number. On the other hand, Louvain assumes overlapping outliers with another true cluster as a big community and underestimates the number of clusters.

The proposed outlier detection method has been implemented with the following parameters. The matrix of centroids $\hat{\mathbf{M}}$ is created for estimated $\hat{K} = 5$ object clusters that include $N = 800$

observations. $N_s = 40$ samples for each centroid are used to form dictionary matrix \mathbf{H} . In total, the dictionary matrix \mathbf{H} includes 200 observations for $\hat{K} = 5$ object clusters. The remaining data matrix \mathbf{X} is used for creating the test sample matrix \mathbf{X}_{test} with equal number of observations by random selection. The extraction of the degrees has been performed $N_t = 300$ times, so as to achieve degrees of all test vectors in data matrix \mathbf{X} . For the ℓ_1 -Graph clustering, the estimated outlier-free data set $\check{\mathbf{X}}$ is equal and randomly separated as dictionary matrix $\check{\mathbf{H}}$ and the test matrix $\check{\mathbf{X}}_{\text{test}}$. Tukey's distance function [ZKO18] where the threshold is defined as $T_{\text{Tukey}} = 4.68$ for 95 percent asymptotic relative efficiency (ARE) is used for initialization with K -medoids in the proposed algorithm.

The average correct clustering results of four competitor clustering methods for a given number of clusters $K = 5$ and the clustering result of the proposed method for the estimated number of clusters $\hat{K} = 5$ are shown in Tab. 4.11. All results presented in this section were obtained using 300 runs. We can see that clustering with the t distribution-based EM algorithm shows better performance than the conventional Gaussian EM one although both clustering algorithms are not able to cope with the two group behaviour of L2. The results show that the application of Louvain solves this problem. However, the average correct clustering result of Louvain based clustering indicates that there is not a noticeable difference in average clustering rate compared to the t -distribution based EM algorithm in average clustering rate of human gait micro Doppler radar data. Furthermore, the application of ℓ_1 -Graph solves the two group behaviour of L2 and gives reasonably good clustering results. Nevertheless, as it can be seen in the table, the correct clustering rate for CW is limited to 71% and also there is a trade-off between NW and CW. Combining classical outlier detection methods [RB18, KSZ08, BS03] with the ℓ_1 -Graph method decreased the clustering performance because outliers could not be correctly identified. Results for combining [RB18] with the ℓ_1 -Graph method are reported in Tab. 4.11.

Considering the results of the proposed method, the outlier detection algorithm increases the correct clustering rate of ℓ_1 -graph by approximately 7%, 3%, 4% and 10% for NW, L1, L2 and CW clusters, respectively, while the correct clustering rate of the CW_{OOS} cluster drops by 2%. This can be explained by the trade-off between correct clustering rates of the CW_{OOS} and other groups. The CW_{OOS} cluster has a completely different group behavior and it results in less nonzero coefficients compared to the other groups. Thus, the clipping based on the overall group behavior causes a small drop in this cluster. On average, the proposed outlier detection approach increases the overall correct clustering rate of ℓ_1 -Graph by 4.5%. Overall, the proposed method clusters the gait data in an unsupervised manner and achieves a correct clustering rate of 92.8%. In comparison to state-

of-the-art parameter-free clustering algorithms, it estimates the cluster number correctly as $\hat{K} = 5$ and provides robustness with a meaningful partition of the data.

4.4.3.5 CONCLUSIONS

In [TMZ20], we proposed an unsupervised graph-based robust clustering algorithm to cluster highly contaminated radar data efficiently. The method shows applicability of sparse regression and graph models on clustering of human gait signatures, providing robustness to noise and sparse outlying entries. The clustering results of human gait signatures showed that the proposed method outperforms existing parameter-free clustering approaches both in the sense of cluster enumeration and partition. Overall, the unsupervised graph-based robust clustering approach shows promising performance on clustering of micro-Doppler radar-based human gait signatures.

Learn from yesterday, live for today and hope for tomorrow. The important thing is not to stop questioning.

—Albert Einstein

5

Conclusion and Outlook

5.1 SUMMARY AND CONCLUSION

This dissertation contributes to robust graph clustering by developing fast, robust and parameter-free graph clustering methods. Considering the effect of outliers on sparse graph construction, the main goal has been designing graph clustering algorithms that jointly address robustness and sparsity.

Since there does not exist a single definition of an optimal graph model for clustering, the first part of the thesis comprised the definition of a sparse graph model and its applicability to sparsity-aware graph clustering. In particular, firstly, the sparse graph model including edges for only intra-cluster associations has been defined by using the advantageous nature of block diagonally structured affinity matrices. Then, motivated by the various applications of eigenvalues and the eigenvectors, a spectral analysis has been conducted based on the eigen-decomposition of the Laplacian matrix associated with the sparse graph. To reduce the computational cost of Laplacian matrix analysis, a vector representing the blocks as a piece-wise linear function of similarity coefficients has been defined. The adaptiveness of obtained spectral properties to sparsity-aware clustering has been shown by formulating graph construction problems as an approximations to

spectral properties of sparse graphs. To demonstrate this argument, two different sparsity-aware BDR methods built upon, respectively, the eigenvalues and the eigenvectors associated with sparse graph have been proposed.

To understand how to best incorporate robustness in sparsity-aware graph clustering, fundamental outlier types have been defined and their effects on spectral properties of sparse graphs have been analyzed. In particular, outliers' effects on sparse graphs have been extensively studied in terms of the affinity matrix, overall edge weights, eigenvalues, eigenvectors and, finally, in terms of a proposed simplified Laplacian matrix analysis. Based on the obtained results regarding the outliers' effect on affinity matrix, the SPARCODE method that shrinks the undesired similarity coefficients associated with outliers to zero has been proposed. Next, FRS-BDR approach building upon outliers' effects on the eigenvalues and the simplified Laplacian matrix analysis have been presented. Different from these affinity matrix construction solutions, RRLPI and RLPFM algorithms that are robust against outliers' effects on the eigenvectors have been introduced. Lastly, motivated by the outliers' effects on the overall edge weights, an outlier detection method using node degree as an outlyingness measure has been proposed and applied to gait analysis.

Real-world applicability of proposed robust graph-clustering methods has been shown for different aspects. For example, the obtained promising results on person identification based on gait signatures, face and handwritten digit recognition has demonstrated the applicability of robust-graph clustering in biometrics. Medical diagnosis is another important application of robust graph clustering which has been detailed for varying experiments on, i.e., Gait, Parkinson A., Diabetic Retinopathy, Cardiotocography data sets. In addition to these well-established data bases, the efficiency of robust graph clustering in image segmentation encourages further applications, e.g., autonomous vehicles, analysing satellite images, medical imaging.

5.2 FUTURE RESEARCH DIRECTIONS

This section presents some possible extensions of the theoretical analysis and proposed robust graph clustering algorithms that have been detailed in Chapters 3 and 4.

5.2.1 ASSUMPTIONS ON SPARSE GRAPH MODEL

In Section 3.1, we have assumed that each block is concentrated around a similarity constant. Then, this assumption has been relaxed in Section 3.2.3 based on the eigenvectors of the BD nonnegative definite Laplacian matrix whose similarity coefficients in blocks are random variables. Even though

this relaxation makes the sparse graph model more realistic, achieving this strictly BD structure may be challenging in real-world scenarios.

SBM is a random graph model that is widely used for clustering [Abb17]. In simple words, every vertex is associated with a cluster and there are undirected edges between vertices based on probabilities that are function of vertices' group memberships [KN11]. Based on this, the sparse graph model that has been defined in Section 3.1, can be considered as a weighted SBM in which the vertices of the same cluster are connected by an edge with probability one while that of different clusters are unconnected with zero probability. This also means that approximations to the sparse graph model can be made in a statistical sense and the assumptions on sparse graph can be relaxed by giving a further degree of freedom [AWF92].

5.2.2 FUNDAMENTAL OUTLIER TYPES IN RANDOM GRAPHS

The earliest theoretical models of a network have been studied in the 1950s and 1960 by Paul Erdős and Alfred Rényi and they generalized to models of web graphs, social networks, biological networks [ZR15, New03]. Motivated by this, the analysis of fundamental outlier types in random graphs is an interesting research direction to design robust graph clustering algorithms that are well-suited to the modelling of real-world networks.

Again referring to SBM [Abb17], which is one of the most commonly used random graph models for clustering, the fundamental outlier types can be determined based on the probability of sharing an edge for the vertices of same cluster. More precisely, the degree of a vertex is an informative measure of outlyingness when the probability of edge existence for the vertices of the same cluster become comparable, or ideally constant such as in the so-called planted partition model. Therefore, the determination of fundamental outlier types can be adapted to random graphs by systematically analyzing and understanding the underlying edge existence probabilities.

5.2.3 ROBUST GRAPH-BASED CLUSTERING FOR LARGE GRAPHS

In real-world scenarios, graphs consist of large numbers of vertices, thing e.g. of the Facebook social network. As it has been highlighted in Section 3.2.4 for large graphs with considerable number of vertices, analysing $N \times N$ matrices is challenging or even unapplicable. In particular, the eigen-decomposition of Laplacian matrix requires noticeable computation time which makes spectral methods unapplicable, especially, for the densely connected large graph structures. In such cases, alternative solutions, such as, transforming the analysis into a vector space as in Section 3.2.4 become crucial. However, the vector \mathbf{v} definition in Section 3.2.4 considers densely

connected clusters which may not always apply to real-world graphs. Additionally, BD ordering and approximating vector \mathbf{v} for many different candidate block sizes as in Sections 3.4.1 and 4.4.1.2 may result in a large computation time for large graphs. Therefore, the adaptation of simplified Laplacian matrix analysis to large graphs is still an open problem and extracting informative subgraphs, such as in [KKV15] is an alternative way for analysing large networks.

5.2.4 TIME-SERIES ANALYSIS APPLICATIONS BASED ON VISIBILITY GRAPHS

An alternative way of analyzing time series is visibility graphs which maps time series into a network according to the visibility criterion that has been detailed in [LLB08]. In recent years, the analysis of time series based on visibility graphs has attracted great interest, e.g., [KM22, SGY15] and, in particular, horizontal visibility graphs are popular tools due to their geometrically simpler and analytically solvable fashion in comparison to the former algorithm in [LLB08]. In horizontal visibility graphs, every vertex represents a datum in the time series and the vertices are connected if their corresponding data heights are larger than all of the data heights that are located between them. According to this simple procedure, a time series can be transformed into an undirected graph and graph clustering can be performed, for example, to capture periodic (or cyclic) time sequences. In terms of periodicity, the size of clusters may provide information about the outlyingness. For instance, when a noisy peak occurs in time series it might obscure the neighboring data heights and lead to a deterioration in the visibility graph structure and thus, graph clustering may produce imbalanced clusters. Therefore, analysing visibility graphs in noisy scenarios and developing robust graph clustering methods that are applicable to visibility graphs is of interest for future work.

A

Proofs and Additional Theoretical Information

Appendix A is organized as follows. In Section A.1, the theorems by reference to spectral analysis of sparse graph model are proved based on the generalized and standard eigen-decompositions, respectively. Similarly, the theorems regarding the outlier effects on sparse graphs are proved in Section A.2. The outliers' effects on the Fiedler vector is the subject of Section A.3. Lastly, the theorems that analyzes the RRLPI method are detailed in Section A.3 and the auxiliary information that is used for theoretical analysis is given in Section A.5.

where $\mathbf{1} \in \mathbb{Z}^{N_i}$ denotes a column vector of ones. Substituting $\mathbf{1}^\top \mathbf{1} = N_i$ in $\det(\mathbf{L}_i - \lambda^{(i)} \mathbf{D}_i) = 0$ leads to

$$\begin{aligned} \left(1 - \frac{N_i w_i}{N_i w_i - \lambda^{(i)} (N_i - 1) w_i}\right) (N_i w_i - \lambda^{(i)} (N_i - 1) w_i)^{N_i} &= 0 \\ \left(\frac{-\lambda^{(i)} (N_i - 1) w_i}{N_i w_i - \lambda^{(i)} (N_i - 1) w_i}\right) (N_i w_i - \lambda^{(i)} (N_i - 1) w_i)^{N_i} &= 0 \\ (-\lambda^{(i)} (N_i - 1) w_i) (N_i w_i - \lambda^{(i)} (N_i - 1) w_i)^{N_i - 1} &= 0 \end{aligned}$$

For $w_i > 0$ and $N_i > 1$, the eigenvalues are given by

$$\lambda_0^{(i)} = 0 \quad \text{and} \quad \lambda_{1, \dots, N_i - 1}^{(i)} = \frac{N_i}{N_i - 1}, \quad i = 1, \dots, K.$$

□

A.1.1.2 PROOF OF THEOREM 2

Based on the information that the K smallest eigenvalues of the Laplacian matrix associated with the BD affinity matrix are zero-valued [Lux07], the associated orthonormal set of eigenvectors yields for the both eigen-decompositions in Eqs. (2.5) and (2.6), i.e.,

$$\begin{aligned} \mathbf{y}_0 &= \left[\overbrace{\pm \sqrt{1/N_1}, \dots, \pm \sqrt{1/N_1}}^{N_1}, \overbrace{0, \dots, 0}^{N_2}, \dots, \overbrace{0, \dots, 0}^{N_K} \right]^\top \\ \mathbf{y}_1 &= \left[0, \dots, 0, \overbrace{\pm \sqrt{1/N_2}, \dots, \pm \sqrt{1/N_2}}^{N_2}, \dots, 0, \dots, 0 \right]^\top \\ &\vdots \\ \mathbf{y}_{K-1} &= \left[0, \dots, 0, 0, \dots, 0, \dots, \overbrace{\pm \sqrt{1/N_K}, \dots, \pm \sqrt{1/N_K}}^{N_K} \right]^\top \end{aligned}$$

where $\mathbf{y}_k \in \mathbb{R}^N$ is the eigenvector associated with the k th zero-valued eigenvalue. The Euclidean distance between any embedding vector pairs \mathbf{e}_i and \mathbf{e}_j associated to distinct blocks k and l is equal to $\|\mathbf{e}_i - \mathbf{e}_j\|_2 = \sqrt{1/N_k + 1/N_l}$ for $k \neq l$ and $i \neq j$. □

A.1.1.3 PROOF OF THEOREM 3

By definition, the vector \mathbf{v} is computed by summing up the rows of the upper triangular part of \mathbf{L} , i.e.,

$$\mathbf{v} = [0, w_1, \dots, d_1, 0, w_2, \dots, d_2, \dots, 0, w_k, \dots, d_k],$$

where $d_i = (N_i - 1)w_i$, $i = 1, \dots, K$. Since each block includes $N_i \in \{N_1, N_2, \dots, N_K\}$ number of nodes the vectors containing lower and upper limits can be defined as follows

$$\boldsymbol{\ell} = \left[1, N_1 + 1, \dots, \sum_{i=1}^{K-1} N_i + 1 \right], \quad \mathbf{u} = \left[N_1, N_1 + N_2, \dots, \sum_{i=1}^K N_i \right].$$

Substituting each m with $m = 1, \dots, N$ in the function $f(m)$ yields the vector:

$$\mathbf{v} = [0, w_1, \dots, (N_1 - 1)w_1, \dots, 0, w_K, \dots, (N_K - 1)w_K],$$

which concludes the proof that two vectors are identical. \square

A.1.2 STANDARD EIGEN-DECOMPOSITION BASED ANALYSIS

Theorem. 1.S. Let $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a sparse affinity matrix in Definition 3.1.1 and $\mathbf{D} \in \mathbb{R}^{N \times N}$ is the associated matrix of overall edge weights. Assuming that $\mathbf{L} \in \mathbb{R}^{N \times N}$ denotes the associated sparse Laplacian matrix, its eigenvalues will be of the following form based on Eq. (2.5)

$$\boldsymbol{\lambda} = \text{sort} \left(\underbrace{0, \dots, 0}_K, \underbrace{N_1 w_1, \dots, N_1 w_1}_{N_1 - 1}, \dots, \underbrace{N_K w_K, \dots, N_K w_K}_{N_K - 1} \right),$$

where $\boldsymbol{\lambda} \in \mathbb{R}^N$ denotes the vector of target eigenvalues and $\text{sort}(\cdot)$ is sorting operation in ascending order.

Let $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a zero-diagonal K block affinity matrix with corresponding Laplacian $\mathbf{L} \in \mathbb{R}^{N \times N}$ as in Section A.1.1.1. To compute the eigenvalues in Eq. (2.5), $\det(\mathbf{L} - \lambda \mathbf{I}) = 0$ is considered which can equivalently be written using the determinant properties of block matrices (for details, see Section 2 in [Sil00]), as follows

$$\det(\mathbf{L} - \lambda \mathbf{I}) = \prod_{i=1}^K \det(\mathbf{L}_i - \lambda^{(i)} \mathbf{I}) = 0,$$

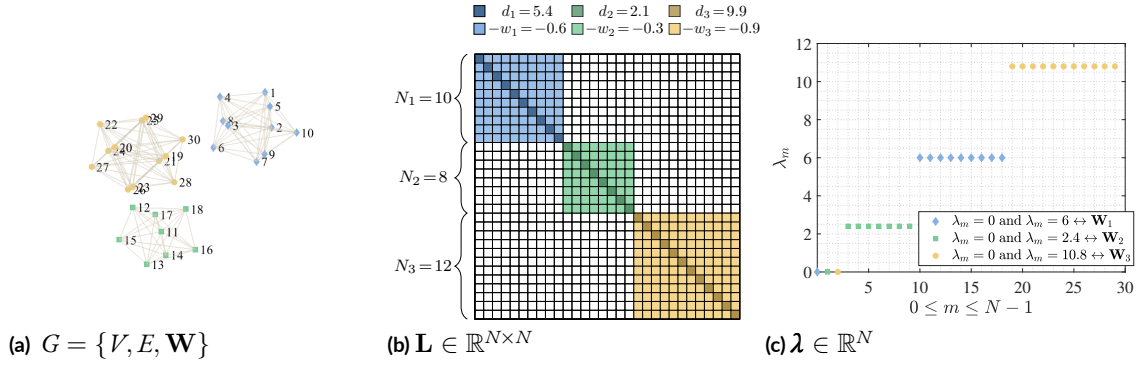


Figure A.1: Exemplary illustration of Theorem 1.S. ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$).

where $\mathbf{L}_i \in \mathbb{R}^{N_i \times N_i}$ and $\lambda^{(i)}$, $i = 1, \dots, K$, denote \mathbf{L} and λ associated with the i th block, respectively. Further, $\mathbf{L}_i - \lambda^{(i)} \mathbf{I}$, $i = 1, \dots, K$ can alternatively be written as

$$\underbrace{\begin{bmatrix} c_i & -w_i & \dots & -w_i \\ -w_i & c_i & \dots & -w_i \\ \vdots & \vdots & \ddots & \vdots \\ -w_i & -w_i & \dots & c_i \end{bmatrix}}_{\mathbf{L}_i - \lambda^{(i)} \mathbf{I}} = \underbrace{\begin{bmatrix} c_i + w_i & 0 & \dots & 0 \\ 0 & c_i + w_i & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & c_i + w_i \end{bmatrix}}_{\mathbf{H}} + \underbrace{\begin{bmatrix} \sqrt{w_i} \\ \sqrt{w_i} \\ \vdots \\ \sqrt{w_i} \end{bmatrix}}_{\mathbf{u}} \underbrace{\begin{bmatrix} -\sqrt{w_i} & -\sqrt{w_i} & \dots & -\sqrt{w_i} \end{bmatrix}}_{\mathbf{v}^\top}$$

with $c_i = (N_i - 1)w_i - \lambda^{(i)}$. For an invertible matrix $\mathbf{H} \in \mathbb{R}^{N_i \times N_i}$ such that $\mathbf{H}^\dagger = (c_i + w_i)^{-1} \mathbf{I}$, the matrix determinant lemma (for details, see Lemma 1.1 in [DZ07]) computes the determinant as $\det(\mathbf{H} + \mathbf{u}\mathbf{v}^\top) = (1 + \mathbf{v}^\top \mathbf{H}^\dagger \mathbf{u}) \det(\mathbf{H})$ where $\mathbf{u} \in \mathbb{R}^{N_i}$ and $\mathbf{v} \in \mathbb{R}^{N_i}$ are two column vectors. Thus, it holds that

$$\det(\mathbf{L}_i - \lambda^{(i)} \mathbf{D}_i) = \left(1 + (-\sqrt{w_i} \mathbf{1})^\top \left(\frac{\sqrt{w_i}}{c_i + w_i} \mathbf{1} \right) \right) (c_i + w_i)^{N_i}.$$

Substituting $\mathbf{1}^\top \mathbf{1} = N_i$ in $\det(\mathbf{L}_i - \lambda^{(i)} \mathbf{I}) = 0$ leads to

$$\begin{aligned} \left(1 - \frac{N_i w_i}{N_i w_i - \lambda^{(i)}} \right) (N_i w_i - \lambda^{(i)})^{N_i} &= 0 \\ \left(\frac{-\lambda^{(i)}}{N_i w_i - \lambda^{(i)}} \right) (N_i w_i - \lambda^{(i)})^{N_i} &= 0 \\ -\lambda^{(i)} (N_i w_i - \lambda^{(i)})^{N_i - 1} &= 0 \end{aligned}$$

For $w_i > 0$ and $N_i > 1$, the eigenvalues are given by

$$\lambda_0^{(i)} = 0 \quad \text{and} \quad \lambda_{1, \dots, N_i - 1}^{(i)} = N_i w_i, \quad i = 1, \dots, K.$$

□

A.2 OUTLIER EFFECTS ON SPARSE GRAPH MODEL

A.2.1 GENERALIZED EIGEN-DECOMPOSITION BASED ANALYSIS

A.2.1.1 PROOF OF THEOREM 4

Let $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ denote the Laplacian matrix associated with a block zero-diagonal symmetric affinity matrix for K blocks with an additional Type II outlier that is correlated with all blocks, i.e.,

$$\tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_{\Pi} & -\tilde{w}_{\Pi,1} & -\tilde{w}_{\Pi,1} & \dots & -\tilde{w}_{\Pi,1} & -\tilde{w}_{\Pi,2} & -\tilde{w}_{\Pi,2} & \dots & -\tilde{w}_{\Pi,2} & \dots & -\tilde{w}_{\Pi,K} & -\tilde{w}_{\Pi,K} & \dots & -\tilde{w}_{\Pi,K} \\ -\tilde{w}_{\Pi,1} & \tilde{d}_1 & -w_1 & \dots & -w_1 & & & & & & & & & & \\ -\tilde{w}_{\Pi,1} & -w_1 & \tilde{d}_1 & \dots & -w_1 & & & & & & & & & & \\ \vdots & \vdots & \vdots & \ddots & & & & & & & & & & & \\ -\tilde{w}_{\Pi,1} & -w_1 & -w_1 & \dots & \tilde{d}_1 & & & & & & & & & & \\ -\tilde{w}_{\Pi,2} & & & & & \tilde{d}_2 & -w_2 & \dots & -w_2 & & & & & & \\ -\tilde{w}_{\Pi,2} & & & & & -w_2 & \tilde{d}_2 & \dots & -w_2 & & & & & & \\ \vdots & & & & & \vdots & \vdots & \ddots & & & & & & & \\ -\tilde{w}_{\Pi,2} & & & & & -w_2 & -w_2 & \dots & \tilde{d}_2 & & & & & & \\ \vdots & & & & & & & & & \ddots & & \vdots & & \vdots & \\ -\tilde{w}_{\Pi,K} & & & & & & & & & & \tilde{d}_K & -w_K & \dots & -w_K & \\ -\tilde{w}_{\Pi,K} & & & & & & & & & & -w_K & \tilde{d}_K & \dots & -w_K & \\ \vdots & & & & & & & & & & \vdots & \vdots & \ddots & & \\ -\tilde{w}_{\Pi,K} & & & & & & & & & & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix}$$

where $\tilde{d}_{\Pi} = \sum_{j=1}^K N_j \tilde{w}_{\Pi,j}$ and $\tilde{d}_j = (N_j - 1)w_j + \tilde{w}_{\Pi,j}$ such that $j = 1, \dots, K$. To compute the eigenvalues of the Laplacian matrix $\tilde{\mathbf{L}}$, $\det(\tilde{\mathbf{L}} - \tilde{\lambda}\tilde{\mathbf{D}}) = 0$ is considered. To simplify this determinant, (for details, see Lemma 1.1 in [DZ07]) can be generalized as follows ¹

$$\det(\mathbf{H} + \mathbf{UV}^{\top}) = \det(\mathbf{H})\det(\mathbf{I} + \mathbf{V}^{\top}\mathbf{H}^{\dagger}\mathbf{U}),$$

where $\mathbf{H} \in \mathbb{R}^{(N+1) \times (N+1)}$ denotes an invertible matrix, \mathbf{I} is the identity matrix and $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{(N+1) \times (N+1)}$. Then, for $\det(\tilde{\mathbf{L}} - \tilde{\lambda}\tilde{\mathbf{D}}) = \det(\mathbf{H} + \mathbf{UV}^{\top}) = 0$, it follows that

¹For a detailed information, see Section A.5.1.

where $z_{\Pi} = \sum_{j=1}^K N_j \tilde{w}_{\Pi,j} - \tilde{\lambda} \sum_{j=1}^K N_j \tilde{w}_{\Pi,j}$ and $z_j = N_j w_j + \tilde{w}_{\Pi,j} - \tilde{\lambda}((N_j - 1)w_j + \tilde{w}_{\Pi,j})$ for $j = 1, \dots, K$. Using the determinant properties of block matrices (for details, see Section 2 in [Sil00]), it holds that

$$0 = \begin{vmatrix} 1 & -N_1 \tilde{w}_{\Pi,1} z_1^{-1} & -N_2 \tilde{w}_{\Pi,2} z_2^{-1} & \dots & -N_K \tilde{w}_{\Pi,K} z_K^{-1} \\ -\tilde{w}_{\Pi,1} z_{\Pi}^{-1} & -N_1 w_1 z_1^{-1} + 1 & 0 & \dots & 0 \\ -\tilde{w}_{\Pi,2} z_{\Pi}^{-1} & 0 & -N_2 w_2 z_2^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{\Pi,K} z_{\Pi}^{-1} & 0 & \dots & \dots & -N_K w_K z_K^{-1} + 1 \end{vmatrix} \det(\mathbf{H}).$$

To simplify the determinant of the first matrix, it transformed into a lower diagonal matrix by applying the following Gaussian elimination steps

$$\begin{aligned} & \frac{N_1 \tilde{w}_{\Pi,1} z_1^{-1}}{-N_1 w_1 z_1^{-1} + 1} R_2 + R_1 \rightarrow R_1 \\ & \frac{N_2 \tilde{w}_{\Pi,2} z_2^{-1}}{-N_2 w_2 z_2^{-1} + 1} R_3 + R_1 \rightarrow R_1 \\ & \quad \vdots \\ & \frac{N_K \tilde{w}_{\Pi,K} z_K^{-1}}{-N_K w_K z_K^{-1} + 1} R_{K+1} + R_1 \rightarrow R_1 \end{aligned}$$

where R_K denotes K th row. Then, the simplified determinant of the first matrix yields

$$0 = c_{\Pi} \left(\prod_{i=1}^K (-N_i w_i z_i^{-1} + 1) \right) z_{\Pi} z_1^{N_1} z_2^{N_2} \dots z_K^{N_K}$$

where

$$c_{\Pi} = \left(1 - \sum_{i=1}^K \left(\frac{N_K \tilde{w}_{\Pi,i}^2 z_i^{-1} z_{\Pi}^{-1}}{-N_i w_i z_i^{-1} + 1} \right) \right).$$

For $z_{\Pi} = \sum_{j=1}^K N_j \tilde{w}_{\Pi,j} - \tilde{\lambda} \sum_{j=1}^K N_j \tilde{w}_{\Pi,j}$ and $z_j = N_j w_j + \tilde{w}_{\Pi,j} - \tilde{\lambda}((N_j - 1)w_j + \tilde{w}_{\Pi,j})$ such that $j = 1, \dots, K$ the determinant yields

$$\begin{aligned}
0 &= \left(\prod_{i=1}^K (-N_i w_i z_i^{-1} + 1) \right) z_{\text{II}} \left(\prod_{j=1}^K z_j^{N_j} \right) z_{\text{II}}^{-1} \left(z_{\text{II}} - \sum_{k=1}^K \frac{N_k \tilde{w}_{\text{II},k}^2 z_k^{-1}}{-N_k w_k z_k^{-1} + 1} \right) \\
0 &= \prod_{i=1}^K (\tilde{w}_{\text{II},i} - \tilde{\lambda} \tilde{d}_i) \prod_{j=1}^K (N_j w_j + \tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j)^{N_j-1} \left(-\tilde{\lambda} \tilde{d}_{\text{II}} + \sum_{k=1}^K N_k \tilde{w}_{\text{II},k} - \sum_{l=1}^K \frac{N_l \tilde{w}_{\text{II},l}^2}{\tilde{w}_{\text{II},l} - \tilde{\lambda} \tilde{d}_l} \right) \\
0 &= \prod_{i=1}^K (\tilde{w}_{\text{II},i} - \tilde{\lambda} \tilde{d}_i) \prod_{j=1}^K (N_j w_j + \tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j)^{N_j-1} \left(-\tilde{\lambda} \tilde{d}_{\text{II}} + \sum_{k=1}^K \left(N_k \tilde{w}_{\text{II},k} - \frac{N_k \tilde{w}_{\text{II},k}^2}{\tilde{w}_{\text{II},k} - \tilde{\lambda} \tilde{d}_k} \right) \right) \\
0 &= \prod_{i=1}^K (\tilde{w}_{\text{II},i} - \tilde{\lambda} \tilde{d}_i) \prod_{j=1}^K (N_j w_j + \tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j)^{N_j-1} \left(-\tilde{\lambda} \tilde{d}_{\text{II}} - \sum_{k=1}^K \frac{N_k \tilde{w}_{\text{II},k} \tilde{\lambda} \tilde{d}_k}{\tilde{w}_{\text{II},k} - \tilde{\lambda} \tilde{d}_k} \right) \\
0 &= \tilde{\lambda} \prod_{i=1}^K (\tilde{w}_{\text{II},i} - \tilde{\lambda} \tilde{d}_i) \prod_{j=1}^K (N_j w_j + \tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j)^{N_j-1} \left(-\tilde{d}_{\text{II}} - \sum_{k=1}^K \frac{N_k \tilde{w}_{\text{II},k} \tilde{d}_k}{\tilde{w}_{\text{II},k} - \tilde{\lambda} \tilde{d}_k} \right)
\end{aligned}$$

Now, the $N + 1 - K$ number of eigenvalues can be written as

$$\left\{ \begin{array}{l} N_1 - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_1 w_1 + \tilde{w}_{\text{II},1}}{\tilde{d}_1}, \\ N_2 - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_2 w_2 + \tilde{w}_{\text{II},2}}{\tilde{d}_2}, \\ \vdots \\ N_K - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_K w_K + \tilde{w}_{\text{II},K}}{\tilde{d}_K}, \\ \text{the smallest element of } \tilde{\lambda} \text{ is equal to zero,} \end{array} \right.$$

and the remaining K eigenvalues are the roots of

$$\prod_{j=1}^K (\tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j) \left(-\sum_{j=1}^K \frac{N_j \tilde{w}_{\text{II},j} \tilde{d}_j}{\tilde{w}_{\text{II},j} - \tilde{\lambda} \tilde{d}_j} - \tilde{d}_{\text{II}} \right) = 0,$$

where $\tilde{d}_{\text{II}} = \sum_{j=1}^K N_j \tilde{w}_{\text{II},j}$ and $\tilde{d}_j = (N_j - 1)w_j + \tilde{w}_{\text{II},j}$. □

A.2.1.2 PROOF OF THEOREM 5

Let $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ denote the Laplacian matrix associated with K block zero diagonal affinity matrix in which i th block has similarity with remaining $K - 1$ number of blocks. For simplicity, let $i = 1$,

$$\text{i.e.,} \quad \tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 & -w_2 & \dots & -w_2 & & & & \\ -\tilde{w}_{1,2} & & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 & \dots & -w_2 & & & & \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots & & & & \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 & & & & \\ \vdots & & \vdots & \vdots & \vdots & & \vdots & & & & \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \vdots & \vdots & & \vdots & \ddots & \vdots & \vdots & \\ -\tilde{w}_{1,K} & & -\tilde{w}_{1,K} & \dots & -w_K & \tilde{d}_K & -w_K & \dots & -w_K & \dots & \\ \vdots & & \vdots & & -w_K & \tilde{d}_K & \dots & \ddots & \vdots & \vdots & \\ -\tilde{w}_{1,K} & & -\tilde{w}_{1,K} & & -w_K & -w_K & \dots & \tilde{d}_K & & & \end{bmatrix}$$

where $\tilde{d}_1 = (N_1 - 1)w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j}$ and $\tilde{d}_j = (N_j - 1)w_j + N_1 \tilde{w}_{1,j}, j = 2, \dots, K$. To estimate the eigenvalues of the Laplacian matrix $\tilde{\mathbf{L}}$, $\det(\tilde{\mathbf{L}} - \tilde{\lambda} \tilde{\mathbf{D}}) = 0$ is considered which can equivalently be written in matrix form as follows

$$\begin{bmatrix} \tilde{d}_1 - \tilde{\lambda} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 - \tilde{\lambda} \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \vdots & & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 - \tilde{\lambda} \tilde{d}_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 - \tilde{\lambda} \tilde{d}_2 & -w_2 & \dots & -w_2 & & & & \\ -\tilde{w}_{1,2} & & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 - \tilde{\lambda} \tilde{d}_2 & \dots & -w_2 & & & & \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots & & & & \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 - \tilde{\lambda} \tilde{d}_2 & & & & \\ \vdots & & \vdots & \vdots & \vdots & & \vdots & & & & \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \vdots & \vdots & & \vdots & \ddots & \vdots & \vdots & \\ -\tilde{w}_{1,K} & & -\tilde{w}_{1,K} & \dots & -w_K & \tilde{d}_K - \tilde{\lambda} \tilde{d}_K & -w_K & \dots & -w_K & \dots & \\ \vdots & & \vdots & & -w_K & \tilde{d}_K - \tilde{\lambda} \tilde{d}_K & \dots & \ddots & \vdots & \vdots & \\ -\tilde{w}_{1,K} & & -\tilde{w}_{1,K} & & -w_K & -w_K & \dots & \tilde{d}_K - \tilde{\lambda} \tilde{d}_K & & & \end{bmatrix} = 0$$

To simplify this determinant, the matrix determinant lemma (for details, see Lemma 1.1 in [DZ07]) can be generalized as follows²

$$\det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = \det(\mathbf{H})\det(\mathbf{I} + \mathbf{V}^\top \mathbf{H}^{-1} \mathbf{U})$$

where $\mathbf{H} \in \mathbb{R}^{(N+1) \times (N+1)}$ denotes an invertible matrix, \mathbf{I} is the identity matrix and $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{(N+1) \times (N+1)}$. Then, for $\det(\tilde{\mathbf{L}} - \tilde{\lambda} \tilde{\mathbf{D}}) = \det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = 0$, it follows that

²For a detailed information, see Section A.5.1.

where $z_1 = N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j} - \tilde{\lambda}((N_1 - 1)w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j})$ and $z_j = N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda}((N_j - 1)w_j + N_1 \tilde{w}_{1,j})$ with $j = 2, \dots, K$. Using determinant properties of block matrices (for details, see Section 2 in [Sil00]), it holds that

$$0 = \det(\mathbf{H}) \begin{vmatrix} -N_1 w_1 z_1^{-1} + 1 & -N_2 \tilde{w}_{1,2} z_2^{-1} & -N_3 \tilde{w}_{1,3} z_3^{-1} & \dots & -N_K \tilde{w}_{1,K} z_K^{-1} \\ -N_1 \tilde{w}_{1,2} z_1^{-1} & -N_2 w_2 z_2^{-1} + 1 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,3} z_1^{-1} & 0 & -N_3 w_3 z_3^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -N_1 \tilde{w}_{1,K} z_1^{-1} & 0 & \dots & \dots & -N_K w_K z_K^{-1} + 1 \end{vmatrix}.$$

To simplify the determinant of the second matrix, it transformed into a lower diagonal matrix by applying the following Gaussian elimination steps

$$\begin{aligned} \frac{N_2 \tilde{w}_{1,2} z_2^{-1}}{-N_2 w_2 z_2^{-1} + 1} R_2 + R_1 &\rightarrow R_1 \\ \frac{N_3 \tilde{w}_{1,3} z_3^{-1}}{-N_3 w_3 z_3^{-1} + 1} R_3 + R_1 &\rightarrow R_1 \\ &\vdots \\ \frac{N_K \tilde{w}_{1,K} z_K^{-1}}{-N_K w_K z_K^{-1} + 1} R_K + R_1 &\rightarrow R_1 \end{aligned}$$

where R_K denotes the K th row. Then, the simplified determinant yields

$$0 = \det(\mathbf{H}) \begin{vmatrix} c_1 & 0 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,2} z_1^{-1} & -N_2 w_2 z_2^{-1} + 1 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,3} z_1^{-1} & 0 & -N_3 w_3 z_3^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -N_1 \tilde{w}_{1,K} z_1^{-1} & 0 & \dots & \dots & -N_K w_K z_K^{-1} + 1 \end{vmatrix}$$

where c_1 is equal to

$$c_1 = -N_1 w_1 z_1^{-1} + 1 - \sum_{i=2}^K \frac{N_i \tilde{w}_{1,i} z_i^{-1} N_1 \tilde{w}_{1,i} z_1^{-1}}{-N_i w_i z_i^{-1} + 1}.$$

For $z_1 = N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j} - \tilde{\lambda}((N_1 - 1)w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j})$ and $z_j = N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda}((N_j - 1)w_j + N_1 \tilde{w}_{1,j})$ with $j = 2, \dots, K$, the determinant $\det(\tilde{\mathbf{L}} - \tilde{\lambda} \tilde{\mathbf{D}}) = 0$ yields

$$\begin{aligned}
0 &= \prod_{i=1}^K z_i^{N_i} \prod_{j=2}^K (-N_j w_j z_j^{-1} + 1) c_1 \\
0 &= z_1^{N_1} \prod_{i=2}^K z_i^{N_i-1} \prod_{j=2}^K (-N_j w_j + z_j) c_1 \\
0 &= z_1^{N_1} \prod_{i=2}^K z_i^{N_i-1} \prod_{j=2}^K (-N_j w_j + z_j) z_1^{-1} \left(-N_1 w_1 + z_1 - \sum_{k=2}^K \frac{N_1 N_k \tilde{w}_{1,k}^2 z_k^{-1}}{-N_k w_k z_k^{-1} + 1} \right) \\
0 &= \prod_{i=1}^K z_i^{N_i-1} \prod_{j=2}^K (-N_j w_j + z_j) \left(-N_1 w_1 + z_1 - \sum_{k=2}^K \frac{N_1 N_k \tilde{w}_{1,k}^2}{z_k - N_k w_k} \right) \\
0 &= \left(N_1 w_1 + \sum_{i=2}^K N_i \tilde{w}_{1,i} - \tilde{\lambda} \tilde{d}_1 \right)^{N_1-1} \prod_{j=2}^K \left(N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda} \tilde{d}_j \right)^{N_j-1} \prod_{k=2}^K (N_1 \tilde{w}_{1,k} - \tilde{\lambda} \tilde{d}_k) \\
&\quad \left(\sum_{l=2}^K N_l \tilde{w}_{1,l} - \tilde{\lambda} \tilde{d}_1 - \sum_{p=1}^K \frac{N_1 N_p \tilde{w}_{1,p}^2}{N_1 \tilde{w}_{1,p} - \tilde{\lambda} \tilde{d}_p} \right) \\
0 &= \left(N_1 w_1 + \sum_{i=2}^K N_i \tilde{w}_{1,i} - \tilde{\lambda} \tilde{d}_1 \right)^{N_1-1} \prod_{j=2}^K \left(N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda} \tilde{d}_j \right)^{N_j-1} \prod_{k=2}^K (N_1 \tilde{w}_{1,k} - \tilde{\lambda} \tilde{d}_k) \\
&\quad \left(-\tilde{\lambda} \tilde{d}_1 - \sum_{l=2}^K \left(N_l \tilde{w}_{1,l} - \frac{N_1 N_l \tilde{w}_{1,l}^2}{N_1 \tilde{w}_{1,l} - \tilde{\lambda} \tilde{d}_l} \right) \right) \\
0 &= \left(N_1 w_1 + \sum_{i=2}^K N_i \tilde{w}_{1,i} - \tilde{\lambda} \tilde{d}_1 \right)^{N_1-1} \prod_{j=2}^K \left(N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda} \tilde{d}_j \right)^{N_j-1} \prod_{k=2}^K (N_1 \tilde{w}_{1,k} - \tilde{\lambda} \tilde{d}_k) \\
&\quad \left(-\tilde{\lambda} \tilde{d}_1 - \sum_{l=2}^K \frac{\tilde{\lambda} \tilde{d}_l N_l \tilde{w}_{1,l}}{N_1 \tilde{w}_{1,l} - \tilde{\lambda} \tilde{d}_l} \right) \\
0 &= \tilde{\lambda} \left(N_1 w_1 + \sum_{i=2}^K N_i \tilde{w}_{1,i} - \tilde{\lambda} \tilde{d}_1 \right)^{N_1-1} \prod_{j=2}^K \left(N_j w_j + N_1 \tilde{w}_{1,j} - \tilde{\lambda} \tilde{d}_j \right)^{N_j-1} \prod_{k=2}^K (N_1 \tilde{w}_{1,k} - \tilde{\lambda} \tilde{d}_k) \\
&\quad \left(-\tilde{d}_1 - \sum_{l=2}^K \frac{\tilde{d}_l N_l \tilde{w}_{1,l}}{N_1 \tilde{w}_{1,l} - \tilde{\lambda} \tilde{d}_l} \right)
\end{aligned}$$

Based on this, $N + 1 - K$ number of eigenvalues are

$$\left\{ \begin{array}{l} N_i - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_i w_i + \sum_{\substack{j=1, \\ j \neq i}}^K N_j \tilde{w}_{i,j}}{\tilde{d}_i}, \\ N_j - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_j w_j + N_i \tilde{w}_{i,j}}{\tilde{d}_j}, \\ \vdots \\ N_K - 1 \text{ elements of } \tilde{\lambda} \text{ are equal to } \frac{N_K w_K + N_i \tilde{w}_{i,K}}{\tilde{d}_K}, \\ \text{the smallest element of } \tilde{\lambda} \text{ is equal to zero,} \end{array} \right.$$

and the remaining $K - 1$ eigenvalues in $\tilde{\lambda}$ are the roots of

$$\prod_{\substack{j=1 \\ j \neq i}}^K (N_i \tilde{w}_{i,j} - \tilde{\lambda} \tilde{d}_j) \left(- \sum_{\substack{j=1 \\ j \neq i}}^K \frac{\tilde{d}_j N_j \tilde{w}_{i,j}}{N_i \tilde{w}_{i,j} - \tilde{\lambda} \tilde{d}_j} - \tilde{d}_i \right) = 0,$$

where $\tilde{d}_j = (N_j - 1)w_j + N_i \tilde{w}_{i,j}$, $\tilde{d}_i = (N_i - 1)w_i + \sum_{\substack{j=1 \\ j \neq i}}^K N_j \tilde{w}_{i,j}$. □

A.2.1.3 PROOF OF PREPOSITION 4.2.1

Let $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ denote the Laplacian matrix associated with a block zero-diagonal symmetric affinity matrix of K blocks with an additional Type II outlier that is correlated with blocks i and j . For simplicity, $i = 1$ and $j = 2$. Further, let $\tilde{\mathbf{y}}_0 \in \mathbb{R}^{N+1}$ denote the eigenvector associated with zero-valued eigenvalues. Since Type II outlier do not affect the remaining $K - 2$ blocks, the eigenvectors associated with zero valued eigenvalues of these blocks can be written as follows:

$$\begin{aligned} \tilde{\mathbf{y}}_{0 \leftrightarrow \mathbf{w}_3} &= \left[\overbrace{0, 0, \dots, 0}^{N_1}, \overbrace{0, 0, 0, \dots, 0}^{\text{out}}, \overbrace{0, 0, 0, \dots, 0}^{N_2}, \overbrace{\frac{1}{\sqrt{N_3}}, \frac{1}{\sqrt{N_3}}, \dots, \frac{1}{\sqrt{N_3}}}^{N_3}, \dots, \overbrace{0, 0, \dots, 0}^{N_K} \right]^\top \\ &\vdots \\ \tilde{\mathbf{y}}_{0 \leftrightarrow \mathbf{w}_K} &= \left[\overbrace{0, 0, \dots, 0}^{N_1}, \overbrace{0, 0, 0, \dots, 0}^{\text{out}}, \overbrace{0, 0, 0, \dots, 0}^{N_2}, \overbrace{0, 0, \dots, 0}^{N_3}, \dots, \overbrace{\frac{1}{\sqrt{N_K}}, \frac{1}{\sqrt{N_K}}, \dots, \frac{1}{\sqrt{N_K}}}^{N_K} \right]^\top \end{aligned}$$

where $\tilde{\mathbf{y}}_{0 \leftrightarrow \mathbf{w}_k}$ denotes the eigenvector associated with the zero-valued eigenvalue of block \mathbf{W}_k for $k = 1, \dots, K$. Further, the eigenvector associated with zero-valued eigenvalue of the large block

can be written as

$$\tilde{\mathbf{y}}_{0 \leftrightarrow \mathbf{w}_{12}} = \left[\overbrace{\frac{1}{\sqrt{N_1 + N_2 + 1}}, \frac{1}{\sqrt{N_1 + N_2 + 1}}, \dots, \frac{1}{\sqrt{N_1 + N_2 + 1}}}^{N_1 + N_2 + 1}, \overbrace{0, 0, \dots, 0}^{N_3}, \dots, \overbrace{0, 0, \dots, 0}^{N_K} \right]^\top$$

Up to now, the eigenvectors associated with $K - 1$ number of eigenvalues are shown. Now, the next step is to examine the eigenvector associated with non-zero eigenvalue which can be written for the preserved distances between embeddings as follows:

$$\tilde{\mathbf{y}}_1 = \left[\overbrace{\sqrt{\frac{N_2}{N_1(N_1 + N_2)}}, \dots, \sqrt{\frac{N_2}{N_1(N_1 + N_2)}}}^{N_1}, \overbrace{0}^{\text{oiI}}, \overbrace{-\sqrt{\frac{N_1}{N_2(N_1 + N_2)}}, \dots, -\sqrt{\frac{N_1}{N_2(N_1 + N_2)}}}^{N_2}, \dots, \overbrace{0, \dots, 0}^{N_K} \right]^\top.$$

Based on this, the embedding vector associated with the Type II outlier yields

$$\tilde{\mathbf{e}}_{\text{II}} = \left[\overbrace{0, \dots, 0}^{K-2}, \frac{1}{\sqrt{N_1 + N_2 + 1}}, 0 \right],$$

which concludes the proof that $\tilde{\mathbf{e}}_{\text{II}}$ is centered between embeddings of blocks i and j if the distance between every pair of embedding vectors correspond to true samples are preserved. \square

A.2.1.4 PROOF OF PREPOSITION 4.2.2

Suppose that the distances between embeddings of true samples are preserved. Then, the squared Euclidean distances from the origin can be written for orthonormal set of eigenvectors associated with K smallest eigenvalues as follows:

$$N_1 \|\tilde{\mathbf{e}}_1\|^2 + \|\tilde{\mathbf{e}}_{\text{II}}\|^2 + N_2 \|\tilde{\mathbf{e}}_2\|^2 + \dots + N_K \|\tilde{\mathbf{e}}_K\|^2 = K$$

If the embedding vectors in Theorem 2 are substituted in this equation, clearly, the embedding vector of Type II outlier is a vector of zeros. \square

A.2.1.5 PROOF OF PREPOSITION 4.2.3

Assuming that the column vectors of the matrices \mathbf{Y} and $\tilde{\mathbf{Y}}$ take values in a range $\{\gamma_{\min}, \gamma_{\max}\}$, without loss of generality, the matrices can be rewritten by defining the scaled eigenvectors of $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m$ as

$$\mathbf{Y} = \begin{bmatrix} s(0) & s(\gamma_{1,1}) \\ s(0) & s(\gamma_{1,1}) \\ s(\gamma_{3,0}) & s(0) \\ s(\gamma_{3,0}) & s(0) \end{bmatrix} \quad \tilde{\mathbf{Y}} = \begin{bmatrix} s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{1,1}) \\ s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{1,1}) \\ s(\tilde{\gamma}_{1,0}) & s(-\tilde{\gamma}_{1,1}) \\ s(\tilde{\gamma}_{1,0}) & s(-\tilde{\gamma}_{1,1}) \end{bmatrix},$$

where $s(\gamma_{n,m})$ and $s(\tilde{\gamma}_{n,m})$ denote the scaled n th embedding result in the m th eigenvector associated with Laplacian matrices \mathbf{L} and $\tilde{\mathbf{L}}$, respectively. The scaling function is given by $s(\gamma_{n,m}) = \gamma_{\min} + \frac{\gamma_{n,m} - \min(\mathbf{y}_m)}{\max(\mathbf{y}_m) - \min(\mathbf{y}_m)}(\gamma_{\max} - \gamma_{\min})$ where $\min(\mathbf{y}_m)$ and $\max(\mathbf{y}_m)$ denoting the minimum and the maximum valued embedding points in the eigenvector \mathbf{y}_m , respectively. To cluster the row vectors of the matrix \mathbf{Y} , the squared Euclidean distances within the blocks and between different blocks are evaluated as

$$\begin{aligned} \|\mathbf{e}_1 - \mathbf{e}_2\|_2^2 &= \|\mathbf{e}_3 - \mathbf{e}_4\|_2^2 = 0 \\ \|\mathbf{e}_1 - \mathbf{e}_3\|_2^2 &= (s(0) - s(\gamma_{3,0}))^2 + (s(\gamma_{1,1}) - s(0))^2 \\ &= 2(\gamma_{\max} - \gamma_{\min})^2, \end{aligned}$$

where $\|\mathbf{e}_m - \mathbf{e}_n\|_2^2$ denotes the squared Euclidean distance between the m th and the n th feature vector with $\|\mathbf{e}_m - \mathbf{e}_n\|_2^2 = \|\mathbf{e}_n - \mathbf{e}_m\|_2^2$ and $\|\mathbf{e}_1 - \mathbf{e}_3\|_2^2 = \|\mathbf{e}_1 - \mathbf{e}_4\|_2^2 = \|\mathbf{e}_2 - \mathbf{e}_3\|_2^2 = \|\mathbf{e}_2 - \mathbf{e}_4\|_2^2$.

Then, the distances within the blocks and between different blocks are evaluated for $\tilde{\mathbf{L}}\tilde{\mathbf{y}}_m = \tilde{\lambda}_m\tilde{\mathbf{y}}_m$ as

$$\begin{aligned} \|\tilde{\mathbf{e}}_1 - \tilde{\mathbf{e}}_2\|_2^2 &= \|\tilde{\mathbf{e}}_3 - \tilde{\mathbf{e}}_4\|_2^2 = 0 \\ \|\tilde{\mathbf{e}}_1 - \tilde{\mathbf{e}}_3\|_2^2 &= (s(\tilde{\gamma}_{1,0}) - s(\tilde{\gamma}_{1,0}))^2 + (s(\tilde{\gamma}_{1,1}) - s(-\tilde{\gamma}_{1,1}))^2 \\ &= (\gamma_{\max} - \gamma_{\min})^2. \end{aligned}$$

The next step is to examine the matrices \mathbf{Y} and $\tilde{\mathbf{Y}}$ for the scaled eigenvectors of $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m$,

as

$$\mathbf{Y} = \begin{bmatrix} s(0) & s(\gamma_{1,1}) \\ s(0) & s(\gamma_{1,1}) \\ s(\gamma_{3,0}) & s(0) \\ s(\gamma_{3,0}) & s(0) \end{bmatrix} \quad \tilde{\mathbf{Y}} = \begin{bmatrix} s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{1,1}) \\ s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{1,1}) \\ s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{3,1}) \\ s(\tilde{\gamma}_{1,0}) & s(\tilde{\gamma}_{3,1}) \end{bmatrix}.$$

As the eigenvectors of $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m$ are equivalent to $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m$ for the Laplacian matrix \mathbf{L} , the distances associated with \mathbf{Y} are also equal. Further, based on the knowledge that $\tilde{\gamma}_{1,1} = \frac{-w_2+2\tilde{w}_u}{w_1+2\tilde{w}_u}\tilde{\gamma}_{3,1}$, the scaled embedding points yield $s(\tilde{\gamma}_{1,1}) = \gamma_{\min}$ and $s(\tilde{\gamma}_{3,1}) = \gamma_{\max}$.³ Thus, the distances are computed for $\tilde{\mathbf{Y}}$ as

$$\begin{aligned} \|\tilde{\mathbf{e}}_1 - \tilde{\mathbf{e}}_2\|_2^2 &= \|\tilde{\mathbf{e}}_3 - \tilde{\mathbf{e}}_4\|_2^2 = 0 \\ \|\tilde{\mathbf{e}}_1 - \tilde{\mathbf{e}}_3\|_2^2 &= (s(\tilde{\gamma}_{1,0}) - s(\tilde{\gamma}_{1,0}))^2 + (s(\tilde{\gamma}_{1,1}) - s(-\tilde{\gamma}_{3,1}))^2 \\ &= (\gamma_{\max} - \gamma_{\min})^2. \end{aligned}$$

This concludes the proof that the distance between the row vectors of the \mathbf{Y} associated with different blocks is greater than that of $\tilde{\mathbf{Y}}$. \square

A.2.1.6 PROOF OF THEOREM 6

To analyze different positions of Type II outlier, the outlier is shifted along the diagonal of the corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ as follows

$$\tilde{\mathbf{L}}_{(m_{\text{II}}=1)} = \begin{bmatrix} \tilde{d}_{\text{II}} & -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},1} & \dots & -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},2} & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},K} & -\tilde{w}_{\text{II},K} & \dots & -\tilde{w}_{\text{II},K} \\ -\tilde{w}_{\text{II},1} & \tilde{d}_1 & -w_1 & \dots & -w_1 & \dots & & & & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & \tilde{d}_1 & \dots & -w_1 & \dots & & & & & & & & & \\ \vdots & \vdots & \vdots & \ddots & \vdots & \dots & & & & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & -w_1 & \dots & \tilde{d}_1 & \dots & & & & & & & & & \\ -\tilde{w}_{\text{II},2} & \vdots & \vdots & & \vdots & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & & & & & \\ -\tilde{w}_{\text{II},2} & & & & & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & & & & & \\ \vdots & & & & & \vdots & \vdots & \ddots & & & & & & & \\ -\tilde{w}_{\text{II},2} & & & & & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & & & & & \\ \vdots & & & & & \vdots & \vdots & & \vdots & \ddots & & & & & \\ -\tilde{w}_{\text{II},K} & & & & & \vdots & \vdots & & \vdots & \ddots & \tilde{d}_K & -w_K & \dots & -w_K & \\ -\tilde{w}_{\text{II},K} & & & & & & & & \dots & -w_K & \tilde{d}_K & \dots & -w_K & & \\ \vdots & & & & & & & & & \vdots & \vdots & \ddots & & & \\ -\tilde{w}_{\text{II},K} & & & & & & & & \dots & -w_K & -w_K & \dots & \tilde{d}_K & & \end{bmatrix}$$

³For a detailed discussion, see Appendix C.1 in [TMZ22].

$$\begin{aligned}
& \vdots \\
\tilde{\mathbf{L}}_{(m_{\text{II}}=\ell_2-1)} &= \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{\text{II},1} & \dots \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{\text{II},1} & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{\text{II},1} & \dots \\ -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},1} & \dots & -\tilde{w}_{\text{II},1} & \tilde{d}_{\text{II}} & -\tilde{w}_{\text{II},2} & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},K} & -\tilde{w}_{\text{II},K} & \dots & -\tilde{w}_{\text{II},K} \\ \vdots & \vdots & \vdots & -\tilde{w}_{\text{II},2} & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & & & & & & \\ & & & -\tilde{w}_{\text{II},2} & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & & & & & & \\ & & & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & & & & \\ & & & -\tilde{w}_{\text{II},2} & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & & & & & & \\ & & & \vdots & \vdots & \vdots & & \vdots & \ddots & & & & & & \\ & & & -\tilde{w}_{\text{II},K} & \vdots & \vdots & & \vdots & \vdots & & \tilde{d}_K & -w_K & \dots & -w_K & \\ & & & -\tilde{w}_{\text{II},K} & \vdots & \vdots & & \vdots & \vdots & & \dots & -w_K & \tilde{d}_K & \dots & -w_K \\ & & & \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots & \\ & & & -\tilde{w}_{\text{II},K} & \vdots & \vdots & & \vdots & \vdots & & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix} \\
& \vdots \\
\tilde{\mathbf{L}}_{(m_{\text{II}}=N+1)} &= \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & \dots & & & & & & & & & & -\tilde{w}_{\text{II},1} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & \dots & & & & & & & & & & -\tilde{w}_{\text{II},1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & & & & & & & & & & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & \dots & & & & & & & & & & -\tilde{w}_{\text{II},1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & & & & \vdots \\ & & & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & & & & & & & -\tilde{w}_{\text{II},2} \\ & & & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & & & & & & & -\tilde{w}_{\text{II},2} \\ & & & \vdots & \vdots & \ddots & \vdots & \vdots & & & & & & & \vdots \\ & & & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & & & & & & & -\tilde{w}_{\text{II},2} \\ & & & \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & \vdots \\ & & & \vdots & \vdots & \vdots & \vdots & \vdots & & & & & & & \vdots \\ & & & & & & & & \tilde{d}_K & -w_K & \dots & -w_K & -\tilde{w}_{\text{II},K} & & \\ & & & & & & & & \dots & -w_K & \tilde{d}_K & \dots & -w_K & -\tilde{w}_{\text{II},K} & \\ & & & & & & & & \vdots & \vdots & \ddots & \vdots & & & \\ & & & & & & & & \dots & -w_K & -w_K & \dots & \tilde{d}_K & -\tilde{w}_{\text{II},K} & \\ -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},1} & \dots & -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},2} & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},2} & \dots & -\tilde{w}_{\text{II},K} & -\tilde{w}_{\text{II},K} & \dots & -\tilde{w}_{\text{II},K} & \tilde{d}_{\text{II}} \end{bmatrix}
\end{aligned}$$

Then, for each position of the outlier $0 < m_{\text{II}} \leq N + 1$, $m_{\text{II}} \in \mathbb{Z}^+$, the vector $\tilde{\mathbf{v}}$ is computed as

$$\begin{aligned}
\tilde{\mathbf{v}}_{(m_{\text{II}}=1)} &= \underbrace{[0]}_{\tilde{v}_{\text{II}} \in \mathbb{R}}, \underbrace{[\tilde{w}_{\text{II},1}, \tilde{w}_{\text{II},1} + w_1, \dots, \tilde{w}_{\text{II},1} + (N_1 - 1)w_1, \dots]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \dots, \underbrace{[\tilde{w}_{\text{II},K}, \tilde{w}_{\text{II},K} + w_K, \dots, \tilde{w}_{\text{II},K} + (N_K - 1)w_K]}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}} \\
& \vdots \\
\tilde{\mathbf{v}}_{(m_{\text{II}}=\ell_2-1)} &= \underbrace{[0, w_1, \dots, (N_1 - 1)w_1]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \underbrace{[N_1 \tilde{w}_{\text{II},1}, \dots]}_{\tilde{v}_{\text{II}} \in \mathbb{R}}, \dots, \underbrace{[\tilde{w}_{\text{II},K}, \tilde{w}_{\text{II},K} + w_K, \dots, \tilde{w}_{\text{II},K} + (N_K - 1)w_K]}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}} \\
& \vdots \\
\tilde{\mathbf{v}}_{(m_{\text{II}}=N+1)} &= \underbrace{[0, w_1, \dots, (N_1 - 1)w_1]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \dots, \underbrace{[0, w_K, \dots, (N_K - 1)w_K]}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}}, \underbrace{\sum_{j=1}^K N_j \tilde{w}_{\text{II},j}}_{\tilde{v}_{\text{II}} \in \mathbb{R}}
\end{aligned}$$

As it can be seen, the components of the corrupted vector $\tilde{\mathbf{v}} \in \mathbb{R}^{N+1}$, whose indexes are valued between the outlier index and the largest index of the j th block, i.e., $m_{\text{II}} < m \leq u_j$, increase by $\tilde{w}_{\text{II},j}$. Further, the component associated with the outlier is given by

$$\tilde{v}_{\text{II}} = \begin{cases} 0 & \text{if } 0 < m_{\text{II}} < \ell_1 \\ (m_{\text{II}} - \ell_1)\tilde{w}_{\text{II},1} & \text{if } \ell_1 < m_{\text{II}} < \ell_2 \\ \vdots & \\ \sum_{j=1}^{K-1} N_j \tilde{w}_{\text{II},j} + (m_{\text{II}} - \ell_K)\tilde{w}_{\text{II},K} & \text{if } \ell_K < m_{\text{II}} \leq N+1 \end{cases},$$

where ℓ_j denotes the lowest index of the j th block for $j = 1, \dots, K$. \square

A.2.1.7 PROOF OF THEOREM 7

To analyze different positions of block i that has similarity with the remaining $K - 1$ blocks, the block i is shifted along the diagonal of corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ as follows

$$\tilde{\mathbf{L}}_{i=1} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & & & & \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & & & & \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & & & & \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & & \ddots & \vdots & \vdots & & \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & & & & & & \tilde{d}_K & -w_K & \dots & -w_K \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & & & & & & \dots & -w_K & \tilde{d}_K & \dots & -w_K \\ \vdots & \vdots & \ddots & \vdots & & & & & & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & & & & & & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix}$$

$$\vdots$$

$$\tilde{\mathbf{L}}_{i=K} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & \dots & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & \dots & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & \dots & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & & \vdots & & & & & \ddots & \vdots \\ \vdots & \vdots & & \vdots & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ & & & & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ & & & & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & & \ddots & \vdots & \vdots & & \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & \tilde{d}_K & -w_K & \dots & -w_K \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & -w_K & \tilde{d}_K & \dots & -w_K \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix}$$

Then, for each position of the block i such that $i = 1, \dots, K$, the vector $\tilde{\mathbf{v}} \in \mathbb{R}^N$ is computed as

$$\begin{aligned} \tilde{\mathbf{v}}_{(i=1)} &= \underbrace{[0, w_1, \dots, (N_1 - 1)w_1, \dots]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \underbrace{[N_1 \tilde{w}_{1,K}, w_K + N_1 \tilde{w}_{1,K}, \dots, (N_K - 1)w_K + N_1 \tilde{w}_{1,K}]}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}} \\ &\vdots \\ \tilde{\mathbf{v}}_{(i=K)} &= \underbrace{[0, w_1, \dots, (N_1 - 1)w_1, \dots]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \underbrace{\left[\sum_{j=1}^{K-1} N_j \tilde{w}_{K,j}, w_K + \sum_{j=1}^{K-1} N_j \tilde{w}_{K,j}, \dots, (N_K - 1)w_K + \sum_{j=1}^{K-1} N_j \tilde{w}_{K,j} \right]}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}}. \end{aligned}$$

Herein, the components associated with the blocks $j > i$ are increase by $N_i \tilde{w}_{i,j}$ while the components associated with block $j \leq i$ will remain the same on the contrary. Further, if $2 \leq i \leq K$ the components associated with the block i increase by $\sum_{j=1}^{i-1} N_j \tilde{w}_{i,j}$ and remain the same otherwise.

In more details, starting from the $i + 1$ th block undesired similarity blocks are located only on the lower triangular side. Therefore, summing the upper triangular part of the Laplacian matrix results in an increase by $N_i \tilde{w}_{i,j}$ in these blocks. Additionally, for the i th block $i - 1$ number of undesired similarity blocks are located on the lower triangular side which results in an increase by $\sum_{j=1}^{i-1} N_j \tilde{w}_{i,j}$. \square

A.2.1.8 PROOF OF COROLLARY 7.1

Let $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ a corrupted Laplacian matrix, that is identical to $\mathbf{L} \in \mathbb{R}^{N \times N}$ except that each block $i = 1, \dots, K$ has non-zero similarity coefficients with the remaining $K - 1$ blocks, i.e.,

$$\tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,2} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & \tilde{d}_K & -w_K & \dots & -w_K \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & -w_K & \tilde{d}_K & \dots & -w_K \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & -\tilde{w}_{2,K} & -\tilde{w}_{2,K} & \dots & -\tilde{w}_{2,K} & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix}.$$

Here $\tilde{w}_{i,j}$ denotes the non-zero value around which the similarity coefficients between blocks i and j are concentrated. In contrast to \mathbf{L} , these non-zero coefficients of $\tilde{\mathbf{L}}$ lead to undesired edges between

vertices associated with different blocks. Then, the vector $\tilde{\mathbf{v}}$ associated with $\tilde{\mathbf{L}}$ reads

$$\tilde{\mathbf{v}} = \underbrace{[0, w_1, \dots, (N_1 - 1)w_1]}_{\tilde{\mathbf{v}}_1 \in \mathbb{R}^{N_1}}, \dots, \underbrace{\sum_{i=1}^{K-1} N_i \tilde{w}_{i,K}, w_K + \sum_{i=1}^{K-1} N_i \tilde{w}_{i,K}, \dots, (N_K - 1)w_K + \sum_{i=1}^{K-1} N_i \tilde{w}_{i,K}}_{\tilde{\mathbf{v}}_K \in \mathbb{R}^{N_K}},$$

which concludes the proof that the vector $\tilde{\mathbf{v}}$ is piece-wise linear function in the following form

$$\tilde{v}_m = \begin{cases} (m - \ell_1)w_1 & \text{if } \ell_1 \leq m \leq u_1 \\ (u_1 - \ell_1 + 1)\tilde{w}_{1,2} + (m - \ell_2)w_2 & \text{if } \ell_2 \leq m \leq u_2 \\ \vdots & \\ \sum_{i=1}^{K-1} (u_i - \ell_i + 1)\tilde{w}_{i,K} + (m - \ell_K)w_K & \text{if } \ell_K \leq m \leq u_K \end{cases}$$

where $\ell_1 = 1, u_1 = N_1, \ell_i = \sum_{k=1}^{i-1} N_k + 1$ and $u_i = \sum_{k=1}^i N_k$ for $i = 2, \dots, K$. \square

A.2.2 STANDARD EIGEN-DECOMPOSITION BASED ANALYSIS

A.2.2.1 TYPE II OUTLIERS' EFFECT ON EIGENVALUES

This section analyzes Type II outliers' effect on eigenvalues based on the standard eigen-decomposition in Eq. (2.5).

Theorem. 4.S. Let $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$ define a symmetric affinity matrix, that is equal to \mathbf{W} , except for an additional Type II outlier that shares similarity coefficients with K blocks where $\tilde{w}_{\Pi,K} > 0$ denotes the similarity coefficient between the outlier \mathbf{o}_{Π} and the K th block. Then, for the associated corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ with eigenvalues $\tilde{\boldsymbol{\lambda}} \in \mathbb{R}^{N+1}$, it holds that

$$\begin{cases} N_1 - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } N_1 w_1 + \tilde{w}_{\Pi,1} \\ N_2 - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } N_2 w_2 + \tilde{w}_{\Pi,2} \\ \vdots \\ N_K - 1 \text{ elements of } \tilde{\boldsymbol{\lambda}} \text{ are equal to } N_K w_K + \tilde{w}_{\Pi,K} \\ \text{the smallest element of } \tilde{\boldsymbol{\lambda}} \text{ is equal to zero} \end{cases}$$

and the remaining K eigenvalues are the roots of

$$\prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda}) \left(- \sum_{j=1}^K \frac{N_j \tilde{w}_{\Pi,j}}{\tilde{w}_{\Pi,j} - \tilde{\lambda}} - 1 \right) = 0.$$

Proof. Let $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$ denote the Laplacian matrix that is associated with a K block zero-diagonal symmetric affinity matrix with an additional Type II outlier that is correlated with all blocks, i.e.,

$$\tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_{\text{II}} & -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},1} & & & & & & & \\ -\tilde{w}_{\text{II},1} & \tilde{d}_1 & -w_1 & & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & \tilde{d}_1 & & & & & & & \\ \vdots & \vdots & \vdots & \ddots & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & -w_1 & \tilde{d}_1 & & & & & & \\ -\tilde{w}_{\text{II},2} & & & & \tilde{d}_2 & -w_2 & & -w_2 & & \\ -\tilde{w}_{\text{II},2} & & & & -w_2 & \tilde{d}_2 & & -w_2 & & \\ \vdots & & & & \vdots & \vdots & \ddots & \vdots & & \\ -\tilde{w}_{\text{II},2} & & & & -w_2 & -w_2 & & \tilde{d}_2 & & \\ \vdots & & & & & & & \ddots & \vdots & \\ -\tilde{w}_{\text{II},K} & & & & & & & & \tilde{d}_K & -w_K & -w_K \\ -\tilde{w}_{\text{II},K} & & & & & & & & -w_K & \tilde{d}_K & -w_K \\ \vdots & & & & & & & & \vdots & \vdots & \ddots \\ -\tilde{w}_{\text{II},K} & & & & & & & & -w_K & -w_K & \tilde{d}_K \end{bmatrix}$$

where $\tilde{d}_{\text{II}} = \sum_{j=1}^K N_j \tilde{w}_{\text{II},j}$ and $\tilde{d}_j = (N_j - 1)w_j + \tilde{w}_{\text{II},j}$. To compute the eigenvalues of the Laplacian matrix $\tilde{\mathbf{L}}$, $\det(\tilde{\mathbf{L}} - \tilde{\lambda}\mathbf{I}) = 0$ is considered which can equivalently be written in matrix form as follows

$$\begin{bmatrix} \tilde{d}_{\text{II}} - \tilde{\lambda} & -\tilde{w}_{\text{II},1} & -\tilde{w}_{\text{II},1} & & & & & & & \\ -\tilde{w}_{\text{II},1} & \tilde{d}_1 - \tilde{\lambda} & -w_1 & & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & \tilde{d}_1 - \tilde{\lambda} & & & & & & & \\ \vdots & \vdots & \vdots & \ddots & & & & & & \\ -\tilde{w}_{\text{II},1} & -w_1 & -w_1 & \tilde{d}_1 - \tilde{\lambda} & & & & & & \\ -\tilde{w}_{\text{II},2} & & & & \tilde{d}_2 - \tilde{\lambda} & -w_2 & & -w_2 & & \\ -\tilde{w}_{\text{II},2} & & & & -w_2 & \tilde{d}_2 - \tilde{\lambda} & & -w_2 & & \\ \vdots & & & & \vdots & \vdots & \ddots & \vdots & & \\ -\tilde{w}_{\text{II},2} & & & & -w_2 & -w_2 & & \tilde{d}_2 - \tilde{\lambda} & & \\ \vdots & & & & & & & \ddots & \vdots & \\ -\tilde{w}_{\text{II},K} & & & & & & & & \tilde{d}_K - \tilde{\lambda} & -w_K & -w_K \\ -\tilde{w}_{\text{II},K} & & & & & & & & -w_K & \tilde{d}_K - \tilde{\lambda} & -w_K \\ \vdots & & & & & & & & \vdots & \vdots & \ddots \\ -\tilde{w}_{\text{II},K} & & & & & & & & -w_K & -w_K & \tilde{d}_K - \tilde{\lambda} \end{bmatrix} = 0$$

To simplify the above determinant, the matrix determinant lemma (for details, see Lemma 1.1 in [DZ07]) can be generalized as follows⁴

$$\det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = \det(\mathbf{H})\det(\mathbf{I} + \mathbf{V}^\top\mathbf{H}^{-1}\mathbf{U})$$

where $\mathbf{H} \in \mathbb{R}^{(N+1) \times (N+1)}$ denotes an invertible matrix, \mathbf{I} is the identity matrix and $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{(N+1) \times (N+1)}$. Then, for $\det(\tilde{\mathbf{L}} - \tilde{\lambda}\mathbf{I}) = \det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = 0$, it follows that

⁴For a detailed information about the generalization of the matrix determinant lemma, see Section A.5.1.

$$\begin{aligned}
& \begin{pmatrix} z_{11}-\tilde{\lambda} & \dots & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ z_{11}-\tilde{\lambda} & \dots & \dots & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ z_{21}-\tilde{\lambda} & \dots & \dots & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ z_{21}-\tilde{\lambda} & \dots & \dots & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ z_{K1}-\tilde{\lambda} & \dots & \dots & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & \dots & z_{K1}-\tilde{\lambda} \end{pmatrix} + \begin{pmatrix} \underbrace{K}_{\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}} & \underbrace{N-K}_{\begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}} & \underbrace{N_1}_{\begin{bmatrix} -\tilde{w}_{11,1} & \dots & -\tilde{w}_{11,1} & \dots & -\tilde{w}_{11,1} \\ -w_1 & \dots & -w_1 & \dots & -w_1 \\ 0 & \dots & 0 & \dots & -w_2 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{11,K} & \dots & -\tilde{w}_{11,K} & \dots & -w_K \end{bmatrix}} & \underbrace{N_2}_{\begin{bmatrix} -\tilde{w}_{11,1} & -\tilde{w}_{11,2} & \dots & -\tilde{w}_{11,2} & \dots \\ 0 & 0 & \dots & 0 & \dots \\ -w_2 & \dots & -w_2 & \dots & -w_2 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ -w_K & \dots & -w_K & \dots & -w_K \end{bmatrix}} & \underbrace{N_K}_{\begin{bmatrix} -\tilde{w}_{11,K} & \dots & -\tilde{w}_{11,K} & \dots & -\tilde{w}_{11,K} \\ 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ -w_K & \dots & -w_K & \dots & -w_K \end{bmatrix}} \\
0 = \det & \begin{pmatrix} 0 & -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & \dots & -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & \dots & -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & \dots & -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} \\ -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & -w_1(z_{11}-\tilde{\lambda})^{-1} & \dots & -w_1(z_{11}-\tilde{\lambda})^{-1} & \dots & -w_1(z_{11}-\tilde{\lambda})^{-1} & \dots & -w_1(z_{11}-\tilde{\lambda})^{-1} \\ -\tilde{w}_{11,2}(z_{11}-\tilde{\lambda})^{-1} & 0 & \dots & 0 & \dots & -w_2(z_{11}-\tilde{\lambda})^{-1} & \dots & -w_2(z_{11}-\tilde{\lambda})^{-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{11,K}(z_{11}-\tilde{\lambda})^{-1} & 0 & \dots & 0 & \dots & -w_K(z_{11}-\tilde{\lambda})^{-1} & \dots & -w_K(z_{11}-\tilde{\lambda})^{-1} \\ 0 & \dots & \dots & 0 & \dots & \dots & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & \dots & \dots & \dots & \dots \end{pmatrix} \det(\mathbf{H}) \\
& \begin{pmatrix} 1 & -N_1\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & \dots & -N_2\tilde{w}_{11,2}(z_{11}-\tilde{\lambda})^{-1} & \dots & -N_K\tilde{w}_{11,K}(z_{11}-\tilde{\lambda})^{-1} & \dots & 0 \\ -\tilde{w}_{11,1}(z_{11}-\tilde{\lambda})^{-1} & -N_1w_1(z_{11}-\tilde{\lambda})^{-1}+1 & \dots & 0 & \dots & 0 & \dots & \vdots \\ -\tilde{w}_{11,2}(z_{11}-\tilde{\lambda})^{-1} & 0 & \dots & -N_2w_2(z_{11}-\tilde{\lambda})^{-1}+1 & \dots & 0 & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{11,K}(z_{11}-\tilde{\lambda})^{-1} & 0 & \dots & -N_Kw_K(z_{11}-\tilde{\lambda})^{-1}+1 & \dots & 0 & \dots & \vdots \\ 0 & \dots & \dots & \dots & \dots & \dots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & \dots & \dots & \dots & \vdots \end{pmatrix} \det(\mathbf{H})
\end{aligned}$$

where $z_{\text{II}} = \sum_{j=1}^K N_j \tilde{w}_{\text{II},j}$ and $z_j = N_j w_j + \tilde{w}_{\text{II},j}$ for $j = 1, \dots, K$. . Using the determinant properties of block matrices (for details, see Section 2 in [Sil00]), it holds that

$$\mathbf{0} = \begin{vmatrix} 1 & -N_1 \tilde{w}_{\text{II},1} (z_1 - \tilde{\lambda})^{-1} & -N_2 \tilde{w}_{\text{II},2} (z_2 - \tilde{\lambda})^{-1} & \dots & -N_K \tilde{w}_{\text{II},K} (z_K - \tilde{\lambda})^{-1} \\ -\tilde{w}_{\text{II},1} (z_{\text{II}} - \tilde{\lambda})^{-1} & -N_1 w_1 (z_1 - \tilde{\lambda})^{-1} + 1 & 0 & \dots & 0 \\ -\tilde{w}_{\text{II},2} (z_{\text{II}} - \tilde{\lambda})^{-1} & 0 & -N_2 w_2 (z_2 - \tilde{\lambda})^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\tilde{w}_{\text{II},K} (z_{\text{II}} - \tilde{\lambda})^{-1} & 0 & \dots & \dots & -N_K w_K (z_K - \tilde{\lambda})^{-1} + 1 \end{vmatrix} \det(\mathbf{H}).$$

To simplify the determinant of the first matrix, it transformed into a lower diagonal matrix by applying the following Gaussian elimination steps

$$\begin{aligned} & \frac{N_1 \tilde{w}_{\text{II},1} (z_1 - \tilde{\lambda})^{-1}}{-N_1 w_1 (z_1 - \tilde{\lambda})^{-1} + 1} R_2 + R_1 \rightarrow R_1 \\ & \frac{N_2 \tilde{w}_{\text{II},2} (z_2 - \tilde{\lambda})^{-1}}{-N_2 w_2 (z_2 - \tilde{\lambda})^{-1} + 1} R_3 + R_1 \rightarrow R_1 \\ & \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \vdots \\ & \frac{N_K \tilde{w}_{\text{II},K} (z_K - \tilde{\lambda})^{-1}}{-N_K w_K (z_K - \tilde{\lambda})^{-1} + 1} R_{K+1} + R_1 \rightarrow R_1 \end{aligned}$$

where R_K denotes the K th row. Then, the simplified determinant can be written as

$$\mathbf{0} = c_{\text{II}}(z_{\text{II}} - \tilde{\lambda}) \prod_{i=1}^K (-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1) \prod_{j=1}^K (z_j - \tilde{\lambda})^{N_j}$$

where

$$c_{\text{II}} = \left(1 - \prod_{i=1}^K \frac{N_i \tilde{w}_{\text{II},i} (z_i - \tilde{\lambda})^{-1} \tilde{w}_{\text{II},i} (z_{\text{II}} - \tilde{\lambda})^{-1}}{-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1} \right).$$

For $z_{\text{II}} = \sum_{j=1}^K N_j \tilde{w}_{\text{II},j}$ and $z_j = N_j w_j + \tilde{w}_{\text{II},j}$ such that $j = 1, \dots, K$ the determinant yields

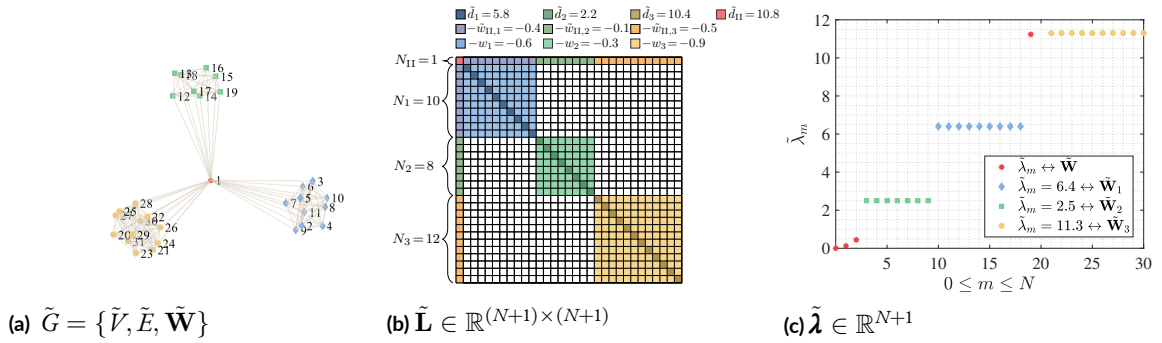


Figure A.2: Exemplary illustration of Theorem 4.5. ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}$, $N + 1 = 31$, $K = 3$).

$$\begin{aligned}
0 &= (z_{\Pi} - \tilde{\lambda})^{-1} \left(z_{\Pi} - \tilde{\lambda} - \sum_{i=1}^K \frac{N_i \tilde{w}_{\Pi,i} (z_i - \tilde{\lambda})^{-1} \tilde{w}_{\Pi,i}}{-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1} \right) (z_{\Pi} - \tilde{\lambda}) \prod_{j=1}^K (-N_j w_j (z_j - \tilde{\lambda})^{-1} + 1) \prod_{k=1}^K (z_k - \tilde{\lambda})^{N_k} \\
0 &= \left(\sum_{i=1}^K (N_i \tilde{w}_{\Pi,i}) - \tilde{\lambda} - \sum_{j=1}^K \frac{N_j \tilde{w}_{\Pi,j}^2}{\tilde{w}_{\Pi,j} - \tilde{\lambda}} \right) \prod_{k=1}^K (\tilde{w}_{\Pi,k} - \tilde{\lambda}) \prod_{l=1}^K (N_l w_l + \tilde{w}_{\Pi,l} - \tilde{\lambda})^{N_l - 1} \\
0 &= \left(-\tilde{\lambda} + \sum_{i=1}^K \left(N_i \tilde{w}_{\Pi,i} - \frac{N_i \tilde{w}_{\Pi,i}^2}{\tilde{w}_{\Pi,i} - \tilde{\lambda}} \right) \right) \prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda}) \prod_{k=1}^K (N_k w_k + \tilde{w}_{\Pi,k} - \tilde{\lambda})^{N_k - 1} \\
0 &= \left(-\tilde{\lambda} - \sum_{i=1}^K \frac{N_i \tilde{w}_{\Pi,i} \tilde{\lambda}}{\tilde{w}_{\Pi,i} - \tilde{\lambda}} \right) \prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda}) \prod_{k=1}^K (N_k w_k + \tilde{w}_{\Pi,k} - \tilde{\lambda})^{N_k - 1} \\
0 &= \tilde{\lambda} \left(-1 - \sum_{i=1}^K \frac{N_i \tilde{w}_{\Pi,i}}{\tilde{w}_{\Pi,i} - \tilde{\lambda}} \right) \prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda}) \prod_{k=1}^K (N_k w_k + \tilde{w}_{\Pi,k} - \tilde{\lambda})^{N_k - 1}
\end{aligned}$$

Now, $N + 1 - K$ number of eigenvalues can be computed as

$$\left\{ \begin{array}{l} N_1 - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } z_1 = N_1 w_1 + \tilde{w}_{\Pi,1} \\ N_2 - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } z_2 = N_2 w_2 + \tilde{w}_{\Pi,2} \\ \vdots \\ N_K - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } z_K = N_K w_K + \tilde{w}_{\Pi,K} \\ \text{the smallest element of } \boldsymbol{\lambda} \text{ is equal to zero} \end{array} \right.$$

and the remaining K number of eigenvalues are the roots of

$$0 = \prod_{j=1}^K (\tilde{w}_{\Pi,j} - \tilde{\lambda}) \left(-\sum_{j=1}^K \frac{N_j \tilde{w}_{\Pi,j}}{\tilde{w}_{\Pi,j} - \tilde{\lambda}} - 1 \right).$$

□

A.2.2.2 GROUP SIMILARITY EFFECT ON EIGENVALUES

This section analyzes group similarity effect on eigenvalues based on the standard eigen-decomposition in Eq. (2.5).

Theorem. 5.S. *Let $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ define an affinity matrix, that is equal to \mathbf{W} , except that block i has similarity with the remaining $K - 1$ blocks with $\tilde{w}_{i,j} = \tilde{w}_{j,i} > 0$ denoting the value around which the similarity coefficients between blocks i and j are concentrated for $j = 1, \dots, K$ and $i \neq j$. Then, the eigenvalues $\tilde{\lambda} \in \mathbb{R}^N$ of $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ are as follows:*

$$\left\{ \begin{array}{l} N_i - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_i w_i + \sum_{\substack{j=1 \\ j \neq i}}^K N_j \tilde{w}_{i,j} \\ N_j - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_j w_j + N_i \tilde{w}_{i,j} \\ \vdots \\ N_K - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_K w_K + N_i \tilde{w}_{i,K} \\ \text{the smallest element of } \boldsymbol{\lambda} \text{ is equal to zero} \end{array} \right.$$

and the remaining $K - 1$ eigenvalues in $\tilde{\lambda}$ are the roots of

$$\prod_{\substack{j=1 \\ j \neq i}}^K (N_i \tilde{w}_{i,j} - \tilde{\lambda}) \left(\sum_{\substack{j=1 \\ j \neq i}}^K -\frac{N_j \tilde{w}_{i,j}}{N_i \tilde{w}_{i,j} - \tilde{\lambda}} - 1 \right) = 0$$

where $\tilde{\lambda} \in \tilde{\lambda}$.

Proof. Let $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ denote the Laplacian matrix associated with K block zero diagonal affinity matrix in which i th block has similarity with the remaining $K - 1$ number of blocks. For simplicity, let $i = 1$, i.e.,

$$\tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 & -w_2 & \dots & -w_2 & \dots & \dots & \dots & \dots \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 & \dots & -w_2 & \dots & \dots & \dots & \dots \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 & \dots & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \tilde{d}_K & -w_K & \dots & -w_K & \dots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \dots & \tilde{d}_K & \dots & -w_K & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \dots & -w_K & -w_K & \dots & \tilde{d}_K \end{bmatrix}$$

where $\tilde{d}_1 = (N_1 - 1)w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j}$ and $\tilde{d}_j = (N_j - 1)w_j + N_1 \tilde{w}_{1,j}, j = 1, \dots, K$. To estimate the eigenvalues of the Laplacian matrix $\tilde{\mathbf{L}}, \det(\tilde{\mathbf{L}} - \tilde{\lambda}\mathbf{I}) = 0$ is considered which can equivalently be written in matrix form as follows

$$\begin{bmatrix} \tilde{d}_1 - \tilde{\lambda} & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -w_1 & \tilde{d}_1 - \tilde{\lambda} & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 - \tilde{\lambda} & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 - \tilde{\lambda} & -w_2 & \dots & -w_2 & \dots & \dots & \dots & \dots \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 - \tilde{\lambda} & \dots & -w_2 & \dots & \dots & \dots & \dots \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 - \tilde{\lambda} & \dots & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \tilde{d}_K - \tilde{\lambda} & -w_K & \dots & -w_K & \dots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \dots & \tilde{d}_K - \tilde{\lambda} & \dots & -w_K & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ -\tilde{w}_{1,K} & \dots & -\tilde{w}_{1,K} & \dots & \dots & \dots & \dots & -w_K & -w_K & \dots & \tilde{d}_K - \tilde{\lambda} \end{bmatrix} = 0.$$

To simplify, the matrix determinant lemma (for details, see Lemma 1.1 in [DZ07]) can be generalized as follows⁵

$$\det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = \det(\mathbf{H})\det(\mathbf{I} + \mathbf{V}^\top \mathbf{H}^{-1} \mathbf{U})$$

where $\mathbf{H} \in \mathbb{R}^{(N+1) \times (N+1)}$ denotes an invertible matrix, \mathbf{I} is the identity matrix and $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{(N+1) \times (N+1)}$. Then, for $\det(\tilde{\mathbf{L}} - \tilde{\lambda}\mathbf{I}) = \det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = 0$, it follows that

⁵For a detailed information about the generalization of the matrix determinant lemma, see Section A.5.1.

where $z_1 = N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j}$ and $z_j = N_j w_j + N_1 \tilde{w}_{1,j}$ for $j = 2, \dots, K$. Using the determinant properties of block matrices (for details, see Section 2 in [Sil00]), it holds that

$$0 = \det(\mathbf{H}) \begin{vmatrix} -N_1 w_1 (z_1 - \tilde{\lambda})^{-1} + 1 & -N_2 \tilde{w}_{1,2} (z_2 - \tilde{\lambda})^{-1} & -N_3 \tilde{w}_{1,3} (z_3 - \tilde{\lambda})^{-1} & \dots & -N_K \tilde{w}_{1,K} (z_K - \tilde{\lambda})^{-1} \\ -N_1 \tilde{w}_{1,2} (z_1 - \tilde{\lambda})^{-1} & -N_2 w_2 (z_2 - \tilde{\lambda})^{-1} + 1 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,3} (z_1 - \tilde{\lambda})^{-1} & 0 & -N_3 w_3 (z_3 - \tilde{\lambda})^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -N_1 \tilde{w}_{1,K} (z_1 - \tilde{\lambda})^{-1} & 0 & \dots & \dots & -N_K w_K (z_K - \tilde{\lambda})^{-1} + 1 \end{vmatrix}.$$

To simplify the determinant of the second matrix, it transformed into a lower diagonal matrix by applying the following Gaussian elimination steps

$$\begin{aligned} & \frac{N_2 \tilde{w}_{1,2} (z_2 - \tilde{\lambda})^{-1}}{-N_2 w_2 (z_2 - \tilde{\lambda})^{-1} + 1} R_2 + R_1 \rightarrow R_1 \\ & \frac{N_3 \tilde{w}_{1,3} (z_3 - \tilde{\lambda})^{-1}}{-N_3 w_3 (z_3 - \tilde{\lambda})^{-1} + 1} R_3 + R_1 \rightarrow R_1 \\ & \quad \vdots \\ & \frac{N_K \tilde{w}_{1,K} (z_K - \tilde{\lambda})^{-1}}{-N_K w_K (z_K - \tilde{\lambda})^{-1} + 1} R_K + R_1 \rightarrow R_1 \end{aligned}$$

where R_K denotes the K th row. Then, the simplified determinant yields

$$0 = \det(\mathbf{H}) \begin{vmatrix} c_1 & 0 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,2} (z_1 - \tilde{\lambda})^{-1} & -N_2 w_2 (z_2 - \tilde{\lambda})^{-1} + 1 & 0 & \dots & 0 \\ -N_1 \tilde{w}_{1,3} (z_1 - \tilde{\lambda})^{-1} & 0 & -N_3 w_3 (z_3 - \tilde{\lambda})^{-1} + 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -N_1 \tilde{w}_{1,K} (z_1 - \tilde{\lambda})^{-1} & 0 & \dots & \dots & -N_K w_K (z_K - \tilde{\lambda})^{-1} + 1 \end{vmatrix}$$

where c_1 equals to

$$c_1 = -N_1 w_1 (z_1 - \tilde{\lambda})^{-1} + 1 - \sum_{i=2}^K \frac{N_i \tilde{w}_{1,i} (z_i - \tilde{\lambda})^{-1} N_1 \tilde{w}_{1,i} (z_1 - \tilde{\lambda})^{-1}}{-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1}.$$

For $z_1 = N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j}$ and $z_j = N_j w_j + N_1 \tilde{w}_{1,j}$ such that $j = 2, \dots, K$, the determinant $\det(\tilde{\mathbf{L}} - \tilde{\lambda} \mathbf{I}) = 0$ yields

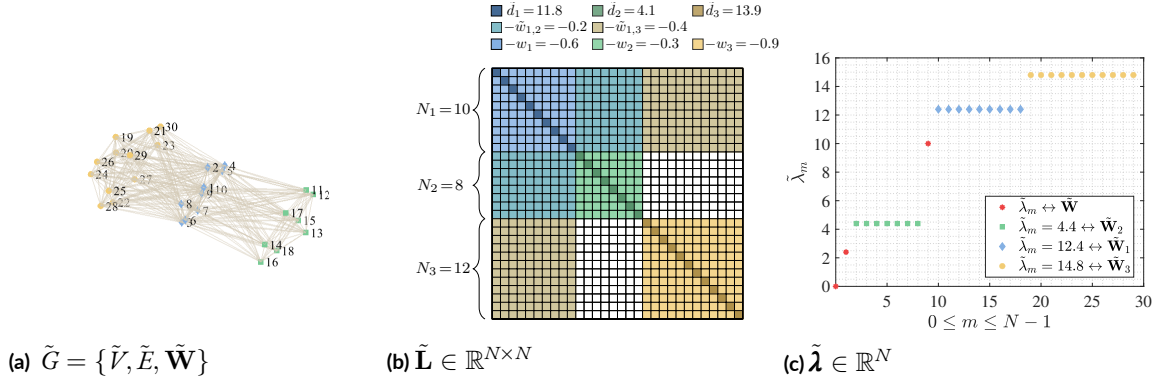


Figure A.3: Exemplary illustration of Theorem 5.5. ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K$, $N = 30$, $K = 3$, $i = 1$).

$$\begin{aligned}
0 &= c_1 \prod_{i=1}^K (z_i - \tilde{\lambda})^{N_i} \prod_{j=2}^K (-N_j w_j (z_j - \tilde{\lambda})^{-1} + 1) \\
0 &= c_1 (z_1 - \tilde{\lambda})^{N_1} \prod_{i=2}^K (-N_i w_i (z_i - \tilde{\lambda})^{N_i - 1} + (z_i - \tilde{\lambda})^{N_i}) \\
0 &= (z_1 - \tilde{\lambda})^{-1} \left(-N_1 w_1 + z_1 - \tilde{\lambda} - \sum_{i=2}^K \frac{N_i \tilde{w}_{1,i} (z_i - \tilde{\lambda})^{-1} N_1 \tilde{w}_{1,i}}{-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1} \right) (z_1 - \tilde{\lambda})^{N_1} \prod_{j=2}^K (z_j - \tilde{\lambda})^{N_j - 1} \prod_{k=2}^K (-N_k w_k + z_k - \tilde{\lambda}) \\
0 &= \left(-N_1 w_1 + z_1 - \tilde{\lambda} - \sum_{i=2}^K \frac{N_i \tilde{w}_{1,i} (z_i - \tilde{\lambda})^{-1} N_1 \tilde{w}_{1,i}}{-N_i w_i (z_i - \tilde{\lambda})^{-1} + 1} \right) \prod_{j=1}^K (z_j - \tilde{\lambda})^{N_j - 1} \prod_{k=2}^K (-N_k w_k + z_k - \tilde{\lambda}) \\
0 &= \left(\sum_{i=2}^K (N_i \tilde{w}_{1,i}) - \tilde{\lambda} - \sum_{j=2}^K \frac{N_1 N_j \tilde{w}_{1,j}^2}{N_1 \tilde{w}_{1,j} - \tilde{\lambda}} \right) \left(N_1 w_1 + \sum_{k=2}^K N_k \tilde{w}_{1,k} - \tilde{\lambda} \right)^{N_1 - 1} \prod_{l=2}^K \left(N_l w_l + N_1 \tilde{w}_{1,l} - \tilde{\lambda} \right)^{N_l - 1} \prod_{p=2}^K (N_1 \tilde{w}_{1,p} - \tilde{\lambda}) \\
0 &= \tilde{\lambda} \left(-1 - \sum_{i=2}^K \frac{N_i \tilde{w}_{1,i}}{N_1 \tilde{w}_{1,i} - \tilde{\lambda}} \right) \left(N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j} - \tilde{\lambda} \right)^{N_1 - 1} \prod_{k=2}^K \left(N_k w_k + N_1 \tilde{w}_{1,k} - \tilde{\lambda} \right)^{N_k - 1} \prod_{l=2}^K (N_1 \tilde{w}_{1,l} - \tilde{\lambda})
\end{aligned}$$

Based on this, the $N + 1 - K$ number of eigenvalues are

$$\left\{ \begin{array}{l} N_1 - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_1 w_1 + \sum_{j=2}^K N_j \tilde{w}_{1,j} \\ N_2 - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_2 w_2 + N_1 \tilde{w}_{1,2} \\ \vdots \\ N_K - 1 \text{ elements of } \boldsymbol{\lambda} \text{ are equal to } N_K w_K + N_1 \tilde{w}_{1,K} \\ \text{the smallest element of } \boldsymbol{\lambda} \text{ is equal to zero} \end{array} \right.$$

and additionally $K - 1$ number of eigenvalues are roots of the following equation

$$\prod_{j=2}^K (N_1 \tilde{w}_{1,j} - \tilde{\lambda}) \left(- \sum_{j=2}^K \frac{N_j \tilde{w}_{1,j}}{N_1 \tilde{w}_{1,j} - \tilde{\lambda}} - 1 \right) = 0.$$

□

A.3 OUTLIER EFFECTS ON THE FIEDLER VECTOR

A.3.0.I PROOF OF PREPOSITION 4.4.I

For simplicity, let $\mathbf{W} \in \mathbb{R}^{N \times N}$ and the corresponding Laplacian $\mathbf{L} \in \mathbb{R}^{N \times N}$ consist of $K = 2$ blocks. To compute the eigenvectors associated with the two smallest (zero-valued) eigenvalues, the $\mathbf{y}_m \in \{\mathbf{y}_0, \mathbf{y}_1\}$, where \mathbf{y}_m is the eigenvector associated with the m th eigenvalue λ_m of \mathbf{L} , we consider the eigen-decompositions $\mathbf{L}\mathbf{y}_m = \lambda_m \mathbf{y}_m$ and $\mathbf{L}\mathbf{y}_m = \lambda_m \mathbf{D}\mathbf{y}_m$ whose corresponding eigenvectors are equivalent and can be written as⁶

$$\begin{bmatrix} \mathcal{Y}_{1,0} \\ \mathcal{Y}_{1,0} \\ \mathcal{Y}_{3,0} \\ \mathcal{Y}_{3,0} \end{bmatrix} \quad \begin{bmatrix} \mathcal{Y}_{1,1} \\ \mathcal{Y}_{1,1} \\ \mathcal{Y}_{3,1} \\ \mathcal{Y}_{3,1} \end{bmatrix}$$

where $y_{n,m}$ denotes the n th embedding results in the \mathbf{y}_m . Adding a single outlier to the affinity matrix \mathbf{W} leads to the corrupted affinity matrix, weight matrix and Laplacian matrix denoted as $\tilde{\mathbf{W}} \in \mathbb{R}^{(N+1) \times (N+1)}$, $\tilde{\mathbf{D}} \in \mathbb{R}^{(N+1) \times (N+1)}$ and $\tilde{\mathbf{L}} \in \mathbb{R}^{(N+1) \times (N+1)}$, respectively. Since a Type I outlier can be considered as a single-element block, the eigenvectors associated with the three smallest (zero-valued) eigenvalues, i.e. $\tilde{\mathbf{y}}_m \in \{\tilde{\mathbf{y}}_0, \dots, \tilde{\mathbf{y}}_2\}$, where $\tilde{\mathbf{y}}_m$ is the eigenvector associated with the m th eigenvalue $\tilde{\lambda}_m$ for $\tilde{\mathbf{L}}$, are equivalent for both eigen-decompositions

$$\begin{bmatrix} \tilde{\mathcal{Y}}_{1,0} \\ \tilde{\mathcal{Y}}_{1,0} \\ \tilde{\mathcal{Y}}_{3,0} \\ \tilde{\mathcal{Y}}_{3,0} \\ \tilde{\mathcal{Y}}_{o_1,0} \end{bmatrix} \quad \begin{bmatrix} \tilde{\mathcal{Y}}_{1,1} \\ \tilde{\mathcal{Y}}_{1,1} \\ \tilde{\mathcal{Y}}_{3,1} \\ \tilde{\mathcal{Y}}_{3,1} \\ \tilde{\mathcal{Y}}_{o_1,1} \end{bmatrix} \quad \begin{bmatrix} \tilde{\mathcal{Y}}_{1,2} \\ \tilde{\mathcal{Y}}_{1,2} \\ \tilde{\mathcal{Y}}_{3,2} \\ \tilde{\mathcal{Y}}_{3,2} \\ \tilde{\mathcal{Y}}_{o_1,2} \end{bmatrix}.$$

⁶For a detailed discussion about the eigen-decomposition of \mathbf{L} , see [Lux07].

Here, $\tilde{y}_{n,m}$ denotes the n th embedding result in the m th eigenvector and $\tilde{y}_{o_1,m}$ is the embedding result of the outlier in the m th eigenvector of the corrupted Laplacian matrix $\tilde{\mathbf{L}}$. According to the information that eigenvectors are indicator vectors of connected components [Lux07], for a Fiedler vector that is associated with the eigenvalue corresponding to Type I outlier $|\tilde{y}_{o_1,F}| \rightarrow 1$, the remaining embedding results associated with different blocks become small-valued to satisfy $\|\tilde{\mathbf{y}}_F\|_2^2 = 1$. As a result, the Euclidean distance between embeddings of different blocks decreases to zero. \square

A.3.0.2 PROOF OF PREPOSITION 4.4.2

The eigenvectors associated with the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ of distinct blocks are identical for both eigen-decompositions $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m$ and $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m$. Then, the matrix $\mathbf{Y} = [\mathbf{y}_0, \dots, \mathbf{y}_{K-1}] \in \mathbb{R}^{N \times K}$, i.e., for $K = 2$

$$\mathbf{Y} = \begin{bmatrix} 0 & \gamma_{1,1} \\ 0 & \gamma_{1,1} \\ \gamma_{3,0} & 0 \\ \gamma_{3,0} & 0 \end{bmatrix}.$$

The next step is to design the matrix $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_0, \dots, \tilde{\mathbf{y}}_{K-1}] \in \mathbb{R}^{N \times K}$ using the eigenvectors of the corrupted Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ based on, respectively, the standard eigen-decomposition $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m$ and the generalized one $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m$ as⁷

$$\tilde{\mathbf{Y}} = \begin{bmatrix} \tilde{\gamma}_{1,0} & \tilde{\gamma}_{1,1} \\ \tilde{\gamma}_{1,0} & \tilde{\gamma}_{1,1} \\ \tilde{\gamma}_{1,0} & -\tilde{\gamma}_{1,1} \\ \tilde{\gamma}_{1,0} & -\tilde{\gamma}_{1,1} \end{bmatrix} \quad \tilde{\mathbf{Y}} = \begin{bmatrix} \tilde{\gamma}_{1,0} & \tilde{\gamma}_{1,1} \\ \tilde{\gamma}_{1,0} & \tilde{\gamma}_{1,1} \\ \tilde{\gamma}_{1,0} & \tilde{\gamma}_{3,1} \\ \tilde{\gamma}_{1,0} & \tilde{\gamma}_{3,1} \end{bmatrix}$$

By looking at the first column of $\tilde{\mathbf{Y}}$ associated with the smallest eigenvalue $\tilde{\lambda}_0$, it is evident that all feature vectors are embedded onto the same location $\tilde{\gamma}_{1,1}$ for the eigen-decompositions $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{y}_m$ and $\mathbf{L}\mathbf{y}_m = \lambda_m\mathbf{D}\mathbf{y}_m$. \square

⁷For a detailed information about the eigenvectors of $\tilde{\mathbf{L}}$, see Appendix C.1 and C.2 in [TMZ22].

A.4 THEORETICAL ANALYSIS OF RRLPI

A.4.0.1 PROOF OF THEOREM 9

The objective function of RLPI, for the transformation vector $\boldsymbol{\beta}_m$, is given by

$$\mathcal{L}(\mathbf{X}, \hat{\mathbf{y}}_m, \hat{\boldsymbol{\beta}}_m) = \|\mathbf{y}_m - \mathbf{X}^\top \boldsymbol{\beta}_m\|^2 + \gamma \|\boldsymbol{\beta}_m\|^2.$$

Introducing the weight matrix $\boldsymbol{\Omega} \in \mathbb{R}^{N \times N}$ leads to:

$$\begin{aligned} \mathcal{L}(\mathbf{X}, \hat{\mathbf{y}}_m, \hat{\boldsymbol{\beta}}_m) &= \boldsymbol{\Omega} \|\mathbf{y}_m - \mathbf{X}^\top \boldsymbol{\beta}_m\|^2 + \gamma \|\boldsymbol{\beta}_m\|^2 \\ &= \text{tr}[\boldsymbol{\Omega} \mathbf{y}_m \mathbf{y}_m^\top - 2\boldsymbol{\Omega} \mathbf{y}_m (\mathbf{X}^\top \boldsymbol{\beta}_m)^\top + \boldsymbol{\Omega} \mathbf{X}^\top \boldsymbol{\beta}_m (\mathbf{X}^\top \boldsymbol{\beta}_m)^\top] + \gamma \text{tr}(\boldsymbol{\beta}_m \boldsymbol{\beta}_m^\top) \\ &= \text{tr}[\boldsymbol{\Omega} \mathbf{y} \mathbf{y}_m^\top - 2\boldsymbol{\Omega} \mathbf{y}_m \boldsymbol{\beta}_m^\top \mathbf{X} + \boldsymbol{\Omega} \mathbf{X}^\top \boldsymbol{\beta}_m \boldsymbol{\beta}_m^\top \mathbf{X}] + \gamma \text{tr}(\boldsymbol{\beta}_m \boldsymbol{\beta}_m^\top) \\ &= \text{tr}[\boldsymbol{\Omega} \mathbf{y}_m \mathbf{y}_m^\top - 2\boldsymbol{\Omega} \mathbf{y}_m \boldsymbol{\beta}_m^\top \mathbf{X} + \boldsymbol{\beta}_m \boldsymbol{\beta}_m^\top (\mathbf{X} \boldsymbol{\Omega} \mathbf{X}^\top + \gamma \mathbf{I})] \end{aligned}$$

Setting the derivative of the right hand side with respect to $\boldsymbol{\beta}_m$ to zero, yields

$$-2\mathbf{X} \boldsymbol{\Omega} \mathbf{y}_m + 2\boldsymbol{\beta}_m (\mathbf{X} \boldsymbol{\Omega} \mathbf{X}^\top + \gamma \mathbf{I}) = 0.$$

Thus,

$$\hat{\boldsymbol{\beta}}_m = (\mathbf{X} \boldsymbol{\Omega} \mathbf{X}^\top + \gamma \mathbf{I})^{-1} \mathbf{X} \boldsymbol{\Omega} \mathbf{y}_m.$$

For a Fiedler vector estimate $\mathbf{y}_m = \mathbf{y}_F$, substituting $\boldsymbol{\Omega} = \mathbf{I}$ in Eq. (4.38) shows that RLPI is a special case of RRLPI and substituting $\boldsymbol{\Omega} = \mathbf{I}$ and $\gamma \rightarrow 0$ leads to LPI. \square

A.4.0.2 PROOF OF THEOREM 10

Suppose that $\text{rank}(\mathbf{X}) = r_{\mathbf{X}}$, the SVD of \mathbf{X} is

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top},$$

where $\mathbf{\Sigma} = \text{diag}(\Sigma_1, \dots, \Sigma_{r_{\mathbf{X}}})$, $\mathbf{U} \in \mathbb{R}^{M \times r_{\mathbf{X}}}$, $\mathbf{V} \in \mathbb{R}^{N \times r_{\mathbf{X}}}$ and $\mathbf{U}^{\top}\mathbf{U} = \mathbf{V}^{\top}\mathbf{V} = \mathbf{I}$. This can be generalized using the weighted singular value decomposition [Gal96], i.e.,

$$\mathbf{X}^* = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}\mathbf{\Omega}.$$

where $\mathbf{\Omega} \in \mathbb{R}^{N \times N}$ is a square positive definite symmetric weight matrix such that $\mathbf{V}^{\top}\mathbf{\Omega}\mathbf{V} = \mathbf{I}$.

Let \mathbf{V}^* be a weighted matrix whose columns are weighted orthonormal eigenvectors of \mathbf{V} , i.e., $\mathbf{V}^* = \mathbf{\Omega}\mathbf{V}$. Then, the orthogonality term can be equivalently written as $\mathbf{V}^{\top}\mathbf{V}^* = \mathbf{I}$. The Fiedler vector \mathbf{y}_F is in the space spanned by the column vectors of \mathbf{V}^* , because \mathbf{y}_F is spanned by row vectors of the weighted data matrix \mathbf{X}^* . Accordingly, the Fiedler vector \mathbf{y}_F can be represented as a unique linear combination of the linearly independent column vectors of \mathbf{V}^* . For a set of combination coefficients $\mathbf{b} \in \mathbb{R}^{r_{\mathbf{X}}}$,

$$\begin{aligned} \mathbf{V}^*\mathbf{b} &= \mathbf{y}_F \\ \mathbf{\Omega}\mathbf{V}\mathbf{b} &= \mathbf{y}_F \\ \mathbf{V}^{\top}\mathbf{\Omega}\mathbf{V}\mathbf{b} &= \mathbf{V}^{\top}\mathbf{y}_F \\ \mathbf{b} &= \mathbf{V}^{\top}\mathbf{y}_F. \end{aligned}$$

Substituting $\mathbf{b} = \mathbf{V}^{\top}\mathbf{y}_F$ into $\mathbf{V}^*\mathbf{b} = \mathbf{y}_F$, yields $\mathbf{V}^*\mathbf{V}^{\top}\mathbf{y} = \mathbf{y}_F$. To continue, using the pseudo inverse of the data matrix \mathbf{X}^{\dagger} and the weighted data matrix $(\mathbf{X}^*)^{\dagger}$ which can be written as ⁸

$$\mathbf{X}^{\dagger} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{\top}$$

and

$$(\mathbf{X}^*)^{\dagger} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{\top}\mathbf{\Psi},$$

⁸For a detailed discussion about the inverse of the weighted data matrix $(\mathbf{X}^*)^{\dagger}$, see Section A.5.2.

it follows that

$$\hat{\boldsymbol{\beta}}_F = (\mathbf{X}\boldsymbol{\Omega}\mathbf{X}^\top + \gamma\sigma^2\mathbf{I})^{-1}\mathbf{X}\boldsymbol{\Omega}\mathbf{y}_F = (\mathbf{X}^*\mathbf{X}^\top + \gamma\sigma^2\mathbf{I})^{-1}\mathbf{X}^*\mathbf{y}_F.$$

For $\gamma \rightarrow 0$

$$\begin{aligned}\hat{\boldsymbol{\beta}}_F &= (\mathbf{X}^*\mathbf{X}^\top + \gamma\sigma^2\mathbf{I})^{-1}\mathbf{X}^*\mathbf{y}_F = (\mathbf{X}^\top)^{-1}(\mathbf{X}^*)^{-1}\mathbf{X}^*\mathbf{y}_F = \mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^\top\boldsymbol{\Psi}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top\boldsymbol{\Omega}\mathbf{y}_F \\ &= \mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{V}(\mathbf{V})^{-1}\mathbf{y}_F = \mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{y}_F.\end{aligned}$$

Further, inserting the equation for $\boldsymbol{\beta}_F$ into $\hat{\mathbf{y}}_F = \mathbf{X}^\top\hat{\boldsymbol{\beta}}_F$ leads to

$$\hat{\mathbf{y}}_F = \mathbf{X}^\top\hat{\boldsymbol{\beta}}_F = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^\top\mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{y}_F = \mathbf{y}_F.$$

This shows that $\hat{\boldsymbol{\beta}}_F$ is the eigenvector of eigen-problem in Eq. (4.29) for $\gamma \rightarrow 0$. □

A.4.0.3 PROOF OF COROLLARY 10.1

Based on Theorem 10, it holds that for $\text{rank}(\mathbf{X}) = N$ all eigenvectors $\mathbf{y}_0, \dots, \mathbf{y}_{N-1}$ associated with $\lambda_0 \leq \lambda_1 \leq \dots \lambda_{N-1}$ are in the space spanned by the row vectors of \mathbf{X} and the transformation vector for the m th eigenvector is

$$\hat{\boldsymbol{\beta}}_m^{(\text{RRLPI})} = \mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{y}_m,$$

where $\gamma \rightarrow 0$, $\mathbf{U}^\top\boldsymbol{\Psi}\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top\boldsymbol{\Omega}\mathbf{V} = \mathbf{I}$. If all feature vectors are linearly independent, each transformation vector is unique and is equal to the transformation functions of LPI, i.e.,

$$\hat{\boldsymbol{\beta}}_m^{(\text{LPI})} = \mathbf{U}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{y}_m = \hat{\boldsymbol{\beta}}_m^{(\text{RRLPI})}.$$

□

A.5 AUXILIARY INFORMATION

A.5.1 THE GENERALIZED MATRIX DETERMINANT LEMMA

Let $\mathbf{M} \in \mathbb{R}^{2N \times 2N}$ be a block matrix that can be shown as

$$\begin{bmatrix} \mathbf{H} & \mathbf{U} \\ -\mathbf{V}^\top & \mathbf{I} \end{bmatrix}$$

where $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix and $\mathbf{H}, \mathbf{U}, \mathbf{V} \in \mathbb{R}^{N \times N}$. Using the determinant properties of block matrices with non-commuting blocks (for details, see Section 2 in [Sil00]), for $-\mathbf{V}^\top \mathbf{I} = \mathbf{I}(-\mathbf{V}^\top)$ the determinant of \mathbf{M} can be written as follows

$$\det(\mathbf{M}) = \det(\mathbf{H}\mathbf{I} - \mathbf{U}(-\mathbf{V}^\top)) = \det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top).$$

Now, the next step is to simplify the determinant $\det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top)$ by computing a block diagonal matrix using Gaussian elimination. First, the entry under \mathbf{H} is eliminated as follows

$$\begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{V}^\top \mathbf{H}^\dagger & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{H} & \mathbf{U} \\ -\mathbf{V}^\top & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & \mathbf{U} \\ 0 & \mathbf{V}^\top \mathbf{H}^\dagger \mathbf{U} + \mathbf{I} \end{bmatrix}.$$

Then, the entry above \mathbf{I} is eliminated as

$$\begin{bmatrix} \mathbf{H} & \mathbf{U} \\ -\mathbf{V}^\top & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{H}^\dagger \mathbf{U} \\ 0 & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & 0 \\ -\mathbf{V}^\top & \mathbf{V}^\top \mathbf{H}^\dagger \mathbf{U} + \mathbf{I} \end{bmatrix}.$$

Combining these two operations leads to

$$\begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{V}^\top \mathbf{H}^\dagger & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{H} & \mathbf{U} \\ -\mathbf{V}^\top & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{H}^\dagger \mathbf{U} \\ 0 & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & 0 \\ 0 & \mathbf{V}^\top \mathbf{H}^\dagger \mathbf{U} + \mathbf{I} \end{bmatrix}.$$

Consequently, the determinant yields

$$\det \left(\begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{V}^\top \mathbf{H}^\dagger & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{H} & \mathbf{U} \\ -\mathbf{V}^\top & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{H}^\dagger \mathbf{U} \\ 0 & \mathbf{I} \end{bmatrix} \right) = \det \left(\begin{bmatrix} \mathbf{H} & 0 \\ 0 & \mathbf{V}^\top \mathbf{H}^\dagger \mathbf{U} + \mathbf{I} \end{bmatrix} \right).$$

$$\det(\mathbf{H} + \mathbf{U}\mathbf{V}^\top) = \det(\mathbf{H})\det(\mathbf{I} + \mathbf{V}^\top \mathbf{H}^\dagger \mathbf{U})$$

A.5.2 MOORE-PENROSE INVERSE OF WEIGHTED DATA MATRIX

Let $\mathbf{X} \in \mathbb{R}^{M \times N}$, $\mathbf{\Omega} \in \mathbb{R}^{N \times N}$ and $\mathbf{\Psi} \in \mathbb{R}^{M \times M}$ denote a data matrix, and positive definite weight matrices, respectively. If there exist matrices $\mathbf{\Omega}$ and $\mathbf{\Psi}$ that satisfy $\mathbf{U}^\top \mathbf{\Psi} \mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top \mathbf{\Omega} \mathbf{V} = \mathbf{I}$ such that $\mathbf{X}^* = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega}$, the weighted Moore-Penrose inverse $(\mathbf{X}^*)^\dagger$ can be written as

$$(\mathbf{X}^*)^\dagger = \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi}$$

if it satisfies four conditions that are examined as

- $\mathbf{X}^* (\mathbf{X}^*)^\dagger \mathbf{X}^* \stackrel{?}{=} \mathbf{X}^*$

$$\mathbf{X}^* (\mathbf{X}^*)^\dagger \mathbf{X}^* = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} = \mathbf{U} \mathbf{\Sigma} \mathbf{\Sigma}^{-1} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} = \mathbf{X}^*$$

- $(\mathbf{X}^*)^\dagger \mathbf{X}^* (\mathbf{X}^*)^\dagger \stackrel{?}{=} (\mathbf{X}^*)^\dagger$

$$(\mathbf{X}^*)^\dagger \mathbf{X}^* (\mathbf{X}^*)^\dagger = \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} = \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{\Sigma} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} = \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} = (\mathbf{X}^*)^\dagger$$

- $(\mathbf{\Psi} \mathbf{X}^* (\mathbf{X}^*)^\dagger)^\top \stackrel{?}{=} \mathbf{\Psi} \mathbf{X}^* (\mathbf{X}^*)^\dagger$

$$(\mathbf{\Psi} \mathbf{X}^* (\mathbf{X}^*)^\dagger)^\top = (\mathbf{\Psi} \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega} \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi})^\top = (\mathbf{\Psi} \mathbf{U} \mathbf{U}^\top \mathbf{\Psi})^\top = ((\mathbf{U}^\top)^{-1} \mathbf{U}^{-1})^\top = (\mathbf{U}^\top)^{-1} \mathbf{U}^{-1} = \mathbf{\Psi} \mathbf{X}^* (\mathbf{X}^*)^\dagger$$

- $(\mathbf{\Omega} (\mathbf{X}^*)^\dagger \mathbf{X}^*)^\top \stackrel{?}{=} \mathbf{\Omega} (\mathbf{X}^*)^\dagger \mathbf{X}^*$

$$(\mathbf{\Omega} (\mathbf{X}^*)^\dagger \mathbf{X}^*)^\top = (\mathbf{\Omega} \mathbf{V} \mathbf{\Sigma}^{-1} \mathbf{U}^\top \mathbf{\Psi} \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega})^\top = (\mathbf{\Omega} \mathbf{V} \mathbf{V}^\top \mathbf{\Omega})^\top = ((\mathbf{V}^\top)^{-1} \mathbf{V}^{-1})^\top = (\mathbf{V}^\top)^{-1} \mathbf{V}^{-1} = \mathbf{\Omega} (\mathbf{X}^*)^\dagger \mathbf{X}^*$$

Thus, for weight matrices $\mathbf{\Omega}$ and $\mathbf{\Psi}$, which satisfy $\mathbf{U}^\top \mathbf{\Psi} \mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top \mathbf{\Omega} \mathbf{V} = \mathbf{I}$ such that $\mathbf{X}^* = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top \mathbf{\Omega}$, there exists a Moore-Penrose inverse of the weighted data matrix $(\mathbf{X}^*)^\dagger$.

B

Additional Information for Robust Graph Clustering Methods

This chapter provides methodological and experimental details, respectively, for the SPARCODE, FRS-BDR and RRLPI methods.

B.1 ADDITIONAL INFORMATION FOR SPARCODE

B.1.1 SCENARIO I

The $\text{SBM}_1(N, \boldsymbol{\alpha}, \mathbf{B})$ is defined using following parameters: $N = 300$ number of vertices, $K = 7$ communities, a probability vector $\boldsymbol{\alpha} = [\frac{N_{c_1}}{N}, \dots, \frac{N_{c_K}}{N}]$, and a symmetric connectivity matrix $\mathbf{B} \in \mathbb{R}^{K \times K}$ whose b_{c_i, c_j} th element denotes the probability of an edge between c_i th and c_j th community

block. For this simulation, \mathbf{B} is chosen as

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The affinity matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$, whose entries vary between zero and one, is symmetric, zero-diagonal and nonnegative. The community blocks are labelled as c_1, \dots, c_7 , and the density parameters (μ and σ) of similarity coefficients for within community and between community connections are summarized in Tab. B.1. The number of vertices for each block is denoted as N_c . The similarity coefficients are generated as $w = b_{c_i, c_j}(\mu_{c_i, c_j} + r\sigma_{c_i, c_j})$ s.t. $w \in b_{c_i, c_j}$ where r is a random real number distributed as $U(0, 1)$.

		c_1	c_2	c_3	c_4	c_5	c_6	c_7	N_c
c_1	μ	0.8	0.005	0.35	-	-	0.25	-	45
	σ	0.19	0.0045	0.1	-	-	0.05	-	
c_2	μ	0.005	0.7	-	0.35	0.19	-	-	40
	σ	0.0045	0.1	-	0.1	0.15	-	-	
c_3	μ	0.35	-	0.9	-	-	-	-	55
	σ	0.1	-	0.09	-	-	-	-	
c_4	μ	-	0.35	-	0.85	-	-	-	42
	σ	-	0.1	-	0.14	-	-	-	
c_5	μ	-	0.19	-	-	0.79	-	-	37
	σ	-	0.15	-	-	0.15	-	-	
c_6	μ	0.25	-	-	-	-	0.85	-	46
	σ	0.05	-	-	-	-	0.14	-	
c_7	μ	-	-	-	-	-	-	0.95	35
	σ	-	-	-	-	-	-	0.04	

Table B.1: Similarity coefficients for seven object communities for Scenario 1. The density parameters of similarity coefficients associated with $b_{c_i, c_j} = 0$ are denoted as '-'.

B.1.2 SCENARIO 2

The $\text{SBM}_2(N, \mathbf{a}, \mathbf{B})$ considers $N = 300$ number of vertices, $K = 3$ number of communities, a probability vector $\mathbf{a} = [\frac{N_{c_1}}{N}, \dots, \frac{N_{c_K}}{N}]$, and a symmetric connectivity matrix $\mathbf{B} \in \mathbb{R}^{K \times K}$ where

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

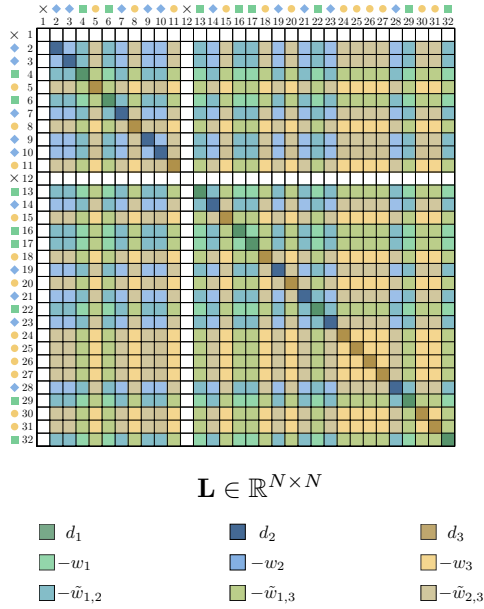
The communities are labelled as c_1, \dots, c_3 , and the outliers as o . The density parameters of similarity coefficients for intra-community, outer-community blocks and outliers are summarized in Tab. B.2. Both the number of vertices of communities and outliers are denoted as N_c . The similarity coefficients are generated with the same principle in the Appendix B.1.1.

		c_1	c_2	c_3	o	N_c
c_1	μ	0.7	0.01	-	0.005	66
	σ	0.29	0.29	-	0.025	
c_2	μ	0.01	0.8	0.27	0.005	102
	σ	0.29	0.19	0.1	0.025	
c_3	μ	-	0.27	0.7	0.005	90
	σ	-	0.1	0.24	0.025	
o	μ	0.005	0.005	0.005	0.005	42
	σ	0.025	0.025	0.025	0.025	

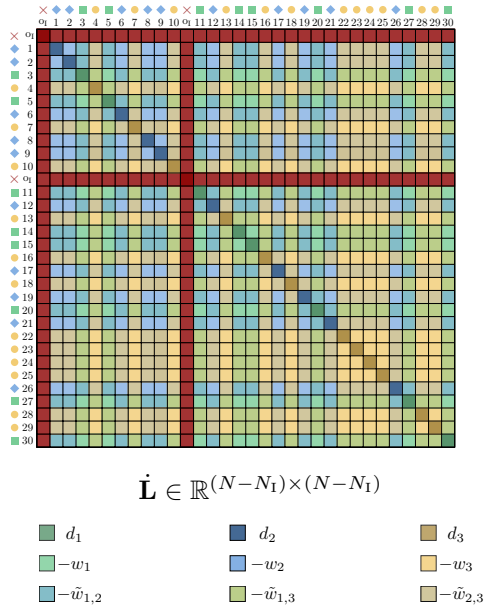
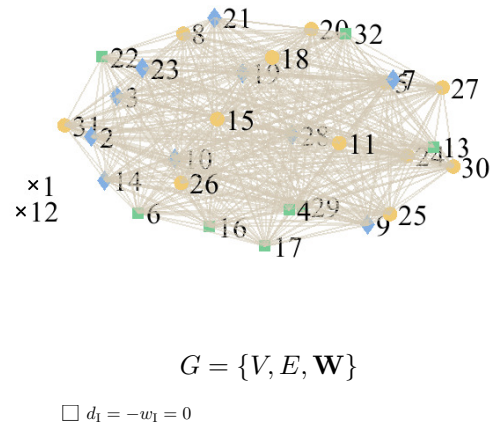
Table B.2: Similarity coefficients for three object communities and outliers for Scenario 2. The density parameters of similarity coefficients associated with $b_{c_i, c_j} = 0$ denotes as '-'.

B.2 ADDITIONAL INFORMATION FOR FRS-BDR

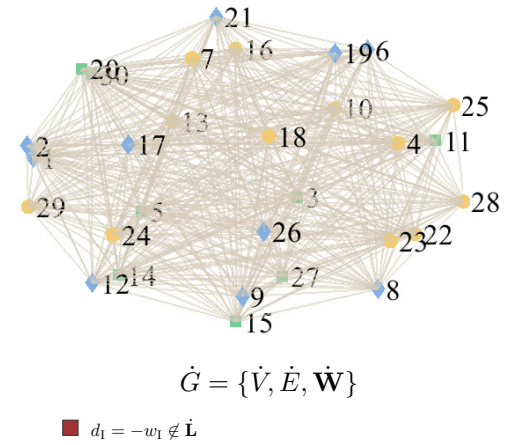
B.2.1 VISUAL SUMMARY OF FRS-BDR

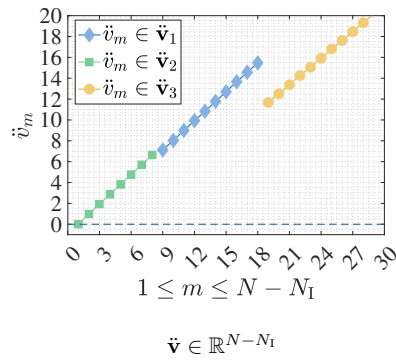
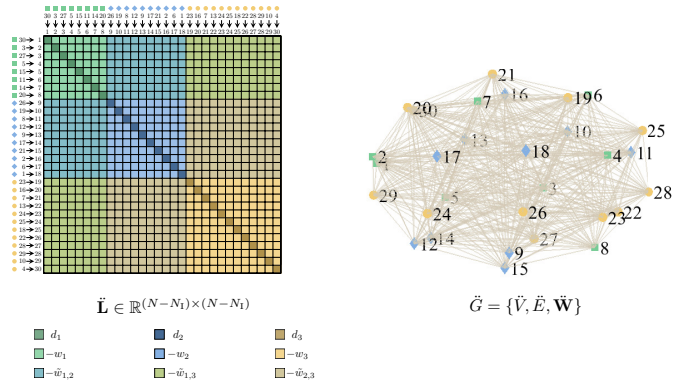


(a) Initialization

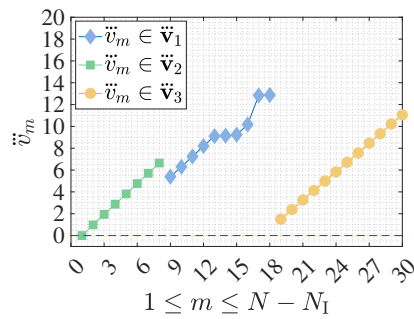
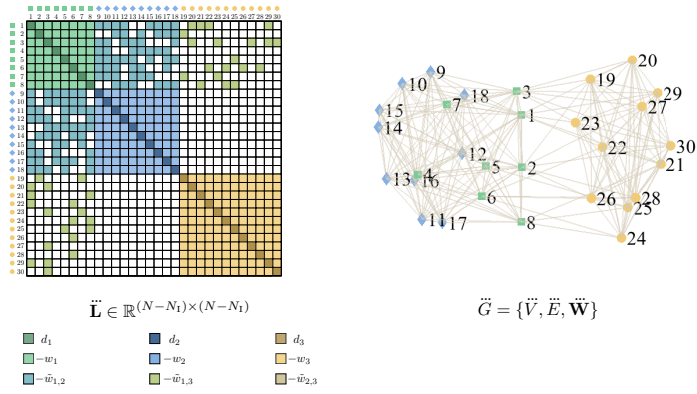


(b) Step 1.1: Type I Outlier Removal

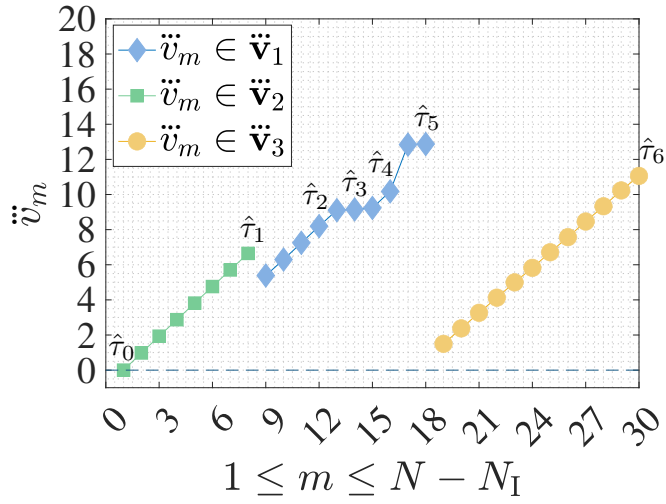




(c) Step 1.2: Similarity-based Block Diagonal Ordering (sBDO)



(d) Step 1.3: Increase Sparsity for Excessive Group Similarity



$$(N_c = 2 \times K_{\text{cand}} = 2 \times 3 = 6)$$

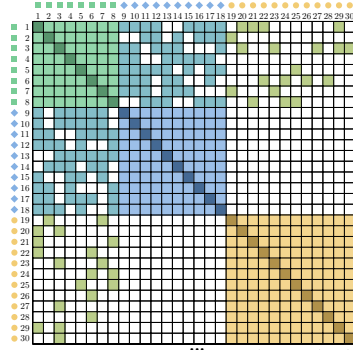
$$\ddot{\mathbf{v}} \in \mathbb{R}^{N-N_I}$$

8	4	18	}	\mathbf{n}_1
8	6	16		
8	8	14		
8	10	12	}	\mathbf{n}_4
8	15	7		
12	2	16	}	\mathbf{n}_6
12	4	14		
12	6	12		
12	11	7	}	\mathbf{n}_9
14	2	14		
14	4	12	}	\mathbf{n}_{11}
14	9	7		
16	2	12		
16	7	7	}	\mathbf{n}_{14}
18	5	7		

$$\mathbf{N}^{(K)} = [\mathbf{n}_1, \mathbf{n}_2, \dots, \mathbf{n}_\zeta]^\top \in \mathbb{Z}_+^{\zeta \times K}$$

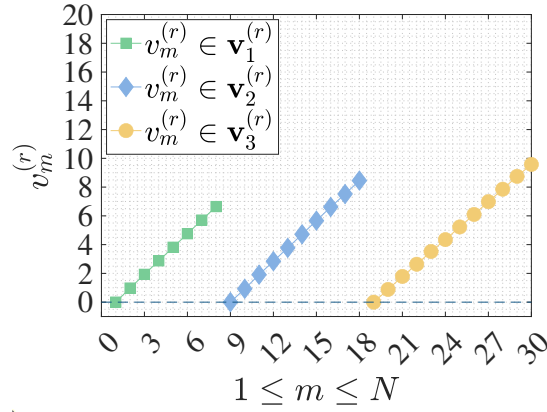
$$\zeta = \binom{N_c}{K-1} = \binom{6}{2} = 15$$

(e) Step 2.1: Compute Candidate Block Sizes



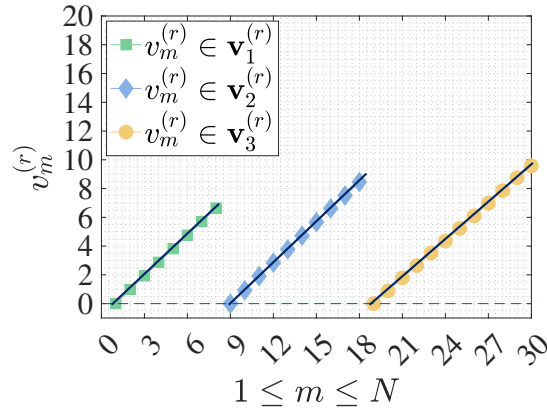
$$\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$$

$$\mathbf{n}_1 \in \mathbb{Z}_+^{(K)} \quad \mathbf{n}_r = \mathbf{n}_4 = [8, 10, 12]^T \in \mathbb{Z}_+^{(K)} \quad v_m^{(r)} = \sum_{n=m}^{u_{r_i}} \ddot{l}_{m,n} \text{ s.t. } \begin{matrix} \ell_{r_i} \leq m \leq u_{r_i} \\ i = 1, \dots, K \end{matrix} \quad \mathbf{n}_\zeta \in \mathbb{Z}_+^{(K)}$$



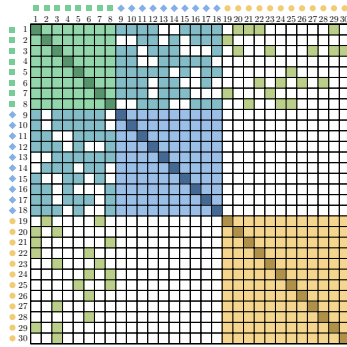
$$\mathbf{v}^{(r)} \in \mathbb{R}^{N-N_1}$$

piece-wise linear fitting



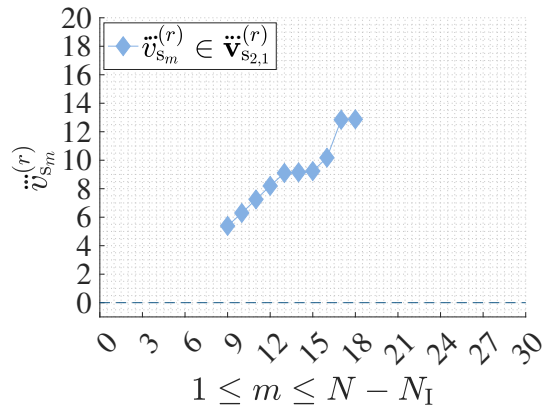
$$\hat{\mathbf{v}}^{(r)} \in \mathbb{R}^{N-N_1}$$

(f) Step 2.2.1: Estimate Target Similarity Coefficients $\text{diag}(\mathbf{W}_{\text{sim}})$

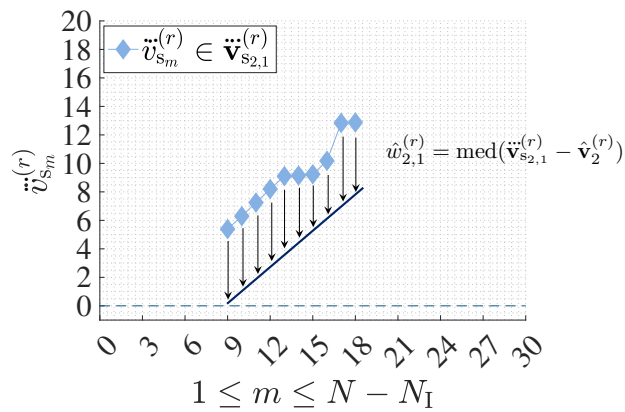


$$\ddot{\mathbf{L}} \in \mathbb{R}^{(N-N_1) \times (N-N_1)}$$

$$\ddot{\mathbf{v}}_{s_{2,1}}^{(r)} = \ddot{\mathbf{v}}_2^{(r)} + \ddot{\mathbf{v}}_{2,1}^{(r)} \quad \text{for} \quad \ddot{\mathbf{v}}_{2,1}^{(r)} = \left[-\sum_{n=\ell_{r_1}}^{u_{r_1}} \ddot{l}_{\ell_{r_2},n}^{(r)}, \dots, -\sum_{n=\ell_{r_1}}^{u_{r_1}} \ddot{l}_{u_{r_2},n}^{(r)} \right]^T \in \mathbb{R}^{N_{r_2}}$$

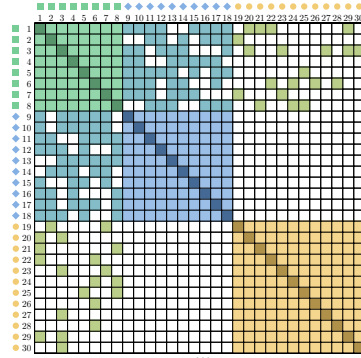


$$\ddot{\mathbf{v}}_{s_{2,1}}^{(r)} \in \mathbb{R}^{N_{r_2}}$$



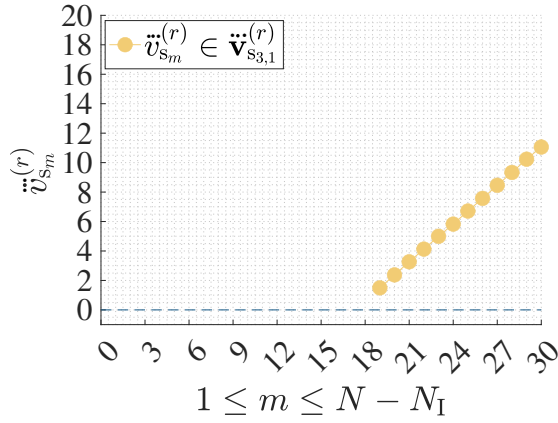
$$\ddot{\mathbf{v}}_{s_{2,1}}^{(r)} \quad \text{and} \quad \hat{\mathbf{v}}_2^{(r)} \in \mathbb{R}^{N_{r_2}}$$

(g) Step 2.2.2: Estimate Undesired Similarity Coefficients ($i = 2, j = 1$)

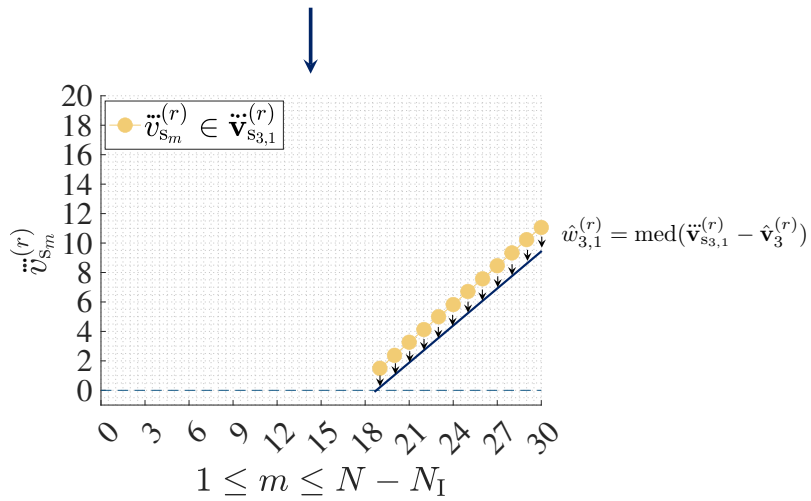


$$\tilde{\mathbf{L}} \in \mathbb{R}^{(N-N_I) \times (N-N_I)}$$

$$\downarrow \ddot{\mathbf{v}}_{s_{3,1}}^{(r)} = \ddot{\mathbf{v}}_3^{(r)} + \hat{\mathbf{v}}_{3,1}^{(r)} \quad \text{for} \quad \hat{\mathbf{v}}_{3,1}^{(r)} = \left[-\sum_{n=\ell_{r_1}}^{u_{r_1}} \ddot{l}_{\ell_{r_3},n}, \dots, -\sum_{n=\ell_{r_1}}^{u_{r_1}} \ddot{l}_{u_{r_3},n} \right]^T \in \mathbb{R}^{N_{r_3}}$$

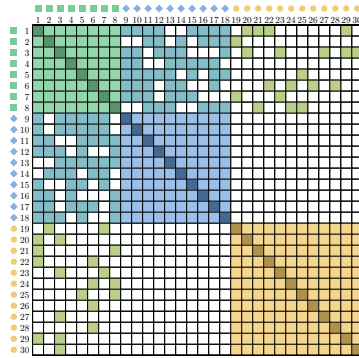


$$\ddot{\mathbf{v}}_{s_{3,1}}^{(r)} \in \mathbb{R}^{N_{r_3}}$$



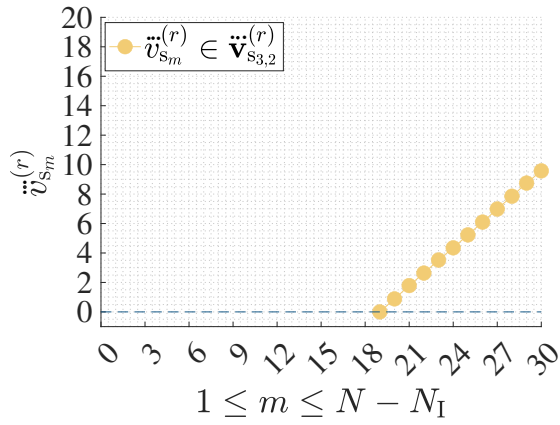
$$\ddot{\mathbf{v}}_{s_{3,1}}^{(r)} \quad \text{and} \quad \hat{\mathbf{v}}_3^{(r)} \in \mathbb{R}^{N_{r_3}}$$

(h) Step 2.2.2: Estimate Undesired Similarity Coefficients ($i = 3, j = 1$)

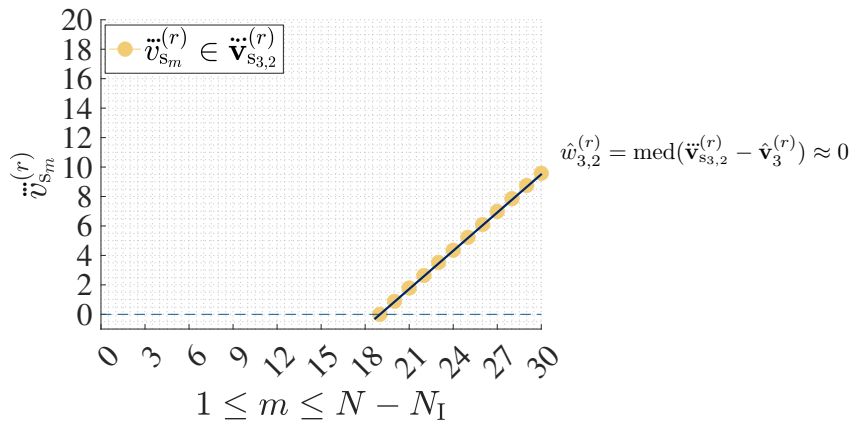


$$\tilde{\mathbf{L}} \in \mathbb{R}^{(N-N_I) \times (N-N_I)}$$

$$\downarrow \ddot{\mathbf{v}}_{s_{3,2}}^{(r)} = \ddot{\mathbf{v}}_3^{(r)} + \ddot{\mathbf{v}}_{3,2}^{(r)} \quad \text{for} \quad \ddot{\mathbf{v}}_{3,2}^{(r)} = \left[-\sum_{n=\ell_{r_2}}^{u_{r_2}} \ddot{l}_{\ell_{r_3},n}, \dots, -\sum_{n=\ell_{r_2}}^{u_{r_2}} \ddot{l}_{u_{r_3},n} \right]^T \in \mathbb{R}^{N_{r_3}}$$

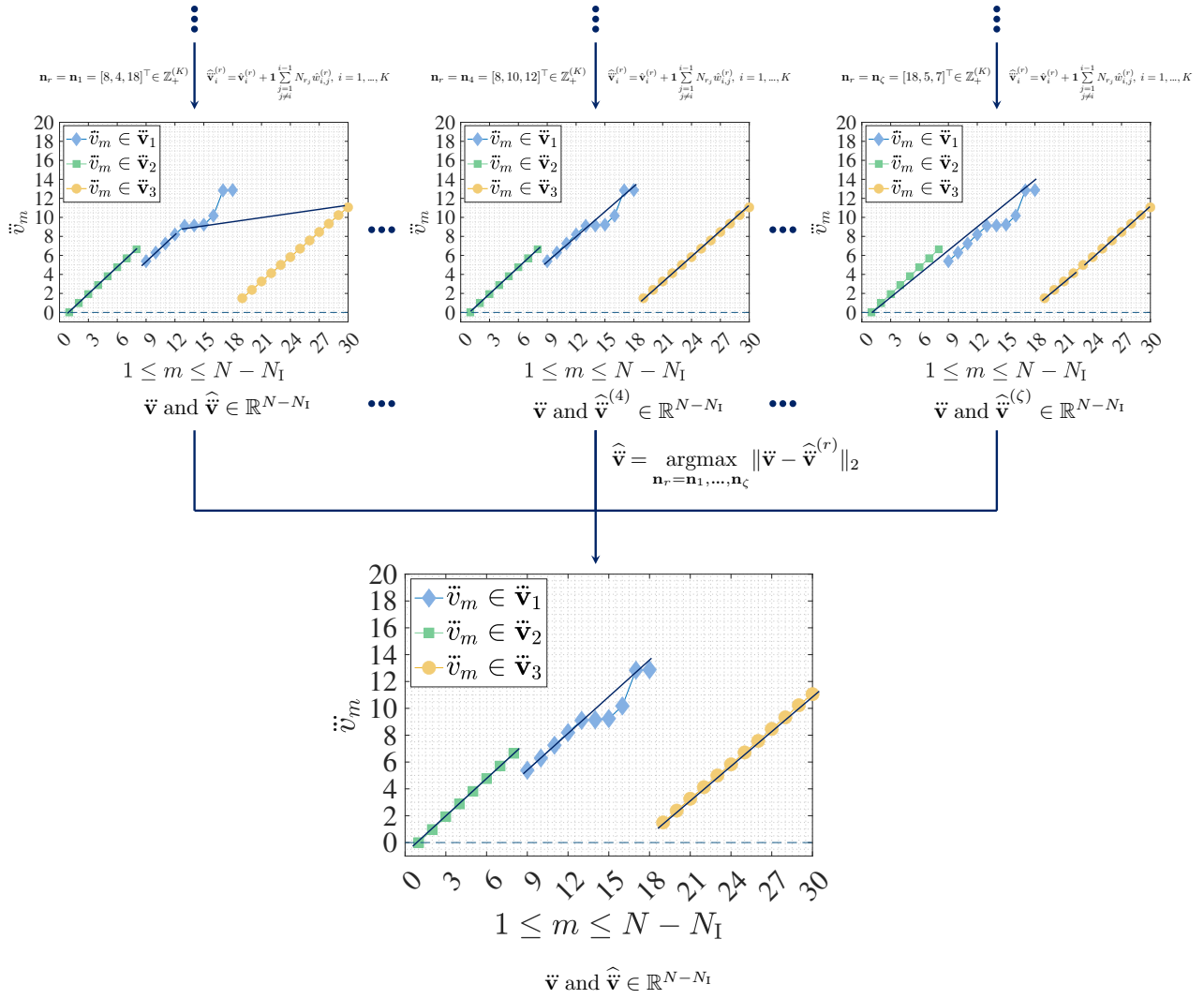


$$\ddot{\mathbf{v}}_{s_{3,2}}^{(r)} \in \mathbb{R}^{N_{r_3}}$$



$$\ddot{\mathbf{v}}_{s_{3,2}}^{(r)} \quad \text{and} \quad \hat{\mathbf{v}}_3^{(r)} \in \mathbb{R}^{N_{r_3}}$$

(i) Step 2.2.2: Estimate Undesired Similarity Coefficients ($i = 3, j = 2$)



(j) Step 2.3: Estimate Vector $\ddot{\mathbf{v}}$

Figure B.1: Visual summary of FRS-BDR

B.2.2 SPARSE LAPLACIAN MATRIX ANALYSIS

In Step 1.3, a sparse Laplacian matrix has been determined in which second smallest eigenvalue λ_1 closes to zero. Considering eigenvalues of Laplacian matrix, that is associated with target block zero-diagonal symmetric affinity matrix, λ_1 is definitely zero-valued. However, in real-world applications the distinct blocks may include negligibly small valued undesired similarity coefficients between different blocks. These coefficients result in an increase of λ_1 and affect definition of "close to zero". Therefore, this section provides set of experiments for determining a sparse Laplacian matrix.

In Theorem 5 it has been shown that multiple group similarity results in additional increase in the vector of eigenvalues. Therefore, for simplicity, $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ defining a $K = 2$ block affinity matrix and associated Laplacian matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{N \times N}$ are considered in the experiments. In $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ each block $\mathbf{W}_i, i = 1, 2$ is associated to a number $N_i \in \mathbb{Z}_+ > 1$ of feature vectors and concentrated around a similarity constant $w_i \in \mathbb{R}_+, i = 1, 2$ with negligibly small variations. Further, $\tilde{w}_{i,j}$ denotes a constant around which the similarity coefficients between blocks i and j are concentrated, i.e.

$$\tilde{\mathbf{W}} = \begin{bmatrix} 0 & w_1 & \dots & w_1 & \tilde{w}_{1,2} & \dots & \tilde{w}_{1,2} \\ w_1 & 0 & \dots & w_1 & \tilde{w}_{1,2} & \dots & \tilde{w}_{1,2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ w_1 & w_1 & \dots & 0 & \tilde{w}_{1,2} & \dots & \tilde{w}_{1,2} \\ \tilde{w}_{1,2} & \dots & \dots & \tilde{w}_{1,2} & 0 & w_2 & \dots & w_2 \\ \tilde{w}_{1,2} & \dots & \dots & \tilde{w}_{1,2} & w_2 & 0 & \dots & w_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \tilde{w}_{1,2} & \dots & \dots & \tilde{w}_{1,2} & w_2 & w_2 & \dots & 0 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{L}} = \begin{bmatrix} \tilde{d}_1 & -w_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} \\ -w_1 & \tilde{d}_1 & \dots & -w_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ -w_1 & -w_1 & \dots & \tilde{d}_1 & -\tilde{w}_{1,2} & \dots & -\tilde{w}_{1,2} \\ -\tilde{w}_{1,2} & \dots & \dots & -\tilde{w}_{1,2} & \tilde{d}_2 & -w_2 & \dots & -w_2 \\ -\tilde{w}_{1,2} & \dots & \dots & -\tilde{w}_{1,2} & -w_2 & \tilde{d}_2 & \dots & -w_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\tilde{w}_{1,2} & \dots & \dots & -\tilde{w}_{1,2} & -w_2 & -w_2 & \dots & \tilde{d}_2 \end{bmatrix},$$

where $\tilde{d}_1 = (N_1 - 1)w_1 + N_2\tilde{w}_{1,2}$, $\tilde{d}_2 = (N_2 - 1)w_2 + N_1\tilde{w}_{1,2}$ and $w_i > \tilde{w}_{1,2}, i = 1, 2$. According to the generalized eigen-decomposition, the second smallest eigenvalue λ_1 of $\tilde{\mathbf{L}}$ is

$$\lambda_1 = \frac{\tilde{d}_1 N_1 \tilde{w}_{1,2} + \tilde{d}_2 N_2 \tilde{w}_{1,2}}{\tilde{d}_1 \tilde{d}_2}.$$

Based on this, for $\tilde{w}_{1,2} \rightarrow 1$ and $w_i > \tilde{w}_{1,2}, i = 1, 2$, λ_1 reaches its maximum value

$$\lambda_{1\max} = \frac{N_1 + N_2}{N_1 + N_2 - 1}$$

which tends to 1 for $N_1 + N_2 \gg 1$. Even though the maximum value of λ_1 is explicit, its minimum value depends on different variables N_i, w_i and $\tilde{w}_{i,j}$ for $i = 1, 2, j = 1, 2$ and $i \neq j$. Therefore, λ_1

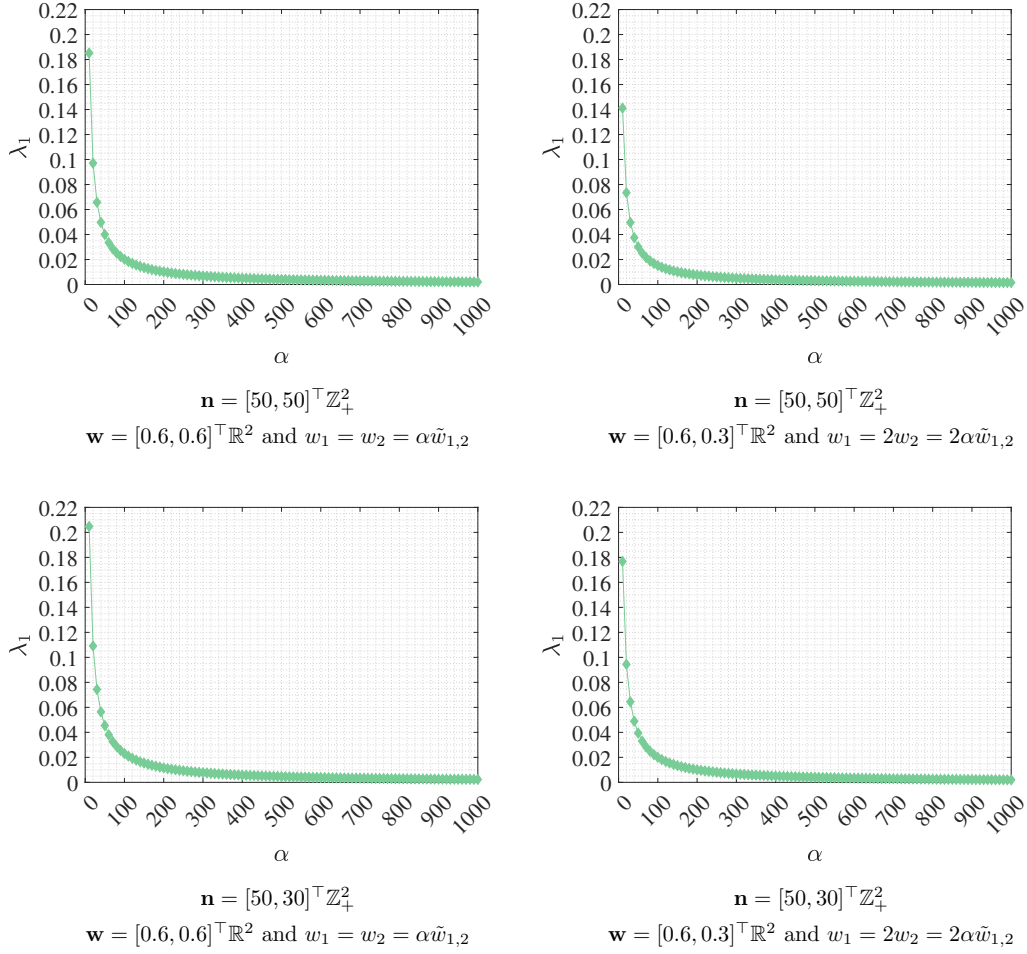


Figure B.2: λ_1 for increasing values of α associated with generalized eigen-decomposition

is analyzed as a function of α for different block size values in Fig. B.2. Here, α denotes the ratio between smallest target similarity coefficient w_{\min} and that of undesired $\tilde{w}_{i,j}$ for $i = 1, 2, j = 1, 2$ and $i \neq j$. As can be seen, the second smallest eigenvalue λ_1 decreases to zero for $w_1 \gg \tilde{w}_{1,2}$ and $w_2 \gg \tilde{w}_{1,2}$.

In contrast to generalized eigen-decomposition, the second smallest eigenvalue λ_1 of $\tilde{\mathbf{L}}$ associated to the standard eigen-decomposition does not affected by target similarity coefficients, i.e. $\lambda_1 = \tilde{w}_{1,2}(N_1 + N_2)$.

As $N_i > 1, i = 1, 2$, λ_1 can be reduced to zero for considerably small-valued undesired similarity coefficients. Therefore, λ_1 is analyzed as a function of $\tilde{w}_{1,2}$ for different block size values in Fig. B.3. The figure implies that the value of undesired similarity coefficients is directly linked to closeness to zero. Therefore, the closeness to zero can be determined according to total sample size and

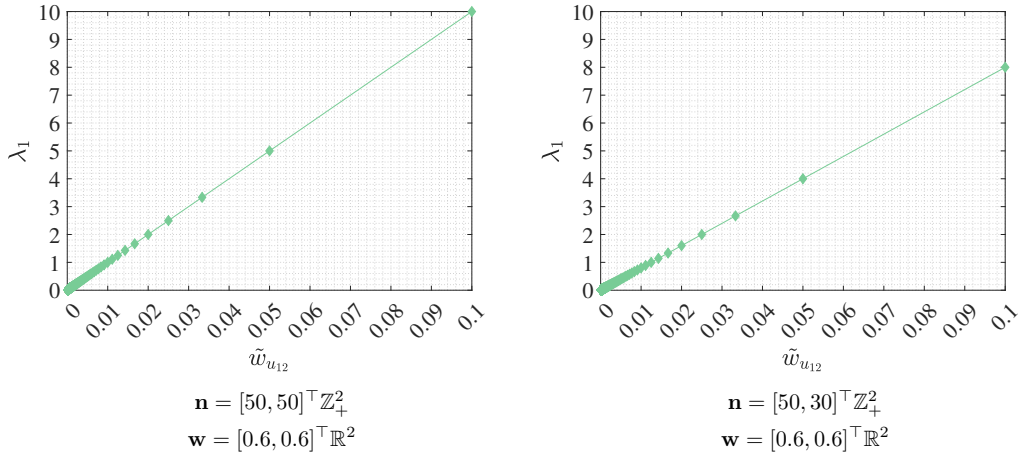


Figure B.3: λ_1 for increasing values of $\tilde{w}_{1,2}$ associated with standard eigen-decomposition

desired value of undesired similarity. To summarize, the eigenvalues based on generalized eigen-decomposition are less sensitive to block sizes which makes definition of close easier. As a result, based on the assumption that target similarity coefficients considerably larger valued than that of undesired coefficients, a selected λ_1 value smaller than 0.05 might be sufficient to obtain sparse Laplacian matrices. In the proposed default setting, a Laplacian matrix is assumed to be sparse if its second smallest eigenvalue is valued by $0 \leq \lambda_1 < 10^{-3}$.

B.2.3 COMPUTATIONAL COMPLEXITY ANALYSIS

Due to its essential role in graph analysis, the computational complexity of the proposed FRS-BDR method is analyzed in terms of its main operations. The computational analysis is detailed using the following terms [Ste01], [Ste98]:

fladd : an operation that consists of a floating-point addition

flmlt : an operation that consists of a floating-point multiplication

fldiv : an operation that consists of a floating-point division

flam : a compound operation that consists of one addition and one multiplication

Additionally, the Landau's big O symbol is used when the complexity is not specified as above terms. In the sequel, the computational complexity of the proposed approach is detailed for the fundamental steps.

B.2.3.0.1 INITIAL GRAPH CONSTRUCTION

As in [CHZ07], the pairwise cosine similarity which takes $\frac{1}{2}N^2M + 2NM$ flam can be used for constructing an initial graph G .

B.2.3.0.2 STEP 1: ENHANCING BD STRUCTURE

Step 1.1 (optional): Type I Outlier Removal

Since removing unconnected vertices associated with Type I outliers results in negligibly smaller cost in comparison to remaining estimation steps, the complexity of Type I outlier removal is ignored in the complexity analysis.

Step 1.2: sBDO Algorithm

In this step, the complexity of sBDO algorithm is detailed for two main operations as follows.

To determine the starting nodes, the overall edge weights must be sorted which is of complexity $O(N \log N)$ and there are computationally efficient alternatives such as [Han20] for which the complexity is reduced to $O(N\sqrt{\log N})$.

The second main operation is adding the most similar node to the vector of previously estimated nodes. For a vector of previously estimated nodes $\hat{\mathbf{b}}^{(s)} \in \mathbb{Z}_+^{s-1}$ at the s th stage, the method sums up the similarity coefficients that takes $s - 2$ additions for every neighbor. For $N_{\text{neigh}}^{(s)}$ number of neighbors, it follows that

$$\sum_{s=2}^N (s-2) N_{\text{neigh}}^{(s)}$$

fladd. Even though the complexity is directly linked to number of neighbors, it can be explicitly calculated if every node is connected to remaining $N - 1$ number of nodes, i.e.

$$\sum_{s=2}^N (s-2)(N-s+1)$$

fladd. In addition to similarity computation, the method sorts vector of neighbor similarities to find the most similar node. This sorting operation results in minimum $O(N_{\text{neigh}}^{(s)} \sqrt{\log N_{\text{neigh}}^{(s)}})$ complexity at every stage $s = 2, \dots, N$.

Step 1.3 (optional): Increase Sparsity for Excessive Group Similarity

This step evaluates computational complexity of sparse Laplacian matrix design for the two exemplary algorithms which have been provided in Section B.2.4.

Sparse Laplacian Matrix Design based on Adaptive Thresholding: As eigen-decomposition and sorting operations are computationally demanding in comparison to thresholding operation, the complexity is detailed in terms of these two main processes. The proposed sparse Laplacian matrix design computes eigenvalues for each iteration in which e.g. MATLAB uses a Krylov Schur decomposition [Ste02]. The decomposition is built upon two main phases that are known as expansion and contraction. The computational cost of decomposition mainly depends on these phases when N is larger than N_{Lan} , where N_{Lan} denotes the number of Lanczos basis vectors (preferably chosen as $N_{\text{Lan}} \geq 2N_{\text{eig}}$ for N_{eig} eigenvectors). In more details, the expansion phase requires between $N(N_{\text{Lan}}^2 - N_{\text{eig}}^2)$ flam and $2N(N_{\text{Lan}}^2 - N_{\text{eig}}^2)$ flam while the contraction phase requires $NN_{\text{Lan}}N_{\text{eig}}$ flam [Ste01]. To find the second smallest eigenvalues, the computed eigenvalues must be sorted which is of complexity $O(N\sqrt{\log N})$ in [Han20]. When $N_{\text{Lan}} \rightarrow N$, the cost of eigen-decomposition dominates the sorting operation. Thus, sparse Laplacian matrix design using adaptive thresholding with respect to flam yields minimally

$$N_{\text{iter}}(N(N_{\text{Lan}}^2 - N_{\text{eig}}^2) + NN_{\text{Lan}}N_{\text{eig}}),$$

where N_{iter} denotes the number of performed iterations to achieve a sparse Laplacian matrix.

Sparse Laplacian Matrix Design based on p -Nearest Neighbor Graph: In contrast to adaptive thresholding-based graph construction, computing a p -nearest neighbor graph results in considerable cost. In addition to eigen-decomposition and sorting operations, the algorithm finds p -nearest neighbors which is of complexity $N^2 \log N$ flam [CHZ07] in each iteration. Therefore, the overall computational cost of sparse Laplacian matrix design using p -nearest graph is written in terms of flam as follows

$$N_{p\text{iter}}(N(N_{\text{Lan}}^2 - N_{\text{eig}}^2) + NN_{\text{Lan}}N_{\text{eig}} + N^2 \log N).$$

B.2.3.0.3 STEP 2: ESTIMATING VECTOR \mathbf{v}

Step 2.1: Computing Candidate Block Sizes

The estimation of candidate block size matrix can mainly be attributed to changepoint detection which is a widely researched topic in the literature and there are variety of different alternatives for changepoint detection, e.g. binary segmentation (BS) or optimal partitioning (OP) approaches. In [KFE12], the computational cost of BS and OP are indicated as $O(M \log N)$ and $O(N^2)$, respectively. Then, a computationally efficient the pruned exact linear time (PELT) method has been provided. The complexity of the PELT method is $O(N)$ (under certain conditions) which can be reach $O(N^2)$ in the worst-case [KFE12].

Step 2.2: Estimating Matrix of Similarity Coefficients

Step 2.2.1: Estimating Target Similarity Coefficients

Since the eigen-decomposition of 2×2 matrix does not require a considerable time, the computational complexity of plane-based piece wise linear fitting algorithm in [YYZ19] mainly depends on covariance matrix and the mean vector estimation.

First, let consider the covariance matrix computation. For two random vectors, the covariance function includes N executions where each execution includes two addition and one multiplication. Therefore, it can be said that calculation of covariance for two random vectors requires $2N \text{fladd} + N \text{flmlt} + 1 \text{fldiv}$. As vector $\check{\mathbf{v}}$ fitting generates covariance matrix of dimension 2×2 , our covariance matrix computation necessitates $6N \text{fladd} + 3N \text{flmlt} + 3 \text{fldiv}$.

After computing covariance matrix, the mean operator including $N - 1$ additions and a division is executed two times for two vectors. Thus, mean vector computation mainly requires $2(N - 1) \text{fladd} + 2 \text{fldiv}$ and total piece-wise linear fitting complexity results in $8N - 2 \text{fladd} + 3N \text{flmlt} + 5 \text{fldiv}$.

Step 2.2.2: Estimating Undesired Similarity Coefficients

The undesired similarity coefficients' estimation consists of two consecutive steps. For every candidate size vector $\mathbf{n}_r = [N_{r_1}, N_{r_2}, \dots, N_{r_{K_{\text{cand}}}}] \in \mathbb{Z}_+^{K_{\text{cand}}}$ the proposed method first computes shifted vector whose complexity principally depends on vector of increase computation. To compute a vector of increase associated with group similarity between block i and j , the method executes N_{r_i} times where each execution includes $N_{r_j} - 1$ addition. This means that the vector of increase associated with group similarity between block i and j results in $N_{r_i}(N_{r_j} - 1) \text{fladd}$. Further, the algorithm finds median of computed vector which is of dimension $N_{r_i} \times 1$. To compute

the median, the vector can be sorted in $O(N_{r_i}\sqrt{\log N_{r_i}})$ complexity using [Han20]. When size of the j th block N_{r_j} is sufficiently large valued, the vector of increase computation dominates median operation. Therefore, computing an undesired similarity coefficient corresponds to group similarity between block i and j takes minimally $N_{r_i}(N_{r_j} - 1)$ fladd and that can be written for $i = 2, \dots, K_{\text{cand}}$ and $j = 1, \dots, i - 1$ as follows

$$\sum_{i=2}^{K_{\text{cand}}} \sum_{j=1}^{i-1} N_{r_i}(N_{r_j} - 1) \text{ fladd.}$$

To summarize, estimating the similarity coefficients matrix $\hat{\mathbf{W}}_{\text{sim}}^{(r)}$ and the vector $\hat{\mathbf{v}}^{(r)} \in \mathbb{R}^N$ associated with a candidate block size vector $\mathbf{n}_r \in \mathbb{Z}_+^{K_{\text{cand}}}$ requires minimally

$$K_{\text{cand}}(8N - 2 \text{ fladd} + 3N \text{ flmlt} + 5 \text{ fldiv}) \\ + \sum_{i=2}^{K_{\text{cand}}} \sum_{j=1}^{i-1} N_{r_i}(N_{r_j} - 1) \text{ fladd.}$$

To compute all possible $\hat{\mathbf{W}}_{\text{sim}}^{(r)} \in \mathbb{R}^{K_{\text{cand}} \times K_{\text{cand}}}$ and $\hat{\mathbf{v}}^{(r)} \in \mathbb{R}^N$ correspond to $\mathbf{N}^{(K_{\text{cand}})} = [\mathbf{n}_1, \mathbf{n}_2, \dots, \mathbf{n}_{\zeta}]^T \in \mathbb{Z}_+^{\zeta \times K_{\text{cand}}}$ for $K_{\text{cand}} = K_{\text{min}}, \dots, K_{\text{max}}$, the proposed method requires

$$\zeta \left(K_{\text{cand}}(8N - 2 \text{ fladd} + 3N \text{ flmlt} + 5 \text{ fldiv}) \\ + \sum_{i=2}^{K_{\text{cand}}} \sum_{j=1}^{i-1} N_{r_i}(N_{r_j} - 1) \text{ fladd} \right).$$

When the number of candidate block size vectors increases, the complexity of these numerous operations is significantly larger than changepoint detection using PELT method. Thus, its computational cost can be ignored in vector \mathbf{v} estimation.

Algorithm 10: Sparse Laplacian Matrix Design using Adaptive Thresholding

Input: a non-sparse affinity matrix $\ddot{\mathbf{W}} \in \mathbb{R}^{(N-N_I) \times (N-N_I)}$, initial threshold T_{ini} (optional, default is $T_{\text{ini}} = 0.5$), increasement constant T_{inc} (optional, default is $T_{\text{inc}} = 10^{-3}$)

Set $T = T_{\text{ini}}$

while $T < 1$ **do**

Compute affinity matrix $\ddot{\mathbf{W}}^{(T)}$ which is equal to $\ddot{\mathbf{W}}$ except that the similarity coefficients smaller than T in

$\ddot{\mathbf{W}}^{(T)}$ are zero, i.e. $\forall \ddot{w}_{m,n}^{(T)} < T \in \ddot{\mathbf{W}}^{(T)}, \ddot{w}_{m,n}^{(T)} = 0$ where $m = 1, \dots, N - N_I$, $n = 1, \dots, N - N_I$ and $m \neq n$

Based on obtained $\ddot{\mathbf{W}}^{(T)}$, compute $\ddot{\mathbf{D}}^{(T)}$ and $\ddot{\mathbf{L}}^{(T)}$ of dimension $\mathbb{R}^{(N-N_I) \times (N-N_I)}$

Compute $\ddot{\boldsymbol{\lambda}}^{(T)} = [\ddot{\lambda}_0^{(T)}, \ddot{\lambda}_1^{(T)}, \dots, \ddot{\lambda}_{N-N_I-1}^{(T)}] \in \mathbb{R}^{N-N_I}$ in ascending order

if $\ddot{\lambda}_1^{(T)} \cong 0$ (For detailed analysis, see Section B.2.2.) and $\forall \ddot{\lambda}_m^{(T)} \in \ddot{\boldsymbol{\lambda}}^{(T)}, \ddot{\lambda}_m^{(T)} \geq 0$ for $m = 1, \dots, N - N_I - 1$ **then**

$\ddot{\mathbf{W}} = \ddot{\mathbf{W}}^{(T)}, \ddot{\mathbf{D}} = \ddot{\mathbf{D}}^{(T)}$ and $\ddot{\mathbf{L}} = \ddot{\mathbf{L}}^{(T)}$

break;

else if $\ddot{\lambda}_1^{(T)} \neq 0$ and $\forall \ddot{\lambda}_m^{(T)} \in \ddot{\boldsymbol{\lambda}}^{(T)}, \ddot{\lambda}_m^{(T)} \geq 0$ for $m = 1, \dots, N - N_I - 1$ **then**

$\ddot{\mathbf{W}}_{\text{alt}} = \ddot{\mathbf{W}}^{(T)}, \ddot{\mathbf{D}}_{\text{alt}} = \ddot{\mathbf{D}}^{(T)}$ and $\ddot{\mathbf{L}}_{\text{alt}} = \ddot{\mathbf{L}}^{(T)}$

$T \leftarrow T + T_{\text{inc}}$

else

$T \leftarrow T + T_{\text{inc}}$

end if

end

if $\ddot{\mathbf{W}}$ does not exist **then**

if $\ddot{\mathbf{W}}_{\text{alt}}$ exists **then**

$\ddot{\mathbf{W}} = \ddot{\mathbf{W}}_{\text{alt}}$

else

error : Please start with a smaller threshold

end if

end if

Output: Estimated sparse matrices $\ddot{\mathbf{W}}, \ddot{\mathbf{D}}$ and $\ddot{\mathbf{L}}$

Algorithm 11: Sparse Laplacian Matrix Design using p -nearest Graph

Input: a non-sparse affinity matrix $\ddot{\mathbf{W}} \in \mathbb{R}^{(N-N_I) \times (N-N_I)}$, initial number of neighbors value p_{ini} (optional, default is $p_{\text{ini}} = N - 2$), decrease constant p_{dec} (optional, default is $p_{\text{dec}} = 1$), minimum number of nodes in per block N_{min} (optional, default is $N_{\text{min}} \approx \frac{N}{K_{\text{max}}} \in \mathbb{Z}_+$)

Set $p = p_{\text{ini}}$

while $p > N_{\text{min}}$ **do**

Construct affinity matrix $\ddot{\mathbf{W}}^{(p)}$ using p nearest neighbors as in [CHZ07]

Based on obtained $\ddot{\mathbf{W}}^{(p)}$, compute $\ddot{\mathbf{D}}^{(p)}$ and $\ddot{\mathbf{L}}^{(p)}$ of dimension $\mathbb{R}^{(N-N_I) \times (N-N_I)}$

Compute $\ddot{\boldsymbol{\lambda}}^{(p)} = [\ddot{\lambda}_0^{(p)}, \ddot{\lambda}_1^{(p)}, \dots, \ddot{\lambda}_{N-N_I-1}^{(p)}] \in \mathbb{R}^{N-N_I}$

if $\ddot{\lambda}_1^{(p)} \cong 0$ (For detailed analysis, see Section B.2.2.) and $\forall \ddot{\lambda}_m^{(p)} \in \ddot{\boldsymbol{\lambda}}^{(p)}, \ddot{\lambda}_m^{(p)} \geq 0$ for

$m = 1, \dots, N - N_I - 1$ **then**

$\ddot{\mathbf{W}} = \ddot{\mathbf{W}}^{(p)}, \ddot{\mathbf{D}} = \ddot{\mathbf{D}}^{(p)}$ and $\ddot{\mathbf{L}} = \ddot{\mathbf{L}}^{(p)}$

break;

else if $\ddot{\lambda}_1^{(p)} \neq 0$ and $\forall \ddot{\lambda}_m^{(p)} \in \ddot{\boldsymbol{\lambda}}^{(p)}, \ddot{\lambda}_m^{(p)} \geq 0$ for $m = 1, \dots, N - N_I - 1$ **then**

$\ddot{\mathbf{W}}_{\text{alt}} = \ddot{\mathbf{W}}^{(p)}, \ddot{\mathbf{D}}_{\text{alt}} = \ddot{\mathbf{D}}^{(p)}$ and $\ddot{\mathbf{L}}_{\text{alt}} = \ddot{\mathbf{L}}^{(p)}$

$p \leftarrow p - p_{\text{dec}}$

else

$p \leftarrow p - p_{\text{dec}}$

end if

end

if $\ddot{\mathbf{W}}$ does not exist **then**

if $\ddot{\mathbf{W}}_{\text{alt}}$ exists **then**

$\ddot{\mathbf{W}} = \ddot{\mathbf{W}}_{\text{alt}}$

else

error : Please start with a greater p_{ini}

end if

end if

Output: Estimated sparse matrices $\ddot{\mathbf{W}}, \ddot{\mathbf{D}}$ and $\ddot{\mathbf{L}}$

B.2.5 EXPERIMENTAL SETTING AND ADDITIONAL EXPERIMENTAL RESULTS

B.2.5.1 EXPERIMENTAL SETTING

FRS-BDR for Unknown Number of Blocks

Eigen-decomposition function: the generalized eigen-decomposition

Minimum number of blocks (K_{\min}): 2

Maximum number of blocks (K_{\max}): $2 \times K$

Minimum number of nodes in the blocks (N_{\min}): $\frac{N}{K_{\max}}$

Maximum number of changepoints ($N_{c_{\max}}$): $2 \times (K_{\max} - 1)$

Sparse Laplacian matrix design algorithm: p -nearest graph (For details, see Algorithm 11.)

FRS-BDR for Known Number of Blocks

Eigen-decomposition function : the generalized eigen-decomposition

Minimum number of nodes in the blocks (N_{\min}): $\frac{N}{2 \times K}$

Maximum number of changepoints $N_{c_{\max}}$: $2 \times (K - 1)$

Sparse Laplacian matrix design algorithm : p -nearest graph (For details, see Algorithm 11.)

B.2.5.2 ADDITIONAL EXPERIMENTAL RESULTS

B.2.5.2.1 MNIST DATA SET

MNIST Data Set	Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods											
	Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters											
	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	87.1	50.5-82.6	50.5-90.3	50.4-53.2	51.1-89.9	51.5-86.8	50.9-91.8	51.0-89.8	50.5-91.6	50.5-92.2	85.5	89.7
3 subjects	72.0	33.9-37.5	33.8-76.1	34.1-37.1	34.3-72.5	34.9-71.8	34.4-67.6	34.4-67.7	33.9-70.0	33.9-79.0	68.6	79.6
5 subjects	60.9	20.6-25.1	20.4-63.7	20.8-24.2	21.1-62.5	20.6-60.8	23.0-48.6	20.5-54.4	20.6-59.2	20.6-65.4	52.3	67.5
8 subjects	53.4	13.1-18.0	12.9-56.0	13.4-16.9	13.7-52.4	13.8-53.7	13.9-37.6	13.3-46.0	13.1-50.1	13.1-57.7	42.3	59.3
10 subjects	51.2	10.7-16.9	10.4-52.7	10.9-14.5	11.2-50.8	11.6-50.1	10.9-33.5	10.8-45.4	10.6-44.1	10.5-53.4	38.9	57.6
Average	64.9	25.8-36.0	25.6-67.8	25.9-29.2	26.3-65.6	26.5-64.6	26.6-55.8	26.0-60.7	25.7-63.0	25.7-69.5	57.5	70.7

Table B.3: Subspace clustering performance of different block diagonal representation approaches on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

MNIST Data Set	Detailed Computation time (t) for FRS-BDR Method			
	Step 1.1	Step 1.2	Step 1.3	Step 2
2 subjects	0.007	0.064	0.194	0.004
3 subjects	0.011	0.171	0.602	0.007
5 subjects	0.033	1.106	3.262	0.018
8 subjects	0.096	5.045	15.797	0.018
10 subjects	0.164	10.053	35.965	0.017

Table B.4: Computation time performance of FRS-BDR method on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

MNIST Data Set	Computation time (t) for Different Block Diagonal Representation Methods											
	Computation Time (t) for Optimally Tuned Regularization Parameters											
	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	0.002	0.971	1.081	3.093	3.224	0.004	0.624	0.959	4.774	13.743	0.009	0.075
3 subjects	0.003	0.666	1.681	5.168	5.517	0.007	1.580	2.166	11.782	1.565	0.017	0.189
5 subjects	0.006	1.574	4.315	14.679	14.725	0.016	7.783	7.776	244.397	8.169	0.043	1.157
8 subjects	0.013	3.766	10.936	26.520	29.069	0.040	25.090	24.991	572.827	21.823	0.116	5.159
10 subjects	0.018	6.068	16.493	34.692	64.749	0.063	44.883	55.796	748.296	35.881	0.208	10.235
Average	0.008	2.609	6.901	16.830	23.457	0.026	15.992	18.338	316.415	16.236	0.079	3.363

Table B.5: Computation time performance of different block diagonal representation approaches on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

B.2.5.2.2 USPS DATA SET

Average Clustering Accuracy (p_{acc}) for Different Block Diagonal Representation Methods												
Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters												
USPS Data Set	W^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	87.6	51.0-69.5	54.3-90.6	50.9-53.5	52.2-85.9	56.0-88.6	51.1-93.0	51.1-93.1	51.0-92.9	51.0-93.2	87.0	92.6
3 subjects	70.4	34.4-55.0	36.8-77.4	34.8-38.3	35.5-69.4	38.3-71.9	36.5-85.1	34.7-85.6	34.4-80.7	34.4-86.6	77.2	87.5
5 subjects	61.1	21.2-44.4	24.7-66.9	21.8-25.4	22.1-54.6	23.1-63.9	22.4-74.5	21.2-76.0	21.2-65.0	21.2-76.8	63.3	77.3
8 subjects	50.7	13.7-35.7	18.1-59.4	14.4-18.1	14.5-46.1	15.3-58.7	13.8-69.2	14.7-68.4	13.7-60.1	13.7-70.7	52.4	65.2
10 subjects	49.4	11.0-41.4	10.6-56.0	11.6-15.0	12.2-39.8	13.8-52.0	12.2-68.6	14.8-68.2	11.0-57.8	11.2-69.8	47.4	59.8
Average	63.9	26.3-49.2	28.9-70.1	26.7-30.1	27.3-59.2	29.3-67.0	27.2-78.1	27.3-78.2	26.3-71.3	26.3-79.4	65.5	76.5

Table B.6: Subspace clustering performance of different block diagonal representation approaches on USPS data set. The results are summarized for the similarity measure $W = X^T X$.

Detailed Computation time (t) for FRS-BDR Method				
USPS Data Set	Step 1.1	Step 1.2	Step 1.3	Step 2
2 subjects	0.003	0.015	0.049	0.002
3 subjects	0.004	0.034	0.154	0.003
5 subjects	0.007	0.108	0.401	0.005
8 subjects	0.019	0.464	1.689	0.004
10 subjects	0.032	1.110	3.444	0.119

Table B.7: Computation time performance of FRS-BDR method on USPS data set. The results are summarized for the similarity measure $W = X^T X$.

Computation time (t) for Different Block Diagonal Representation Methods												
Computation Time (t) for Optimally Tuned Regularization Parameters												
USPS Data Set	W^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	0.001	0.096	0.108	0.561	0.587	0.001	0.260	0.256	0.561	3.971	0.006	0.020
3 subjects	0.001	0.228	0.252	1.239	1.115	0.001	0.607	0.606	2.358	0.952	0.008	0.041
5 subjects	0.001	0.631	0.632	2.639	2.509	0.002	1.757	1.781	5.302	1.962	0.011	0.120
8 subjects	0.003	1.429	2.011	4.106	4.286	0.006	4.837	5.095	11.177	4.579	0.028	0.487
10 subjects	0.003	2.260	2.929	4.257	4.713	0.008	7.127	7.664	14.232	8.551	0.050	1.261
Average	0.002	0.929	1.186	2.561	2.642	0.004	2.918	3.080	6.726	4.003	0.020	0.386

Table B.8: Computation time performance of different block diagonal representation approaches on USPS data set. The results are summarized for the similarity measure $W = X^T X$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

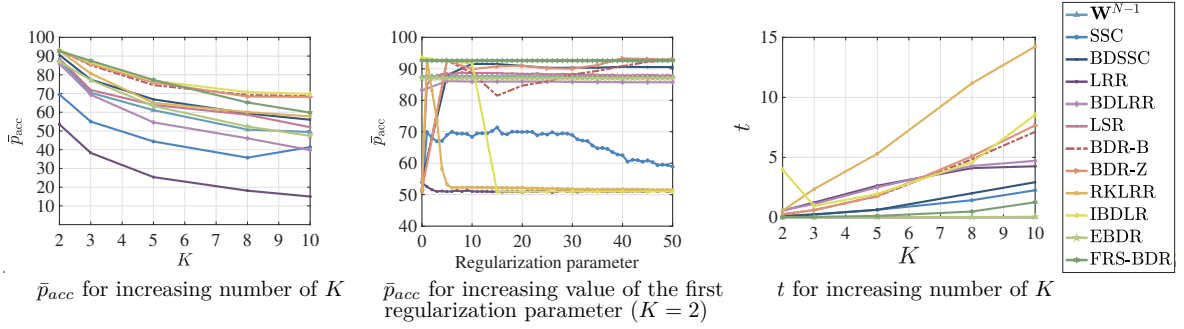


Figure B.4: Numerical results for USPS data set.

B.2.5.2.3 COIL20 DATA SET

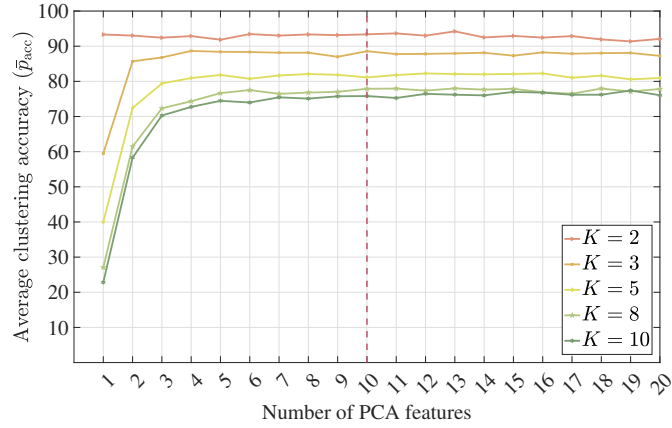


Figure B.5: Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.

Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods												
COIL20 Data Set	\mathbf{W}^{N-1}	Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters										
		SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLR	IBDLR	EBDR	FRS-BDR
2 subjects	68.9	52.5-86.6	54.5-65.9	53.5-61.0	60.8-63.6	58.0-61.8	54.3-95.9	53.3-95.6	52.5-72.9	52.5-69.7	95.8	93.1
3 subjects	42.3	36.0-83.3	39.3-54.9	39.9-41.1	58.2-67.1	44.1-47.3	38.2-89.1	37.8-88.8	36.2-66.0	35.9-72.1	90.1	88.1
5 subjects	26.4	23.0-80.6	26.2-57.6	28.9-29.7	65.1-76.4	34.3-37.0	25.3-83.3	27.1-83.1	23.1-73.6	22.8-75.4	80.9	82.5
8 subjects	17.3	15.6-75.6	18.2-63.3	21.9-22.8	64.0-73.4	25.2-28.0	18.0-75.6	21.9-75.6	15.5-71.3	15.5-74.2	72.4	77.3
10 subjects	14.6	13.1-72.5	15.6-65.3	19.2-20.3	63.7-73.0	20.6-24.9	15.3-74.5	20.0-74.2	13.1-72.0	13.0-73.4	69.1	75.9
Average	33.9	28.1-79.7	30.8-61.4	32.7-35.0	62.4-70.7	36.4-39.8	30.2-83.7	32.0-83.5	28.1-71.2	27.9-73.0	81.7	83.4

Table B.9: Subspace clustering performance of different block diagonal representation approaches on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$.

Detailed Computation time (t) for FRS-BDR Method				
COIL20 Data Set	Step 1.1	Step 1.2	Step 1.3	Step 2
2 subjects	0.002	0.004	0.014	0.002
3 subjects	0.003	0.007	0.029	0.003
5 subjects	0.006	0.014	0.071	0.007
8 subjects	0.019	0.039	0.192	0.098
10 subjects	0.032	0.064	0.319	0.206

Table B.10: Computation time performance of FRS-BDR method on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

Computation time (t) for Different Block Diagonal Representation Methods												
COIL20 Data Set	\mathbf{W}^{N-1}	Computation Time (t) for Optimally Tuned Regularization Parameters										
		SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	3×10^{-4}	0.017	0.031	0.012	0.021	3×10^{-4}	0.043	0.040	0.001	1.120	0.003	0.007
3 subjects	3×10^{-4}	0.041	0.049	0.013	0.017	2×10^{-4}	0.098	0.119	0.002	0.275	0.005	0.013
5 subjects	4×10^{-4}	0.092	0.104	0.038	0.054	3×10^{-4}	0.244	0.249	0.004	3.688	0.005	0.027
8 subjects	0.001	0.190	0.184	0.050	0.080	0.001	0.686	0.702	0.006	1.075	0.008	0.156
10 subjects	0.001	0.250	0.254	0.058	0.108	0.001	0.908	0.878	0.011	1.058	0.041	0.302
Average	4×10^{-4}	0.118	0.124	0.034	0.056	5×10^{-4}	0.396	0.397	0.005	1.443	0.012	0.101

Table B.11: Computation time performance of different block diagonal representation approaches on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

B.2.5.2.4 ORL DATA SET

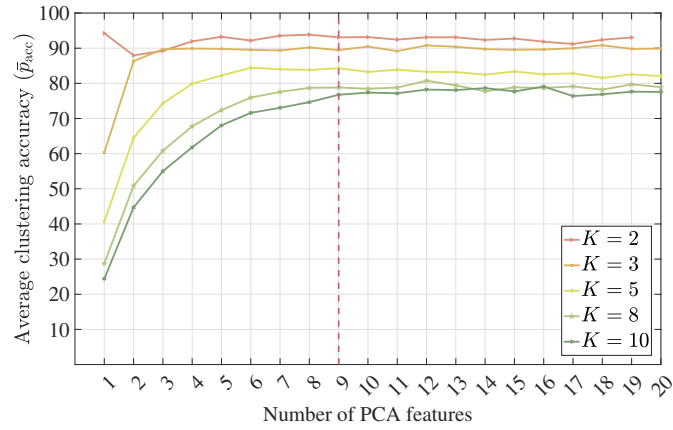


Figure B.6: Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.

Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods												
Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters												
ORL Data Set	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	56.8	55.0-81.9	55.1-64.0	55.6-57.6	59.1-69.1	57.7-60.5	55.0-95.6	55.3-95.4	55.0-73.7	x	95.2	93.1
3 subjects	41.5	38.8-80.8	43.4-50.7	42.0-44.3	58.0-59.7	45.1-46.6	38.9-90.5	40.0-90.5	39.0-62.9	38.8-69.0	90.5	90.1
5 subjects	28.4	26.0-77.1	31.5-41.0	32.0-33.3	57.0-66.7	36.3-38.9	26.2-80.1	30.8-80.1	26.1-60.6	25.7-70.8	83.0	84.2
8 subjects	21.8	18.6-74.4	25.3-35.8	26.3-27.5	56.7-68.5	26.8-28.5	18.9-79.1	34.6-78.5	18.6-75.4	18.5-75.1	74.1	79.4
10 subjects	18.7	16.3-74.2	22.7-36.1	24.1-25.1	62.7-72.8	24.0-25.3	16.4-78.4	31.2-78.9	16.5-72.5	16.1-72.4	69.9	77.2
Average	33.5	30.9-77.7	35.6-45.5	36.0-37.6	58.7-67.3	38.0-40.0	31.1-84.7	38.4-84.7	31.0-69.0	24.8-71.8	82.5	84.8

Table B.12: Subspace clustering performance of different block diagonal representation approaches on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$. 'x' denotes the failed results.

Detailed Computation time (t) for FRS-BDR Method					
ORL Data Set	Step 1.1	Step 1.2	Step 1.3	Step 2	
2 subjects	0.001	0.002	0.005	0.002	
3 subjects	0.001	0.002	0.009	0.002	
5 subjects	0.002	0.005	0.018	0.010	
8 subjects	0.004	0.010	0.044	0.119	
10 subjects	0.007	0.014	0.079	1.810	

Table B.13: Computation time performance of FRS-BDR method on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$.

Computation time (t) for Different Block Diagonal Representation Methods												
Computation Time (t) for Optimally Tuned Regularization Parameters												
ORL Data Set	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	3×10^{-4}	0.016	0.023	0.011	0.012	4×10^{-5}	0.009	0.009	2×10^{-4}	x	0.003	0.005
3 subjects	3×10^{-4}	0.018	0.025	0.012	0.014	5×10^{-5}	0.027	0.026	4×10^{-4}	0.133	0.003	0.006
5 subjects	3×10^{-4}	0.034	0.047	0.013	0.017	8×10^{-5}	0.053	0.054	0.001	0.171	0.003	0.017
8 subjects	4×10^{-4}	0.114	0.078	0.032	0.051	2×10^{-4}	0.063	0.065	0.003	2.089	0.011	0.133
10 subjects	5×10^{-4}	0.111	0.108	0.036	0.051	3×10^{-4}	0.116	0.085	0.004	3.274	0.232	1.831
Average	4×10^{-4}	0.058	0.056	0.021	0.029	2×10^{-4}	0.054	0.048	0.002	1.417	0.050	0.398

Table B.14: Computation time performance of different block diagonal representation approaches on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

B.2.5.2.5 JAFFE DATA SET

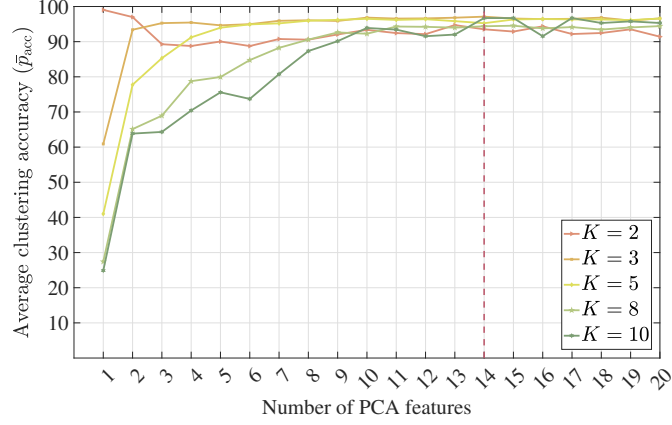


Figure B.7: Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.

Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods												
		Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters										
JAFFE Data Set	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLR	IBDLR	EBDR	FRS-BDR
2 subjects	56.1	52.1-93.3	52.8-70.4	52.8-56.1	57.1-59.0	53.4-57.1	52.3-99.7	52.4-99.7	52.4-82.0	52.0-79.6	94.9	93.4
3 subjects	39.7	36.2-95.2	37.1-59.7	37.8-40.8	49.0-65.3	39.8-45.1	36.1-98.0	36.5-98.0	36.2-76.7	35.9-75.5	92.4	96.9
5 subjects	26.4	23.3-95.3	26.4-76.5	27.9-28.9	60.0-85.0	33.1-37.8	23.2-94.2	28.3-94.2	23.3-69.0	23.1-88.0	90.7	96.4
8 subjects	19.1	15.8-93.6	18.6-91.9	21.8-22.6	80.3-89.6	24.2-38.1	15.8-94.3	34.2-94.3	15.8-87.4	15.7-94.2	82.4	94.4
10 subjects	x	13.1-93.0	16.4-93.9	17.4-25.4	85.9-92.0	11.7-55.9	12.2-91.5	25.8-91.5	13.1-92.0	12.7-94.8	77.9	96.7
Average	35.3	28.1-94.1	30.3-78.5	31.5-34.8	66.5-78.2	32.4-46.8	27.9-95.6	35.4-95.6	28.2-81.4	27.9-86.4	87.7	95.5

Table B.15: Subspace clustering performance of different block diagonal representation approaches on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$. 'x' denotes the failed results.

Detailed Computation time (t) for FRS-BDR Method				
JAFFE Data Set	Step 1.1	Step 1.2	Step 1.3	Step 2
2 subjects	0.002	0.004	0.015	0.002
3 subjects	0.003	0.008	0.034	0.003
5 subjects	0.007	0.016	0.087	0.013
8 subjects	0.022	0.047	0.245	0.298
10 subjects	0.043	0.076	0.469	0.777

Table B.16: Computation time performance of FRS-BDR method on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$.

Computation time (t) for Different Block Diagonal Representation Methods												
Computation Time (t) for Optimally Tuned Regularization Parameters												
JAFFE Data Set	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	3×10^{-4}	0.022	0.032	0.015	0.018	8×10^{-5}	0.060	0.060	0.006	0.505	0.004	0.008
3 subjects	4×10^{-4}	0.053	0.044	0.050	0.056	3×10^{-4}	0.106	0.108	0.002	1.794	0.004	0.015
5 subjects	4×10^{-4}	0.109	0.109	0.066	0.087	4×10^{-4}	0.246	0.247	0.004	0.563	0.005	0.037
8 subjects	5×10^{-4}	0.215	0.216	0.068	0.129	0.001	0.561	0.555	0.010	0.972	0.036	0.367
10 subjects	6×10^{-4}	0.291	0.235	0.063	0.120	0.001	0.625	0.635	0.013	1.044	0.083	0.896
Average	4×10^{-4}	0.138	0.127	0.052	0.082	4×10^{-4}	0.320	0.321	0.007	0.976	0.026	0.264

Table B.17: Computation time performance of different block diagonal representation approaches on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

B.2.5.2.6 YALE DATA SET

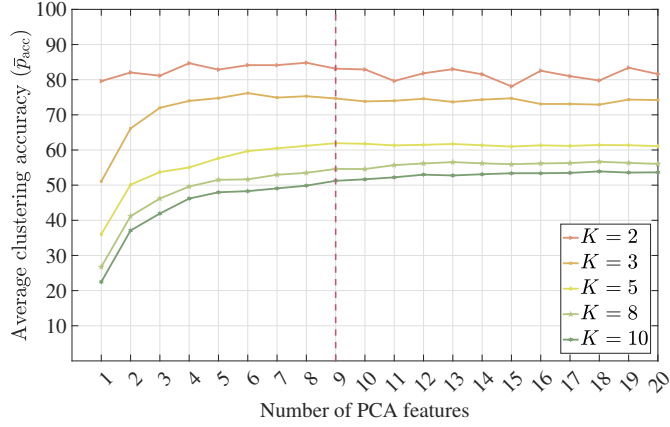


Figure B.8: Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.

Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods												
Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters												
Yale Data Set	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects	55.3	54.5-75.6	54.7-58.8	55.5-60.2	57.5-60.9	54.5-57.3	53.6-84.2	54.5-83.9	54.5-60.4	x	84.1	83.1
3 subjects	41.9	38.4-67.2	43.0-46.7	42.3-43.9	50.9-53.6	42.1-43.8	38.6-68.5	39.0-68.6	38.6-52.4	38.2-55.1	71.6	74.5
5 subjects	29.8	25.5-57.9	30.5-37.4	32.0-33.2	49.2-54.4	31.1-33.2	25.7-62.3	29.9-62.2	25.4-49.9	25.2-52.9	60.2	61.7
8 subjects	21.6	18.1-52.6	22.3-29.7	25.7-26.7	47.3-50.7	23.6-25.4	18.3-51.9	28.5-51.5	18.3-48.6	17.9-49.6	53.8	54.8
10 subjects	18.7	15.6-50.0	20.2-30.2	23.4-24.7	46.2-47.9	20.8-22.0	15.8-51.6	24.5-51.6	15.9-46.3	15.4-46.3	49.2	51.2
Average	33.5	30.4-60.7	34.1-40.6	35.8-37.7	50.2-53.5	34.4-36.3	30.4-63.7	35.3-63.6	30.5-51.5	24.2-51.0	63.8	65.1

Table B.18: Subspace clustering performance of different block diagonal representation approaches on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. 'x' denotes the failed results.

Detailed Computation time (t) for FRS-BDR Method					
Yale Set	Data	Step 1.1	Step 1.2	Step 1.3	Step 2
2 subjects		0.001	0.002	0.007	0.002
3 subjects		0.001	0.003	0.013	0.003
5 subjects		0.002	0.006	0.025	0.013
8 subjects		0.005	0.012	0.056	0.046
10 subjects		0.008	0.018	0.099	0.160

Table B.19: Computation time performance of FRS-BDR method on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

Computation time (t) for Different Block Diagonal Representation Methods													
Computation Time (t) for Optimally Tuned Regularization Parameters													
Yale Set	Data	\mathbf{W}^{N-1}	SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
2 subjects		3×10^{-4}	0.011	0.018	0.013	0.011	5×10^{-5}	0.010	0.010	2×10^{-4}	x	0.003	0.005
3 subjects		4×10^{-4}	0.022	0.023	0.010	0.011	6×10^{-5}	0.039	0.049	4×10^{-4}	0.059	0.003	0.007
5 subjects		3×10^{-4}	0.039	0.040	0.013	0.017	1×10^{-4}	0.043	0.038	0.002	0.276	0.005	0.021
8 subjects		4×10^{-4}	0.093	0.090	0.031	0.053	3×10^{-4}	0.055	0.053	0.003	0.705	0.006	0.063
10 subjects		4×10^{-4}	0.138	0.126	0.036	0.062	4×10^{-4}	0.061	0.069	0.004	1.077	0.022	0.186
Average		4×10^{-4}	0.061	0.059	0.021	0.031	2×10^{-4}	0.042	0.044	0.002	0.529	0.008	0.056

Table B.20: Computation time performance of different block diagonal representation approaches on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

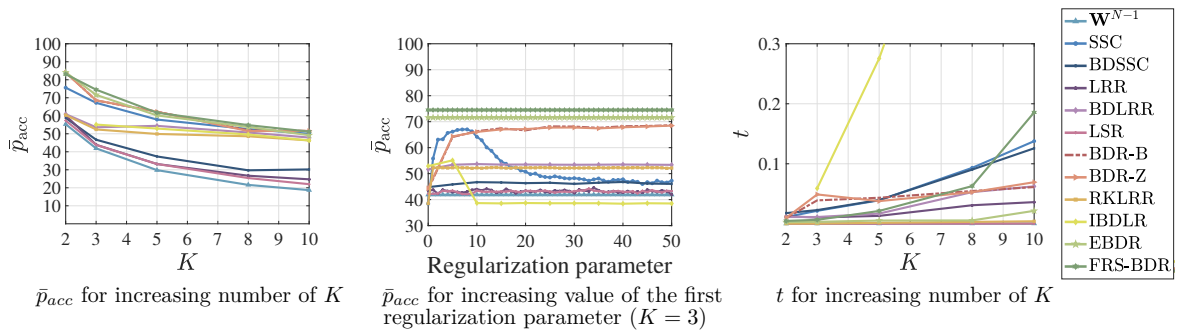


Figure B.9: Numerical results for Yale data set.

B.2.5.2.7 COMPARISONS WITH POPULAR BLOCK DIAGONAL REPRESENTATION APPROACHES BASED ON ADDITIONAL CLUSTERING DATA SETS

Estimated Parameters of FRS-BDR for Different Clustering Data Sets											
Data Set	N	K	\hat{K}	\bar{p}_{acc}	$\mathbf{n} = [n_1, n_2, \dots, n_K]^T$	$\hat{\mathbf{n}} = [\hat{n}_1, \hat{n}_2, \dots, \hat{n}_{\hat{K}}]^T$	Detailed Computation Time (t)				Estimation Error $\ \mathbf{v} - \hat{\mathbf{v}}\ $
							Step 1.1	Step 1.2	Step 1.3	Step 2	
Breast Cancer [WM89],	569	2	2	90.128	$[212, 357]^T$	$[173, 396]^T$	0.038	1.584	4.674	0.021	604.387
Ceramic [HZZ16][HZZ16],	88	2	2	98.864	$[44, 44]^T$	$[44, 44]^T$	0.002	0.010	0.031	0.007	2.4271
Vertebral Column [RSB11],	310	2,3	2	75.784	$[100, 210]^T$	$[120, 190]^T$	0.009	0.183	0.543	0.011	140.272
Fisheriris [Fis36],	150	3	3	96.667	$[50, 50, 50]^T$	$[51, 49, 50]^T$	0.003	0.033	0.098	0.018	48.121
Gait [SAZ19],	800	5	5	77.125	$[160, 160, 160, 160, 160]^T$	$[100, 128, 166, 201, 205]^T$	0.087	5.011	15.327	1.139	466.761
O. Cancer [CFR04],	216	2	2	77.315	$[95, 121]^T$	$[65, 151]^T$	0.012	0.075	0.208	0.011	131.006
Person Id. [TSM18],	187	4	4	96.791	$[38, 40, 47, 62]^T$	$[34, 36, 52, 65]^T$	0.027	0.053	0.190	0.108	70.456
Parkinson A. [NPC16],	240	2	2	58.208	$[120, 120]^T$	$[78, 162]^T$	0.006	0.095	0.266	0.013	111.626

Table B.21: Estimation performance of FRS-BDR on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$.

Average Clustering Accuracy (\bar{p}_{acc}) for Different Block Diagonal Representation Methods												
Data Set	\mathbf{W}^{N-1}	Minimum-Maximum Clustering Accuracy ($p_{accmin} - p_{accmax}$) for Different Regularization Parameters										
		SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
Breast Cancer [WM89],	88.2	51.0-74.7	50.3-88.2	54.3-90.3	88.0-90.0	73.5-88.2	62.4-90.0	52.9-90.2	62.6-91.7	60.3-90.0	85.2	90.1
Ceramic [HZZ16],	98.9	51.1-98.9	51.1-100	95.5-98.9	95.5-98.9	54.5-98.9	51.1-100	51.1-98.9	51.1-95.5	51.1-98.9	98.9	98.9
Vertebral Column [RSB11],	73.2	50.0-77.7	50.3-74.8	53.9-72.6	72.6-72.6	62.6-75.8	67.4-76.8	71.9-76.8	67.4-71.3	67.4-76.1	74.8	75.8
Fisheriris [Fis36],	78.0	34.7-82.7	34.0-83.3	38.7-80.7	80.0-98.0	78.0-82.7	34.0-96.7	65.3-96.7	34.0-80.0	34.7-84.0	98.0	96.7
Gait [SAZ19],	77.3	20.3-77.4	20.1-77.5	26.1-83.9	78.9-83.5	55.4-75.9	20.3-84.8	26.4-84.5	20.5-85.5	20.4-81.6	81.1	77.1
O. Cancer [CFR04],	61.7	51.4-73.6	50.9-71.3	52.3-76.4	54.2-76.4	51.9-66.2	53.7-75.9	51.9-74.1	55.6-88.4	55.6-75.5	77.8	77.3
Person Id. [TSM18],	x	33.7-96.8	31.6-95.7	49.7-94.7	71.1-94.7	33.2-64.2	31.6-96.3	59.4-95.7	34.2-94.1	33.7-95.7	97.3	96.8
Parkinson A. [NPC16],	61.3	50.4-58.8	50.0-61.3	50.4-54.2	50.4-61.3	57.9-61.3	50.4-61.3	50.0-61.3	50.4-61.7	50.4-61.3	56.7	58.2
Average	76.9	42.8-80.1	42.3-81.5	52.6-81.4	73.8-84.4	58.4-76.6	46.4-85.2	53.6-84.8	47.0-83.5	46.7-82.9	83.7	83.9

Table B.22: Subspace clustering performance of different block diagonal representation approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$. 'x' denotes the failed results due to the complex-valued eigenvectors.

Computation time (t) for Different Block Diagonal Representation Methods												
Data Set	\mathbf{W}^{N-1}	Computation Time (t) for Optimally Tuned Regularization Parameters										
		SSC	BDSSC	LRR	BDLRR	LSR	BDR-B	BDR-Z	RKLRR	IBDLR	EBDR	FRS-BDR
Breast Cancer [WM89],	0.003	1.123	2.967	0.456	1.034	0.009	4.510	4.502	2.793	10.782	0.116	1.643
Ceramic [HZZ16],	4×10^{-4}	0.074	0.076	0.074	0.086	3×10^{-4}	0.139	0.137	0.174	0.918	0.007	0.019
Vertebral Column [RSB11],	0.001	0.518	0.490	0.122	0.230	0.002	0.897	0.902	0.031	2.018	0.038	0.203
Fisheriris [Fis36],	5×10^{-4}	0.161	0.166	0.070	0.066	0.001	0.030	0.025	0.024	9.330	0.012	0.054
Gait [SAZ19],	0.006	6.465	6.642	1.214	2.877	0.018	1.377	1.212	7.392	865.055	0.583	6.237
O. Cancer [CFR04],	0.008	3.036	3.883	8.316	8.387	0.016	0.503	0.506	9.192	17.590	0.026	0.098
Person Id. [TSM18],	5×10^{-4}	0.234	0.236	0.039	0.125	0.001	0.905	0.485	0.012	0.873	0.017	0.188
Parkinson A. [NPC16],	0.001	0.350	0.354	0.321	0.399	0.001	0.066	0.065	0.170	2.397	0.025	0.113
Average	0.002	1.495	1.851	1.327	1.651	0.006	1.053	0.979	2.473	113.620	0.103	1.069

Table B.23: Computation time performance of different block diagonal representation approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^T \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.

B.2.5.2.8 COMPARISONS WITH POPULAR COMMUNITY DETECTION APPROACHES BASED ON ADDITIONAL CLUSTERING DATA SETS

Data Set	\hat{K} for Different Cluster Enumeration Methods								K
	Louvain	Martelot	BNMF	DenPeak	Combo	MAP	Sparcode	FRS-BDR	
Breast Cancer [WM89],	2	1	1	3	2	1	2	2	2
Ceramic [HZZ16],	2	2	1	3	2	1	2	2	2
Vertebral Column [RSB11],	2	2	1	3	2	1	2	2	2,3
Fisheriris [Fis36],	2	2	1	3	2	1	1	3	3
Gait [SAZ19],	3	2	1	2	3	1	2	5	5
O. Cancer [CFR04],	2	2	1	3	2	1	4	2	2
Person Id. [TSM18],	3	3	1	47	2	2	3	4	4
Parkinson A. [NPC16],	1	1	1	3	1	1	2	2	2

Table B.24: Performance of different cluster enumeration approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

Data Set	mod for Different Cluster Enumeration Methods							
	Louvain	Martelot	BNMF	DenPeak	Combo	MAP	Sparcode	FRS-BDR
Breast Cancer [WM89],	0.001	0.000	0.000	0.000	0.001	0.000	0.022	0.345
Ceramic [HZZ16],	0.055	0.055	0.000	0.040	0.055	0.000	0.440	0.441
Vertebral Column [RSB11],	0.015	0.015	0.000	0.000	0.015	0.000	0.330	0.418
Fisheriris [Fis36],	0.016	0.016	0.000	0.016	0.016	0.000	0.000	0.472
Gait [SAZ19],	0.017	0.014	0.000	0.000	0.017	0.000	0.375	0.641
O. Cancer [CFR04],	0.006	0.006	0.000	0.005	0.006	0.000	0.019	0.334
Person Id. [TSM18],	0.128	0.124	0.000	0.051	0.128	0.000	0.515	0.694
Parkinson A. [NPC16],	0.000	0.000	0.000	0.000	0.000	0.000	0.179	0.342
Average	0.030	0.029	0.000	0.014	0.030	0.000	0.235	0.461

Table B.25: Partitioning performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for mod using the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

Data Set	Conductance (cond) for Different Cluster Enumeration Methods							
	Louvain	Martelot	BNMF	DenPeak	Combo	MAP	Sparcode	FRS-BDR
Breast Cancer [HZZ16],	0.448	0.000	0.000	0.007	0.448	0.000	0.079	0.098
Ceramic [HZZ16],	0.445	0.445	0.000	0.449	0.445	0.000	0.058	0.059
Vertebral Column [RSB11],	0.484	0.484	0.000	0.022	0.484	0.000	0.119	0.079
Fisheriris [Fis36],	0.424	0.429	0.000	0.433	0.424	0.000	0.000	0.183
Gait [SAZ19],	0.615	0.479	0.000	0.003	0.615	0.000	0.121	0.138
O. Cancer [CFR04],	0.451	0.476	0.000	0.487	0.451	0.000	0.047	0.140
Person Id. [TSM18],	0.296	0.294	0.000	0.660	0.296	-0.035	0.095	0.047
Parkinson A. [NPC16],	0.000	0.000	0.000	0.017	0.000	0.000	0.312	0.157
Average	0.395	0.326	0.000	0.260	0.395	-0.004	0.104	0.113

Table B.26: Partitioning performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for conductance (cond) using the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$.

Data Set	Computation Time (t) for Different Cluster Enumeration Methods							
	Louvain	Martelot	BNMF	DenPeak	Combo	MAP	Sparcode	FRS-BDR
Breast Cancer [WM89],	0.270	0.547	22.839	0.080	3.965	2.016	3.644	5.334
Ceramic [HZZ16],	0.008	0.004	0.273	0.009	0.083	0.219	0.103	0.059
Vertebral Column [RSB11],	0.059	0.066	4.154	0.036	1.040	0.719	0.820	0.818
Fisheriris [Fis36],	0.012	0.010	0.752	0.017	0.241	0.375	0.323	0.178
Gait [SAZ19],	0.629	1.654	48.883	0.118	3.706	2.766	5.723	21.321
O. Cancer [CFR04],	0.026	0.023	1.769	0.024	0.546	0.484	0.509	0.337
Person Id. [TSM18],	0.027	0.016	1.363	0.047	0.366	0.344	0.420	0.366
Parkinson A. [NPC16],	0.029	0.032	2.307	0.027	0.003	0.578	0.540	0.425
Average	0.132	0.294	10.293	0.045	1.243	0.938	1.510	3.605

Table B.27: Computation performance of different cluster enumeration approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and FRS-BDR is detailed for all steps.

B.3 ADDITIONAL INFORMATION FOR RRLPI

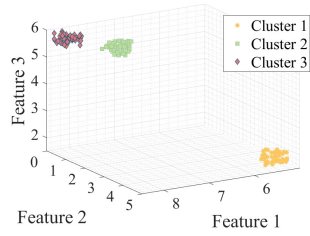
B.3.1 EXPERIMENTAL SETTING

Group Information	Feature Space Parameters		Number of samples
	$\boldsymbol{\mu}$	\mathcal{D}	
Cluster-1 (c_1)	$\boldsymbol{\mu}_{c_1}=[5.50; 4.50; 2.00; 0.75; 2.50; 4.50]$	$\mathcal{D}_{c_1} = 0.50$	$N_{c_1} = 50$ for $N_{\text{out}} = 0$
Cluster-2 (c_2)	$\boldsymbol{\mu}_{c_2}=[7.50; 1.00; 5.50; 2.50; 1.00; 1.50]$	$\mathcal{D}_{c_2} = 0.50$	$N_{c_2} = 50$ for $N_{\text{out}} = 0$
Cluster-3 (c_3)	$\boldsymbol{\mu}_{c_3}=[8.50; 0.75; 6.00; 4.50; 1.50; 1.25]$	$\mathcal{D}_{c_3} = 0.50$	$N_{c_3} = 50$ for $N_{\text{out}} = 0$
Type II Outliers (o_2)	$\boldsymbol{\mu}_2=[7.00; 0.25; 5.00; 2.00; 0.50; 0.75]$	$\mathcal{D}_2 = 1.50$	$N_{\text{out}} \in \{0, 5, 10, \dots, 20\}$

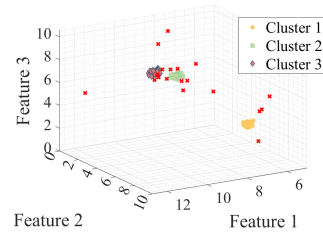
Table B.28: Detailed numerical information for the synthetic data set

- The original m th feature vector $\mathbf{x}_{m,i}$ associated with the i th cluster c_i , such that $i = 1, \dots, K$ and $m = 1, \dots, N_i$, is computed as $\mathbf{x}_{m,i} = \boldsymbol{\mu}_{c_i} + \mathcal{D}_{c_i} \boldsymbol{v}$ where \boldsymbol{v} denotes a vector of independently and identically distributed random variables from a uniform distribution on the interval $[-0.5, 0.5]$.
- If $\mathbf{x}_{m,i}$ is replaced with a Type I outlier, the outlier $\tilde{\mathbf{x}}_m^{(1)}$ is computed as $\tilde{\mathbf{x}}_m^{(1)} = \mathbf{x}_{m,i} + \mathcal{D}_1 \boldsymbol{v}$ where $\mathcal{D}_1 \in \{0, 1, 2, \dots, 10\}$.
- If $\mathbf{x}_{m,i}$ is replaced with a Type II outlier, the outlier $\tilde{\mathbf{x}}_m^{(2)}$ is computed as $\tilde{\mathbf{x}}_m^{(2)} = \boldsymbol{\mu}_2 + \mathcal{D}_2 \boldsymbol{v}$.

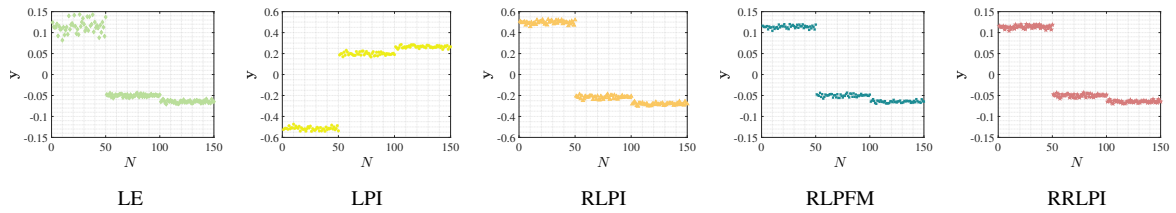
B.3.2 ADDITIONAL RESULTS FOR SYNTHETIC DATA SETS



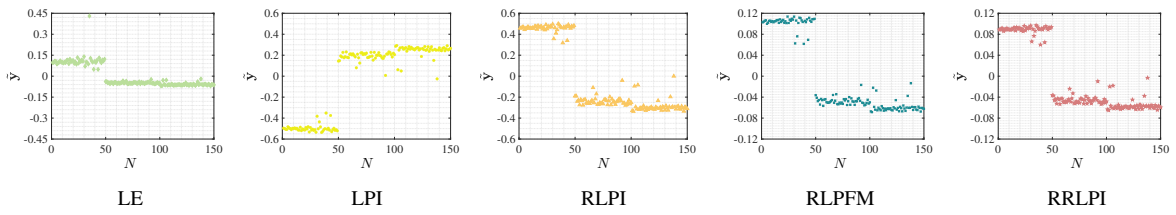
Exemplary plot of the first three features of the uncorrupted synthetic data set.



Exemplary plot of the first three features of the synthetic data set after corruption with Type I and Type II outliers (red crosses).



Estimated eigenvectors for the uncorrupted data set.



Estimated eigenvectors for the corrupted data set.

Figure B.10: Estimated eigenvectors for the uncorrupted and corrupted data sets.

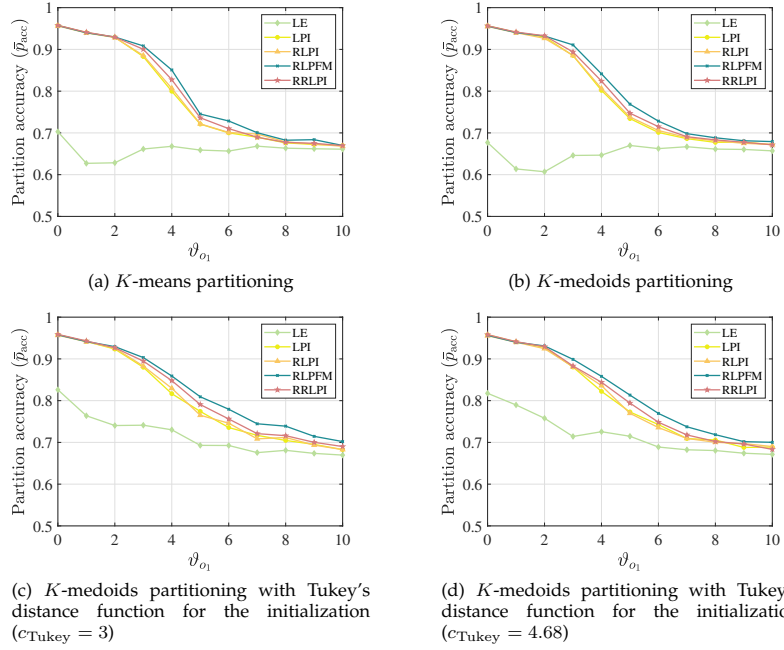


Figure B.11: \bar{p}_{acc} performance of different partitioning methods for increasing \mathcal{D}_1 associated with Type I outlier ($N = 300, N_{\text{out}} = 10, \mathcal{D}_2 = 1.5, \mathcal{D}_{c_K} = 0.5$ s.t. $K = 1, \dots, K$)

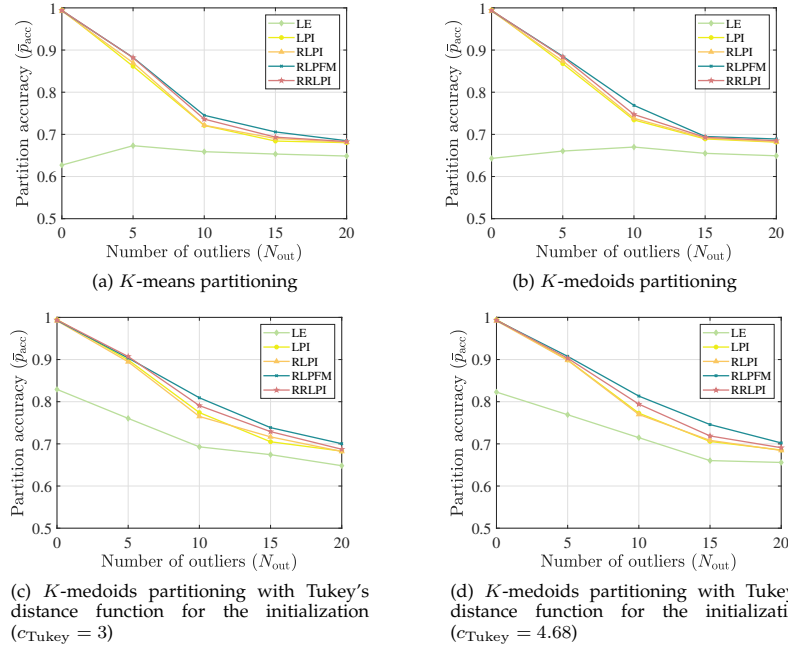


Figure B.12: \bar{p}_{acc} performance of different partitioning methods for each of outlier type with increasing N_{out} ($N = 300, \mathcal{D}_1 = 5, \mathcal{D}_2 = 1.5, \mathcal{D}_{c_K} = 0.5$ s.t. $K = 1, \dots, K$)

B.3.3 ADDITIONAL RESULTS FOR CLUSTER ENUMERATION

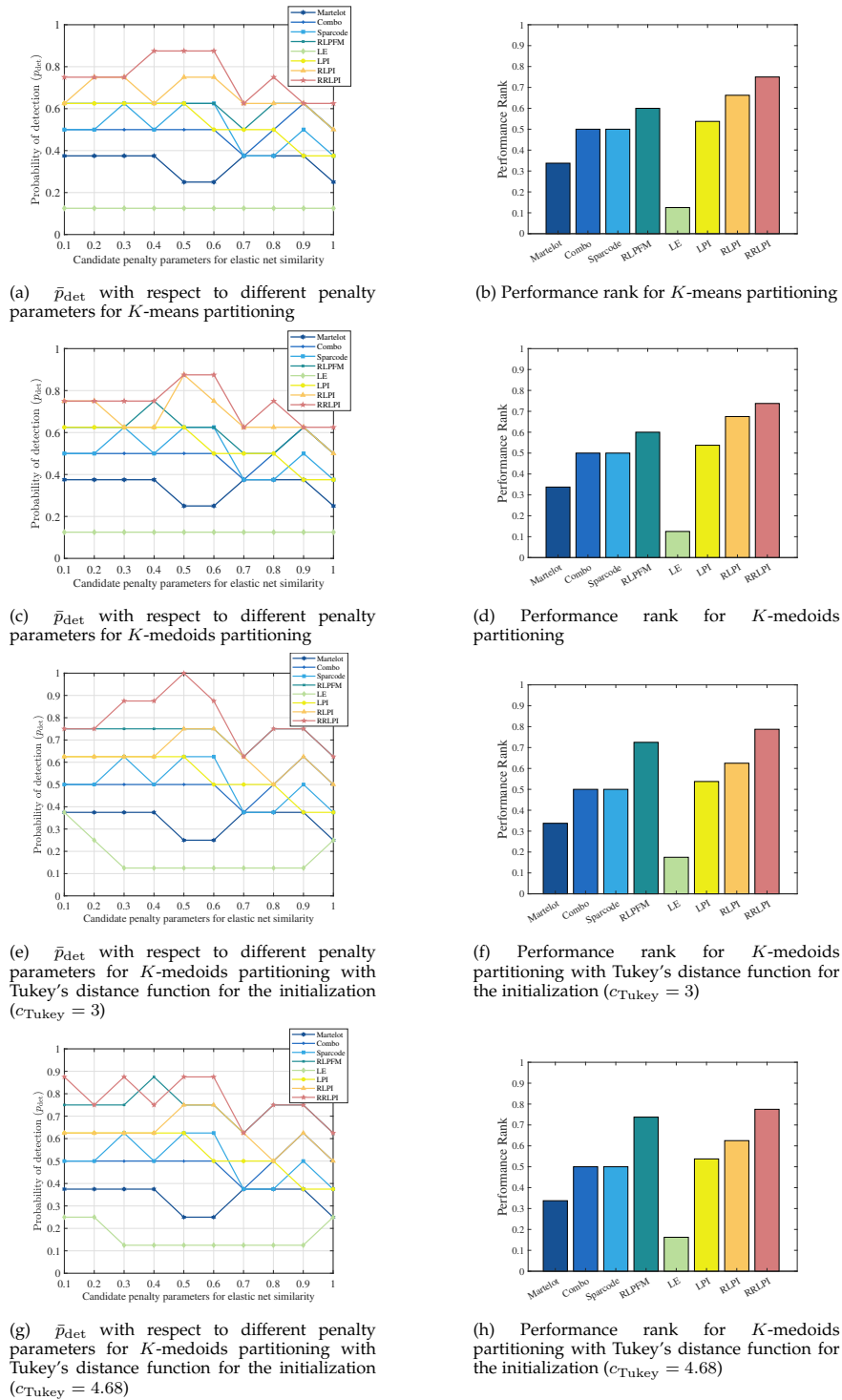


Figure B.13: Numerical results for cluster enumeration using different partitioning algorithms.

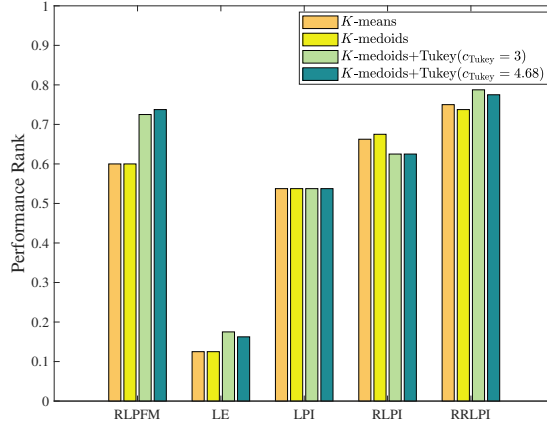


Figure B.14: Overall performance rank for different partitioning algorithms.

Performance Analysis for Different Values of Huber's Tuning Parameter								
Data Set	$c = 2$ (99% ARE)		$c = 1.345$ (95% ARE)		$c = 0.73175$ (85% ARE)		K	Similarity
	\hat{K}	mod	\hat{K}	mod	\hat{K}	mod		
Gait [SAZ19],	4	0.5331	4	0.5462	4	0.5442	5	enet
Breast Cancer [WM89],	2	0.0004	2	0.0005	2	0.0005	2	cos
Fisheriris [Fis36],	3	0.4985	3	0.4985	3	0.4985	3	enet
Person Id. [TSM18],	4	0.4015	4	0.4079	4	0.3911	4	enet
Sonar [GS88],	2	0.0194	2	0.0199	2	0.0203	2	cos
Ionosphere [SWH89],	2	0.0883	2	0.1196	6	0.0145	2	cos
D. Retinopathy [AH14],	2	0.1088	2	0.1080	2	0.1080	2	cos
Gesture Phase S. [WPM14],	5	0.0040	5	0.0042	5	0.0040	5	cos

Table B.29: Performance for different c values on well-known clustering data sets. The results are summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\rho = 0.5$. The partitioning is determined by K -means.

Performance Analysis for Different Values of Huber's Tuning Parameter								
Data Set	$c = 2$ (99% ARE)		$c = 1.345$ (95% ARE)		$c = 0.73175$ (85% ARE)		K	Similarity
	\hat{K}	mod	\hat{K}	mod	\hat{K}	mod		
Gait [SAZ19],	4	0.5323	5	0.5480	4	0.5442	5	enet
Breast Cancer [WM89],	2	0.0004	2	0.0005	2	0.0005	2	cos
Fisheriris [Fis36],	3	0.4985	3	0.4985	3	0.4985	3	enet
Person Id. [TSM18],	4	0.4015	4	0.4074	4	0.3945	4	enet
Sonar [GS88],	2	0.0188	2	0.0195	2	0.0203	2	cos
Ionosphere [SWH89],	2	0.0865	2	0.1196	6	0.0134	2	cos
D. Retinopathy [AH14],	2	0.1088	2	0.1080	2	0.1080	2	cos
Gesture Phase S. [WPM14],	5	0.0040	5	0.0041	5	0.0041	5	cos

Table B.30: Performance for different c values on well-known clustering data sets. The results are summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\rho = 0.5$. The partitioning is determined by K -medoids with Tukey's distance function for the initialization ($c_{Tukey} = 3$).

Data Set	\hat{K} for Different Cluster Enumeration Methods									Similarity
	Martelot	Combo	Sparcode	RLPFM	LE	LPI	RLPI	RRLPI	K	
Gait [SAZ19],	4	6	5	4	4	4	4	5	5	enet
Breast Cancer [WM89],	1	2	2	2	4	2	2	2	2	Pearson
Fisheriris [Fis36],	2	3	2	3	5	3	3	3	3	enet
Person Id. [TSM18],	6	7	4	5	10	4	4	4	4	enet
Sonar [GS88],	2	2	2	2	5	2	2	2	2	Pearson
Ionosphere [SWH89],	3	3	3	3	10	3	2	2	2	Pearson
D. Retinopathy [AH14],	2	2	2	2	2	2	2	2	2	Pearson
Gesture Phase S. [WPM14],	2	4	3	3	4	2	4	5	5	Pearson

Table B.31: Performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -medoids with Tukey's distance function [ZKO18] for the initialization ($c_{\text{Tukey}} = 3$).

Data Set	$\text{mod}_{\hat{K}}$ for Different Cluster Enumeration Methods									Similarity
	Martelot	Combo	Sparcode	RLPFM	LE	LPI	RLPI	RRLPI	K	
Gait [SAZ19],	0.612	0.669	0.459	0.534	0.570	0.551	0.544	0.548	enet	
Breast Cancer [WM89],	0.000	0.001	0.047	0.001	0.000	0.001	0.001	0.001	Pearson	
Fisheriris [Fis36],	0.440	0.502	0.282	0.496	0.451	0.498	0.499	0.499	enet	
Person Id. [TSM18],	0.706	0.727	0.119	0.414	0.406	0.516	0.405	0.407	enet	
Sonar [GS88],	0.058	0.058	0.172	0.041	0.036	0.058	0.040	0.045	Pearson	
Ionosphere [SWH89],	0.206	0.213	0.340	0.148	0.144	0.114	0.193	0.194	Pearson	
D. Retinopathy [AH14],	0.178	0.178	0.485	0.176	0.171	0.178	0.177	0.176	Pearson	
Gesture Phase S. [WPM14],	0.137	0.157	0.193	0.024	0.049	0.043	0.020	0.019	Pearson	

Table B.32: Modularity performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -medoids with Tukey's distance function [ZKO18] for the initialization ($c_{\text{Tukey}} = 3$).

Data Set	\hat{K} for Different Cluster Enumeration Methods									Similarity
	Martelot	Combo	Sparcode	RLPFM	LE	LPI	RLPI	RRLPI	K	
Gait [SAZ19],	4	6	5	4	4	4	4	5	4	enet
Breast Cancer [WM89],	1	2	2	2	4	2	2	2	2	Pearson
Fisheriris [Fis36],	2	3	2	3	5	3	3	3	3	enet
Person Id. [TSM18],	6	7	4	5	10	4	4	4	4	enet
Sonar [GS88],	2	2	2	2	5	2	2	2	2	Pearson
Ionosphere [SWH89],	3	3	3	3	10	3	2	2	2	Pearson
D. Retinopathy [AH14],	2	2	2	2	2	2	2	2	2	Pearson
Gesture Phase S. [WPM14],	2	4	3	3	5	2	4	5	5	Pearson

Table B.33: Performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -means.

mod _k for Different Cluster Enumeration Methods									
Data Set	Martelot	Combo	Sparcode	RLPFMLE	LPI	RLPI	RRLPI	Similarity	
Gait [SAZ19],	0.612	0.669	0.459	0.530	0.570	0.551	0.538	0.530	enet
Breast Cancer [WM89],	0.000	0.001	0.047	0.001	0.000	0.001	0.001	0.001	Pearson
Fisheriris [Fis36],	0.440	0.502	0.282	0.498	0.451	0.498	0.498	0.498	enet
Person Id. [TSM18],	0.706	0.727	0.119	0.410	0.406	0.516	0.405	0.407	enet
Sonar [GS88],	0.058	0.058	0.172	0.041	0.037	0.058	0.041	0.047	Pearson
Ionosphere [SWH89],	0.206	0.213	0.340	0.143	0.144	0.117	0.193	0.194	Pearson
D. Retinopathy [AH14],	0.178	0.178	0.485	0.050	0.043	0.049	0.049	0.049	Pearson
Gesture Phase S. [WPM14],	0.137	0.157	0.193	0.064	0.063	0.042	0.020	0.019	Pearson

Table B.34: Modularity performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -means.

B.3.4 ADDITIONAL RESULTS FOR IMAGE SEGMENTSTION











I_{seg}	Index	K	Cluster Index	Annotation	I_{seg} Color Code
	$I_{\text{seg}}^{(1)}$	2	1 2	[90 116] [0 198]	[1.000 1.000 1.000] [0.612 0.812 0.902]
	$I_{\text{seg}}^{(2)}$	2	1 2	[160 150] [6 230]	[0.490 0.278 0.243] [0.612 0.812 0.902]
	$I_{\text{seg}}^{(3)}$	2	1 2	[140 120] [6 230]	[0.902 0.902 0.902] [0.612 0.812 0.902]
	$I_{\text{seg}}^{(4)}$	2	1 2	[0 0] [143 255]	[0.902 0.902 0.902] [0.490 0.278 0.243]
	$I_{\text{seg}}^{(5)}$	3	1 2 3	[6 230] [112 9] [4 200]	[0.612 0.812 0.902] [0.910 0.910 0.490] [0.129 0.478 0.310]
	$I_{\text{seg}}^{(6)}$	3	1 2 3	[4 200] [4 250] [6 230]	[0.129 0.478 0.310] [0.635 0.800 0.431] [0.612 0.812 0.902]
	$I_{\text{seg}}^{(7)}$	4	1 2 3 4	[9 7] [6 230] [4 200] [160 150]	[0.173 0.416 0.580] [0.612 0.812 0.902] [0.129 0.478 0.310] [0.839 0.788 0.663]
	$I_{\text{seg}}^{(8)}$	2	1 2	[4 250] [255 0]	[0.635 0.800 0.431] [0.969 0.929 0.843]
	$I_{\text{seg}}^{(9)}$	2	1 2	[140 120] [6 230]	[0.839 0.788 0.663] [0.612 0.812 0.902]
	$I_{\text{seg}}^{(10)}$	4	1 2 3 4	[4 250] [4 200] [143 255] [0 0]	[0.635 0.800 0.431] [0.129 0.478 0.310] [0.612 0.812 0.902] [0.490 0.278 0.243]

Table B.35: Reference annotations for the ground truth images.





Image	Index	K	Performance Measure	FastEFM	LSC	RLPFM	LE	LPI	RLPI	RRLPI	ξ	
	$I^{(1)}$	2	\bar{F}_{score}	0.297	0.905	0.906	0.294	0.908	0.905	0.867	0.178	
			\bar{J}	0.421	0.867	0.867	0.421	0.868	0.868	0.854		
			\bar{p}_{acc}	0.841	0.960	0.960	0.841	0.961	0.960	0.957		
	$I^{(2)}$	2	\bar{F}_{score}	0.390	0.949	0.696	0.297	0.687	0.592	0.635	0.332	
			\bar{J}	0.700	0.986	0.942	0.265	0.940	0.915	0.928		
			\bar{p}_{acc}	0.825	0.993	0.970	0.531	0.969	0.955	0.963		
	$I^{(3)}$	2	\bar{F}_{score}	0.177	0.911	0.904	0.179	0.904	0.178	0.927	0.426	
			\bar{J}	0.400	0.941	0.939	0.400	0.938	0.287	0.942		
			\bar{p}_{acc}	0.800	0.980	0.979	0.800	0.979	0.573	0.980		
	$I^{(4)}$	2	\bar{F}_{score}	0.194	0.198	0.395	0.196	0.409	0.395	0.348	0.195	
			\bar{J}	0.284	0.288	0.618	0.284	0.644	0.619	0.555		
			\bar{p}_{acc}	0.568	0.570	0.787	0.568	0.801	0.787	0.748		
	$I^{(5)}$	3	\bar{F}_{score}	0.758	0.755	0.838	0.860	0.842	0.839	0.838	0.509	
			\bar{J}	0.683	0.652	0.867	0.867	0.877	0.862	0.863		
			\bar{p}_{acc}	0.932	0.924	0.971	0.970	0.973	0.970	0.971		
	$I^{(6)}$	3	\bar{F}_{score}	0.162	0.159	0.359	0.368	0.361	0.369	0.388	0.477	
			\bar{J}	0.250	0.250	0.635	0.558	0.637	0.633	0.621		
			\bar{p}_{acc}	0.750	0.750	0.769	0.856	0.772	0.764	0.738		
	$I^{(7)}$	4	\bar{F}_{score}	0.401	0.207	0.372	0.140	0.381	0.374	0.373	0.477	
			\bar{J}	0.572	0.314	0.576	0.125	0.580	0.589	0.599		
			\bar{p}_{acc}	0.737	0.631	0.748	0.498	0.755	0.775	0.800		
	$I^{(8)}$	2	\bar{F}_{score}	0.310	0.310	0.562	0.318	0.573	0.422	0.500	0.230	
			\bar{J}	0.420	0.420	0.747	0.420	0.753	0.642	0.710		
			\bar{p}_{acc}	0.840	0.840	0.912	0.840	0.915	0.845	0.890		
	$I^{(9)}$	2	\bar{F}_{score}	0.170	0.843	0.755	0.170	0.755	0.766	0.732	0.470	
			\bar{J}	0.281	0.967	0.952	0.281	0.952	0.954	0.949		
			\bar{p}_{acc}	0.562	0.984	0.976	0.562	0.976	0.977	0.974		
	$I^{(10)}$	4	\bar{F}_{score}	0.309	0.381	0.401	0.374	0.386	0.402	0.403	0.260	
			\bar{J}	0.392	0.499	0.431	0.414	0.442	0.432	0.429		
			\bar{p}_{acc}	0.645	0.755	0.545	0.763	0.593	0.545	0.538		
Average Results			\bar{F}_{score}	0.317	0.562	0.619	0.320	0.621	0.524	0.601		
			\bar{J}	0.440	0.618	0.757	0.404	0.763	0.680	0.745		
			\bar{p}_{acc}	0.750	0.839	0.862	0.723	0.869	0.815	0.856		

Table B.36: Detailed performance results for the original images

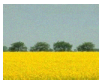
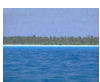
Image	Index	K	Performance Measure	FastEFM	LSC	RLPFM	LE	LPI	RLPI	RRLPI	ξ
	$\tilde{I}^{(1)}$	2	\bar{F}_{score}	0.290	0.300	0.904	0.294	0.908	0.905	0.900	0.178
\bar{J}			0.421	0.422	0.865	0.421	0.866	0.866	0.864		
\bar{p}_{acc}			0.841	0.841	0.960	0.841	0.960	0.960	0.959		
	$\tilde{I}^{(2)}$	2	\bar{F}_{score}	0.295	0.721	0.497	0.297	0.492	0.305	0.393	0.332
\bar{J}			0.735	0.958	0.907	0.265	0.905	0.801	0.868		
\bar{p}_{acc}			0.849	0.979	0.951	0.531	0.950	0.890	0.929		
	$\tilde{I}^{(3)}$	2	\bar{F}_{score}	0.179	0.181	0.885	0.179	0.887	0.172	0.899	0.426
\bar{J}			0.400	0.400	0.934	0.400	0.935	0.392	0.939		
\bar{p}_{acc}			0.800	0.800	0.977	0.800	0.978	0.572	0.979		
	$\tilde{I}^{(4)}$	2	\bar{F}_{score}	0.195	0.196	0.332	0.196	0.338	0.327	0.294	0.195
\bar{J}			0.284	0.284	0.612	0.284	0.627	0.604	0.582		
\bar{p}_{acc}			0.568	0.568	0.781	0.568	0.788	0.776	0.762		
	$\tilde{I}^{(5)}$	3	\bar{F}_{score}	0.754	0.849	0.828	0.855	0.833	0.824	0.828	0.509
\bar{J}			0.678	0.885	0.869	0.869	0.877	0.862	0.866		
\bar{p}_{acc}			0.930	0.974	0.972	0.970	0.974	0.970	0.971		
	$\tilde{I}^{(6)}$	3	\bar{F}_{score}	0.256	0.406	0.289	0.253	0.299	0.289	0.303	0.477
\bar{J}			0.484	0.560	0.586	0.509	0.592	0.587	0.590		
\bar{p}_{acc}			0.586	0.858	0.730	0.831	0.730	0.733	0.725		
	$\tilde{I}^{(7)}$	4	\bar{F}_{score}	0.376	0.364	0.362	0.135	0.370	0.372	0.374	0.477
\bar{J}			0.576	0.396	0.572	0.124	0.579	0.581	0.581		
\bar{p}_{acc}			0.749	0.737	0.743	0.497	0.757	0.764	0.763		
	$\tilde{I}^{(8)}$	2	\bar{F}_{score}	0.316	0.310	0.385	0.320	0.394	0.346	0.359	0.230
\bar{J}			0.420	0.420	0.635	0.421	0.640	0.597	0.605		
\bar{p}_{acc}			0.840	0.840	0.838	0.840	0.842	0.808	0.815		
	$\tilde{I}^{(9)}$	2	\bar{F}_{score}	0.170	0.170	0.699	0.170	0.710	0.708	0.678	0.470
\bar{J}			0.281	0.281	0.947	0.281	0.949	0.949	0.945		
\bar{p}_{acc}			0.562	0.562	0.973	0.562	0.974	0.974	0.972		
	$\tilde{I}^{(10)}$	4	\bar{F}_{score}	0.180	0.372	0.383	0.375	0.373	0.380	0.383	0.260
\bar{J}			0.203	0.512	0.430	0.416	0.433	0.432	0.426		
\bar{p}_{acc}			0.670	0.624	0.546	0.765	0.575	0.560	0.536		
Average Results			\bar{F}_{score}	0.301	0.387	0.556	0.307	0.560	0.463	0.541	
			\bar{J}	0.448	0.512	0.736	0.399	0.740	0.667	0.727	
			\bar{p}_{acc}	0.740	0.778	0.847	0.721	0.853	0.801	0.841	

Table B.37: Detailed performance results for the corrupted images. ($\sigma^{(\xi)} = 10^{-3}$)

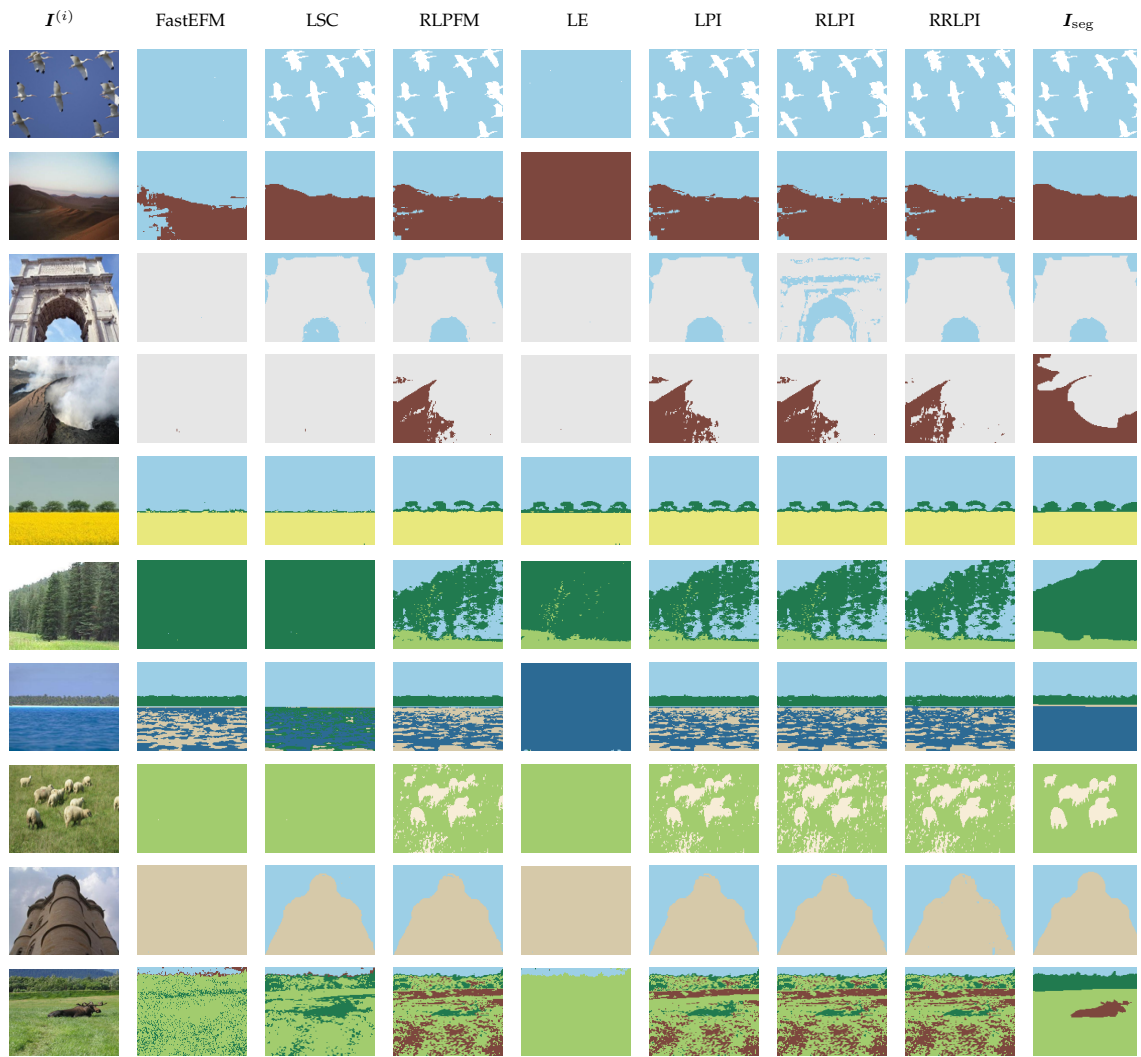


Figure B.15: Image segmentation results for the original images

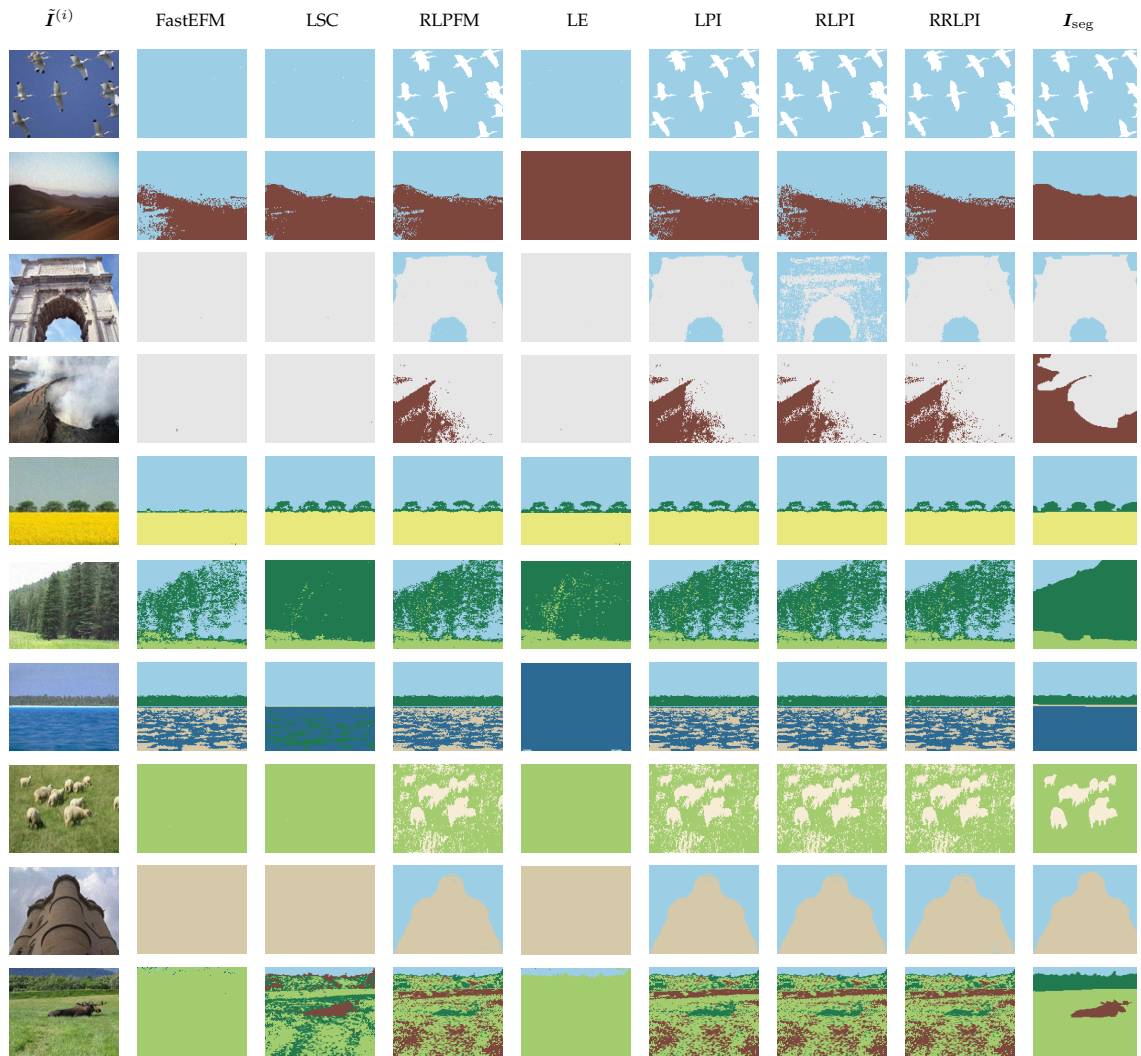


Figure B.16: Image segmentation results for the corrupted images

List of Acronyms

ARE	asymptotic relative efficiency
BC	Bayesian clustering approach
BCE	Bayesian cluster enumeration
BD	block diagonal
BDR	block diagonal representation
BDR-B	BDR using matrix \mathbf{B}
BDSSC	subspace segmentation with BD prior
BNMF	Bayesian nonnegative matrix factorization
Breast Cancer	Breast Cancer Wisconsin data set
COIL20	Columbia object image library
Con-DBSCAN	computationally efficient DBSCAN
CW	walking with a cane
CW_{oos}	walking with a cane out of synchronisation
DBSCAN	density-based spatial clustering
EBDR	eigenvalue-based BDR
EM	expectation maximisation
EnSC	elastic net subspace clustering
FastEFM	fast large-scale spectral clustering via effective feature mapping
Fisheriris	Fisher's iris data set
Gait	radar-based human gait data set
IBDLR	implicit BD low-rank representation

JAFFE	Japanese female facial expression data set
L ₁	limping with one leg
L ₂	limping with both legs
LE	Laplacian eigenmaps
LFR	Lancichinetti-Fortunato-Radicchi
LPI	locality preserving indexing
MAP	Infomap community detection method
MNIST	Modified National Institute of Standards and Technology
MST	minimal spanning tree
NE	Newman's eigenvector method
NGA	Newman's greedy algorithm
NW	normal walking
O. Cancer	ovarian cancer data set
ORL	data base of faces
Parkinson A.	Parkinson's disease's acoustic features
PCA	principal component analysis
Person Id.	person identification
RKLRR	robust kernel low-rank representation
RLPFM	robust locality preserving feature mapping
RLPI	regularized locality preserving indexing
RRLPI	robust regularized locality preserving indexing
SABDR	sparsity-aware BDR
SBM	stochastic block model
SC	spectral clustering
Sonar	connectionist bench data set
SPARCODE	sparsity-aware robust community detection
SSC	sparse subspace clustering
SVD	singular value decomposition
USPS	data set of handwritten text digits

List of Notations & Symbols

The following list comprises the most important notations, operators and symbols. The remaining ones have been determined upon usage.

NOTATIONS

n	lowercase letter denotes a scalar
N	uppercase letter denotes a scalar
\mathbf{x}	bold lowercase letter denotes a column vector
\mathbf{X}	bold uppercase letter denotes a matrix
$\tilde{\mathbf{X}}$	a corrupted matrix
\mathbb{R}	set of real numbers
\mathbb{R}_+	set of positive real numbers
\mathbb{Z}_+	set of positive integers
$\mathbb{R}^{n \times 1}$	set of column vectors of size n on \mathbb{R}
$\mathbb{R}^{n \times n}$	set of matrices of size $n \times n$ on \mathbb{R}
$\mathbb{1}$	the indicator function
$f(\cdot)$	probability density function
$\arg \max_x F(x)$	returns the x value that maximizes $F(x)$
$\arg \min_x F(x)$	returns the x value that minimizes $F(x)$

OPERATORS

$(\cdot)^{-1}$	matrix inverse
$(\cdot)^\dagger$	Moore-Penrose inverse
$(\cdot)^\top$	vector of matrix transpose
$ \cdot $	determinant of a matrix or absolute value of a scalar
$\det(\cdot)$	determinant of a matrix
$\text{med}(\cdot)$	median operator
\log	natural logarithm
$\ \cdot\ _1$	ℓ_1 norm
$\ \cdot\ _2$	ℓ_2 norm

SYMBOLS

X	a data matrix
W	the affinity matrix
A	the adjacency matrix
D	the diagonal matrix of overall edge weights
L	the unnormalized Laplacian matrix
<i>G</i>	graph
<i>V</i>	set of vertices
<i>E</i>	set of edges
<i>i</i>	index operator for the blocks/clusters
<i>j</i>	index operator for the blocks/clusters
<i>k</i>	index operator for the blocks/clusters
<i>m</i>	index operator for the samples
<i>n</i>	index operator for the samples
<i>r</i>	index operator for the samples
$w_{m,n}$	m, n th similarity coefficient
$w_{i,j}$	a constant around which the similarity coefficients between block i and j are concentrated

\mathbf{W}_i	i th block of the BD affinity matrix
λ_m	m th eigenvalue of \mathbf{L}
\mathbf{y}_m	eigenvector associated with m th eigenvalue of \mathbf{L}
\mathbf{y}_F	the Fiedler vector
$y_{n,m}$	n th embedding result in \mathbf{y}_m
$y_{n,F}$	n th embedding result in the Fiedler vector
β_m	transformation vector associated with \mathbf{y}_m
β_F	transformation vector associated with \mathbf{y}_F
c_m	label of the m th feature vector
M	dimension of feature vectors
N	number of feature vectors
N_i	N associated with the i th block
N_I	number of Type I outliers
N_{II}	number of Type II outliers
N_{\min}	minimum number of samples in clusters or blocks
N_c	number of changepoints
$N_{c_{\max}}$	determined maximum number of changepoints
N_K	candidate number of clusters or blocks
N_E	number of experiments
K	number of clusters or blocks
K_{cand}	candidate number of clusters or blocks
K_{\min}	minimum number of clusters or blocks
K_{\max}	maximum number of clusters or blocks
mod	modularity
cond	conductance
\mathbf{n}	column vector of block sizes
\mathbf{n}_r	r th column vector of candidate block sizes
\mathbf{w}	column vector of similarity coefficients that the blocks concentrated around
\mathbf{e}_i	embedding vector associated with i th feature vector

\mathbf{v}	column vector that represents \mathbf{L} as a piece-wise linear function
$\boldsymbol{\tau}$	column vector containing changepoint locations
\hat{p}_{det}	the empirical probability of detection
\bar{p}_{acc}	average clustering accuracy
t	computation time

List of Figures

1.1	Exemplary graph clustering. Left: graph representation of digit samples from the MNIST data base [HS98]. The red edges represent connections to outliers which are ones that have connections to more than one group of digit samples. The green, blue and yellow lines represent the within-cluster edges of digits 9, 4 and 3, respectively. Right: cluster assignment based on the general graph clustering idea that maximizes the number of intra-cluster edges while minimizing the number of inter-cluster edges.	2
1.2	Example graph constructions for handwritten digit samples from MNIST data base [HS98].	3
2.1	Exemplary graph construction process.	8
2.2	Exemplary Laplacian, overall edge weight and affinity matrices.	10
2.3	Exemplary balanced graph partitioning.	12
2.4	Exemplary BD affinity matrix and associated graph.	17
3.1	Exemplary illustration of Definition 3.1.1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	19
3.2	Exemplary illustration of the sparse affinity matrix \mathbf{W} and associated matrices \mathbf{D} and $\mathbf{L} \in \mathbb{R}^{N \times N}$ ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	20
3.3	Exemplary illustration of Theorem 1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	21
3.4	Spectral embedding according to the eigenvectors of the Laplacian matrix \mathbf{L} when $K = 3$	22
3.5	Exemplary sparse Affinity matrix, sparse Laplacian matrix and corresponding vector \mathbf{v} ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	24
3.6	Numerical results for the parameter $N_{c_{\max}}$	30

3.7	Robustness in spectral analysis.	33
3.8	Exemplary illustration of ε definition.	35
4.1	Illustration of Type I outliers. The colored cells in the corrupted BD affinity matrix $\tilde{\mathbf{W}}$ represent non-zero edge weights in graph \tilde{G}	42
4.2	Illustration of Type II outliers. The red colored cells in $\tilde{\mathbf{W}}$ correspond to edges of Type II outliers.	43
4.3	Illustration of outliers' effect on the affinity matrix.	44
4.4	Exemplary outlyigness measure of Person Id. [TSM18] data set based on the overall edge weights.	45
4.5	Exemplary outlyigness measure of Gait [SAZ19] data set based on the overall edge weights.	46
4.6	Exemplary plot of Type I outliers' effect on eigenvalues ($\mathbf{n} = [10, 8, 12, 1]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3$).	47
4.7	Exemplary illustration of Theorem 4 ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3$).	48
4.8	Exemplary illustration of Theorem 5 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3, i = 1$).	49
4.9	Exemplary illustration of Theorem 5 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3, i = K$).	49
4.10	Spectral embedding according to the eigenvectors of the corrupted Laplacian matrix $\tilde{\mathbf{L}}$ when $K = 3, i = 1$ and $j = 2$	51
4.11	Exemplary plot of Type I outliers' effect on vector \mathbf{v} ($\mathbf{n} = [10, 8, 12, 1]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3$).	52
4.12	Exemplary illustration of Theorem 6 ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3, m_{\text{II}} = 1$).	53
4.13	Exemplary illustration of Theorem 6 ($\mathbf{n} = [10, 1, 8, 12]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3, m_{\text{II}} = \ell_2 - 1$).	54
4.14	Exemplary illustration of Theorem 7 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3, i = 1$).	54
4.15	Exemplary illustration of Theorem 7 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3, i = K$).	55
4.16	Exemplary illustration of Corollary 7.1 ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	56
4.17	Example graphs for increasing sparsity ($\mathbf{W} = \mathbf{X}^\top \mathbf{X}$).	57

4.18	The main steps of SPARCODE.	61
4.19	The empirical distribution of similarity coefficients for growing penalty parameter values.	64
4.20	The thresholding operation of SPARCODE.	65
4.21	Graphical models of Scenario 1.	71
4.22	Graphical models of Scenario 1.	72
4.23	Computation time for a growing number of vertices for Scenario 2. The results are reported in seconds. The upper figure zooms into the region concerning networks up to a size of 1000 vertices.	73
4.24	Performance of different partitioning methods in terms of computational complexity. The results are reported in seconds.	74
4.25	Modularity and conductance for a growing values of ζ for LFR networks.	75
4.26	Scatter plot for three important features of Gait data belonging to five object communities.	78
4.27	Graphical models of Gait data set.	78
4.28	Scatter plot for the first three principal components of Person Id. data over four object communities.	79
4.29	Graphical models of Person Id. data set.	79
4.30	Empirical probability of detection and average normalized performance rank of each algorithm in terms of modularity, conductance and computation time.	82
4.31	Performance of different algorithms based on equal weights on performance metrics.	83
4.32	Performance of different algorithms based on rank in terms of probability of detection.	84
4.33	Exemplary graph partitioning digit samples from MNIST data base [HS98].	86
4.34	High-level flow diagram illustrating the key steps of FRS-BDR using a generic example with $K = 3$ clusters.	88
4.35	Exemplary plot of the sBDO algorithm.	89
4.36	Exemplary plot of candidate block sizes.	92
4.37	Exemplary illustration of $\ddot{\mathbf{v}}$ and \mathbf{W}_{sim} with $K_{\text{cand}} = K$, $\mathbf{n} = [10, 8, 12]^{\top} \in \mathbb{R}^K$, $\text{diag}(\mathbf{W}_{\text{sim}}) = [0.6, 0.3, 0.9]^{\top} \in \mathbb{R}^K$, $\tilde{w}_{1,2} = 0.2$, $\tilde{w}_{1,3} = 0.4$, and $\tilde{w}_{2,3} = 0.1$	94

4.38	Numerical results for MNIST data set. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.	97
4.39	Numerical results for COIL20 data set. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.	98
4.40	Numerical results for ORL and JAFFE data sets. The regularization parameters of the competing methods are tuned for optimal performance in all settings while the proposed method determines the parameters using Algorithms 5 and 6. In the regularization parameter performance analysis, for all competing methods that use two parameters, the second one is tuned optimally while varying the first.	99
4.41	Exemplary image segmentation result comparing the popular LE [BN01] and the proposed RRLPI methods for Fiedler vector estimation.	103
4.42	Fiedler vector computation for an ideal $K = 2$ blocks affinity matrix.	107
4.43	Fiedler vector computation for a corrupted $K = 2$ blocks affinity matrix. The corruptions of the affinity matrix by Type I and Type II outliers are highlighted by coloring the corresponding affected elements in light red and dark red, respectively. In the Fiedler vector, outliers are positioned as shown in the right illustration.	108
4.44	Exemplary plot of the Fiedler vector computation based on LE and RRLPI methods. The weighting operation in RRLPI on Type I and Type II outliers results in two clusters of concentrated mappings that include the outliers. In this way, the true structure of the non-outlying data becomes visible, even in the presence of outliers. By contrast, for LE, the outliers deteriorate the underlying two-cluster structure. Further, the weights provide a robust measure of outlyingness, which may be used to detect and analyze outliers, which is of high interest in some applications.	111
4.45	Example of Δ -separated sets \mathbf{s} and \mathbf{t}	113
4.46	Exemplary plot of the first three features of the uncorrupted synthetic data set.	119

4.47	Computed eigenvectors for the uncorrupted data set.	120
4.48	Exemplary plot of the first three features of the synthetic data set after corruption with Type I and Type II outliers (red crosses). See Section 4.4.2.1.3, for a discussion.	120
4.49	Computed eigenvectors for the corrupted data set.	121
4.50	Average partition accuracy as a function of \mathcal{D}_{OI} and N_{out} for each outlier type	121
4.51	Computation time performance analysis. The results are reported in seconds.	122
4.52	Numerical results for cluster enumeration based on RRLPI.	123
4.53	Image segmentation results for the original images.	124
4.54	Example segmentations for LE and RRLPI methods. The embeddings that are mapped far away from the group of pixels are pointed out using arrows.	125
4.55	Image segmentation results for the corrupted images. ($\sigma^{(\xi)} = 10^{-3}$).	125
4.56	Numerical results for the image segmentation.	126
4.57	The eigenvectors associated with $K = 3$ smallest eigenvalues for a data matrix.	129
4.58	Examples of estimated feature spaces for corrupted and non-corrupted versions of the Fisheriris data set.	133
4.59	\bar{p}_{acc} for increasing \mathcal{D}_c and N_{out} values.	134
4.60	Scatter plot for three important features of radar-based human gait data belonging to five object communities.	136
4.61	Normalized histogram of degrees for graph of radar-based gait signatures.	137
4.62	Example of (a), (c) spectrograms and (b), (d) corresponding micro-Doppler envelopes for two object clusters.	142
A.1	Exemplary illustration of Theorem 1.S. ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3$).	155
A.2	Exemplary illustration of Theorem 4.S. ($\mathbf{n} = [1, 10, 8, 12]^\top \in \mathbb{R}^{K+1}, N+1 = 31, K = 3$).	175
A.3	Exemplary illustration of Theorem 5.S. ($\mathbf{n} = [10, 8, 12]^\top \in \mathbb{R}^K, N = 30, K = 3, i = 1$).	180
B.1	Visual summary of FRS-BDR	198
B.2	λ_1 for increasing values of α associated with generalized eigen-decomposition	200
B.3	λ_1 for increasing values of $\tilde{w}_{1,2}$ associated with standard eigen-decomposition	201
B.4	Numerical results for USPS data set.	211
B.5	Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.	211

B.6	Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.	212
B.7	Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.	214
B.8	Average clustering accuracy (\bar{p}_{acc}) of FRS-BDR for increasing number of PCA features.	215
B.9	Numerical results for Yale data set.	216
B.10	Estimated eigenvectors for the uncorrupted and corrupted data sets.	220
B.11	\bar{p}_{acc} performance of different partitioning methods for increasing ϑ_1 associated with Type I outlier ($N = 300, N_{\text{out}} = 10, \vartheta_2 = 1.5, \vartheta_{c_K} = 0.5$ s.t. $K = 1, \dots, K$.)	221
B.12	\bar{p}_{acc} performance of different partitioning methods for each of outlier type with increasing N_{out} ($N = 300, \vartheta_1 = 5, \vartheta_2 = 1.5, \vartheta_{c_K} = 0.5$ s.t. $K = 1, \dots, K$.)	221
B.13	Numerical results for cluster enumeration using different partitioning algorithms.	222
B.14	Overall performance rank for different partitioning algorithms.	223
B.15	Image segmentation results for the original images	229
B.16	Image segmentation results for the corrupted images	230

List of Tables

3.1	Numerical results for real-world datasets ($N_{c_{\max}} = 8$).	30
3.2	$\bar{p}_{\text{acc}}(\%)$ for real-world datasets. ‘x’ denotes the results that produce complex-valued eigenvectors, $N_{c_{\max}} = 8$. The numbers indicate the best \bar{p}_{acc} for the competitors.	31
3.3	t (seconds) for real-world datasets. Except for EBDR the level of sparsity assumed to be known and it is defined as \hat{p} for EBDR(\hat{p}). $N_{c_{\max}} = 8$ in all cases.	31
3.4	Face clustering performance of different BDR methods on ORL data set. ‘x’ denotes the failed results.	38
3.5	Face clustering performance of different BDR methods on JAFFE data set. ‘x’ denotes the failed results.	39
3.6	Object clustering performances of different BDR methods on COIL20 data set.	39
3.7	Handwritten-digit clustering performances of different BDR methods on USPS data set.	40
4.1	Performance of 10 graph-based approaches on Scenario 1 where $K = 7$	71
4.2	Performance of 10 graph-based approaches on Scenario 2 where $K = 3$	73
4.3	Performance of graph-based approaches on LFR data sets for $\beta_w = 1$ and $K = 3$	75
4.4	Performance of graph-based approaches on well-known networks. The results whose computation takes more than 12 hours and nontarget networks for SVD method are denoted as ”-”.	77
4.5	Performance of graph-based approaches on clustering data sets. The results whose computation takes more than 12 hours and nontarget networks for SVD method are denoted as ”-”.	81

4.6	Performance of cluster-based approaches on clustering data sets. The results whose computation take more than 12 hours and nontarget networks for SVD method denoted as "x".	83
4.7	Subspace clustering performance of different BDR approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. 'x' denotes the failed results due to the complex-valued eigenvectors. . . .	100
4.8	Performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\text{pen} = 0.5$	123
4.9	K -means partitioning performance for real-world datasets. The average probability of detection shown in %.	132
4.10	Cluster enumeration results of different parameter-free clustering algorithms for five object clusters.	143
4.11	Confusion matrices of different gait clustering algorithms for five object clusters. Numbers are shown in % and best performance results are indicated in bold font.	144
B.1	Similarity coefficients for seven object communities for Scenario 1. The density parameters of similarity coefficients associated with $b_{c_i, c_j} = 0$ are denoted as 'x'. . . .	189
B.2	Similarity coefficients for three object communities and outliers for Scenario 2. The density parameters of similarity coefficients associated with $b_{c_i, c_j} = 0$ denoted as 'x'.	190
B.3	Subspace clustering performance of different block diagonal representation approaches on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	209
B.4	Computation time performance of FRS-BDR method on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	209
B.5	Computation time performance of different block diagonal representation approaches on MNIST data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	209
B.6	Subspace clustering performance of different block diagonal representation approaches on USPS data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	210

B.7	Computation time performance of FRS-BDR method on USPS data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	210
B.8	Computation time performance of different block diagonal representation approaches on USPS data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	210
B.9	Subspace clustering performance of different block diagonal representation approaches on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	211
B.10	Computation time performance of FRS-BDR method on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	212
B.11	Computation time performance of different block diagonal representation approaches on COIL20 data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	212
B.12	Subspace clustering performance of different block diagonal representation approaches on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. ‘x’ denotes the failed results.	213
B.13	Computation time performance of FRS-BDR method on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	213
B.14	Computation time performance of different block diagonal representation approaches on ORL data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	213
B.15	Subspace clustering performance of different block diagonal representation approaches on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. ‘x’ denotes the failed results.	214
B.16	Computation time performance of FRS-BDR method on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	214

B.17	Computation time performance of different block diagonal representation approaches on JAFFE data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	215
B.18	Subspace clustering performance of different block diagonal representation approaches on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. ‘x’ denotes the failed results.	215
B.19	Computation time performance of FRS-BDR method on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	216
B.20	Computation time performance of different block diagonal representation approaches on Yale data set. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	216
B.21	Estimation performance of FRS-BDR on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	217
B.22	Subspace clustering performance of different block diagonal representation approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$. ‘x’ denotes the failed results due to the complex-valued eigenvectors.	217
B.23	Computation time performance of different block diagonal representation approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and sparsity assumed to be known for all sparse representation methods which means that computation time of FRS-BDR is detailed for Steps 1.1, 1.2 and 2.	217
B.24	Performance of different cluster enumeration approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	218
B.25	Partitioning performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for mod using the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	218

B.26	Partitioning performance of different cluster enumeration approaches on well-known clustering data sets. The results summarized for conductance (cond) using the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$	218
B.27	Computation performance of different cluster enumeration approaches on well-known clustering data sets. The results are summarized for the similarity measure $\mathbf{W} = \mathbf{X}^\top \mathbf{X}$ and FRS-BDR is detailed for all steps.	219
B.28	Detailed numerical information for the synthetic data set	219
B.29	Performance for different c values on well-known clustering data sets. The results are summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\rho = 0.5$. The partitioning is determined by K -means.	223
B.30	Performance for different c values on well-known clustering data sets. The results are summarized for similarity measures cosine (cos) and elastic net (enet) using a penalty parameter of $\rho = 0.5$. The partitioning is determined by K -medoids with Tukey's distance function for the initialization ($c_{\text{Tukey}} = 3$).	223
B.31	Performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -medoids with Tukey's distance function [ZKO18] for the initialization ($c_{\text{Tukey}} = 3$).	224
B.32	Modularity performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -medoids with Tukey's distance function [ZKO18] for the initialization ($c_{\text{Tukey}} = 3$).	224
B.33	Performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -means.	224
B.34	Modularity performance of different cluster enumeration approaches on well-known clustering data sets. The partitioning is determined by K -means.	225
B.35	Reference annotations for the ground truth images.	226
B.36	Detailed performance results for the original images	227
B.37	Detailed performance results for the corrupted images. ($\sigma^{(\xi)} = 10^{-3}$)	228

References

- [Abb17] E. Abbe, “Community detection and stochastic block models: recent developments,” *J. Mach. Learn. Res.*, vol. 18, pp. 6446-6531, 2017.
- [ABG00] D. Ayres-de-Campos, J. Bernardes, A. Garrido, J. Marques-de-Sa and L. Pereira-Leite, “SisPorto 2.0: a program for automated analysis of cardiocograms,” *J. Maternal-Fetal Med.*, vol. 9, pp. 311-318, 2000.
- [ABK99] M. Ankerst, M. M. Breunig, H. -P. Kriegel and J. Sander, “OPTICS: Ordering points to identify the clustering structure,” *ACM Sigmod Rec.*, vol. 28, p. 49-60, 1999.
- [AG05] L. A. Adamic and N. Glance, “The political blogosphere and the 2004 US election: divided they blog,” in *Proc. 3rd Intl. Workshop Link Discovery Res.*, pp. 36-43, 2005.
- [AGR19] M. Abdolali, N. Gillis and M. Rahmati, “Scalable and robust sparse subspace clustering using randomized clustering and multilayer graphs,” *Signal Process.*, vol. 163, pp. 166-180, 2019.
- [AH14] B. Antal and A. Hajdu, “An ensemble-based system for automatic screening of diabetic retinopathy,” *Knowledge Based Syst.*, vol. 60, pp. 20-27, 2014.
- [AHK10] S. Arora, E. Hazan and S. Kale, “ $O\sqrt{\log n}$ approximation to sparsest cut in $\tilde{O}(n^2)$ time,” *SIAM J. Comput.*, vol. 39, pp. 1748-1771, 2010.
- [ARV08] S. Arora, S. Rao and U. Vazirani, “Geometry, flows, and graph-partitioning algorithms,” *Commun. ACM*, vol. 51, pp. 96-105, 2008.
- [ARV09] S. Arora, S. Rao and U. Varizani, “Expander flows, geometric embeddings and graph partitioning” *J. ACM*, vol. 56, pp. 1-37, 2009.

- [ASD12] R. Aragues, G. Shi, D. V. Dimarogonas, C. Sagues and K.H. Johansson, “Distributed algebraic connectivity estimation for adaptive event-triggered consensus” in *Proc. Am. Control Conf.*, pp. 32-37, 2012.
- [ASI20] M. Ahmed, R. Seraj and S. M. S. Islam, “The k-means algorithm: A comprehensive survey and performance evaluation,” *Electron.*, vol. 9, p. 1295, 2020.
- [AWF92] C. J. Anderson, S. Wasserman and K. Faust, “Building stochastic blockmodels,” *Social Networks*, vol. 14, pp. 137-161, 1992.
- [AZA16] M. G. Amin, Y. D. Zhang, F. Ahmad and K. C. Ho, “Radar signal processing for elderly fall detection: The future for in-home monitoring,” *IEEE Signal Process. Mag.*, vol. 33, pp. 71-80, 2016.
- [BEL14] L. Bohlin, D. Edler, A. Lancichinetti, and M. Rosvall, “Community detection and visualization of networks with the map equation framework,” *Measuring Scholarly Impact*, pp. 3-34, 2014.
- [BGL08] V. D. Blondel, J. L. Guillaume, R. Lambiotte and E. Lefebvre, “Fast unfolding of communities in large networks,” *J. Stat. Mech: Theory Exp.*, vol. 10, pp. P10008, 2008.
- [BHK97] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, “Eigenfaces vs. fisherfaces: Recognition using class specific linear projection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, pp. 711-720, 1997.
- [BM13] A. Bertrand and M. Moonen, “Seeing the bigger picture: How nodes can learn their place within a complex ad hoc network topology” *IEEE Signal Process. Mag.*, vol. 30, pp. 71-82, 2013.
- [BM21] P. Bhattacharjee and P. Mitra, “A survey of density based clustering algorithms,” *Front. Comput. Sci.*, vol. 15, pp. 1-27, 2021.
- [BMS16] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders and C. Schulz, “Recent advances in graph partitioning,” *Algorithm Eng.*, pp. 117-158, 2016.
- [BN01] M. Belkin and P. Niyogi, “Laplacian eigenmaps and spectral techniques for embedding and clustering,” in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, vol. 14, 2001.

- [BPB21] I. -D. Borlea, R. -E. Precup, A. -B. Borlea and D. Iercan, “A unified form of fuzzy C-means and K-means algorithms and its partitional implementation,” *Knowledge-Based Syst.*, vol. 214, p. 106731, 2021.
- [BS03] S. D. Bay and M. Schwabacher, “Mining distance-based outliers in near linear time with randomization and a simple pruning rule,” in *9th ACM SIGKDD Intl. Conf. Knowl. Discovery and Data Mining*, ACM, pp. 29-38, 2003.
- [BYL15] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou and A. Song, “Efficient agglomerative hierarchical clustering,” *Expert Syst. Appl.*, vol. 42, pp. 2785-2797, 2015.
- [BYS17] X. Bai, P. Yang and X. Shi, “An overlapping community detection algorithm based on density peaks,” *Neurocomput.*, vol. 226, pp. 7-15, 2017.
- [CC14] D. Cai and X. Chen, “Large scale spectral clustering with landmark-based sparse representation,” *IEEE Trans. Cybern.*, vol. 45, pp. 1669-1680, 2014.
- [CC15] A. Cloninger and W. Czaja, “Eigenvector localization on data-dependent graphs,” in *Int. Conf. Sampling Theory Appl.*, pp. 608-612, 2015.
- [CFR04] T. P. Conrads, V. A. Fusaro, S. Ross, D. Johann, V. Rajapakse, B. A. Hitt, S. M. Steinberg, E. C. Kohn, D. A. Fishman, G. Whitely, J. C. Barrett, L. A. Liotta, E. F. Petricoin and T. D. Veenstra, “High-resolution serum proteomic features for ovarian cancer detection,” *Endocrine-related Cancer*, vol. 11, pp. 163-178, 2004.
- [CHC06] B. L. Chen, D. H. Hall and D. B. Chklovskii, “Wiring optimization can relate neuronal structure and function,” in *Proc. Natl. Acad. Sci.*, vol. 103, pp. 4723-4728, 2006.
- [CHH05] D. Cai, X. He and J. Han, “Document clustering using locality preserving indexing” *IEEE Trans. Knowl. Data Eng.*, vol. 17, pp. 1624-1637, 2005.
- [CHH10] D. Cai, X. He, J. Han and T. S. Huang, “Graph regularized nonnegative matrix factorization for data representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, pp. 1548-1560, 2010.
- [Chu97] F. R. K. Chung, *Spectral graph theory*, American Mathematical Soc., 1997.
- [CHZ07] D. Cai, X. He, W. V. Zhang and J. Han, “Regularized locality preserving indexing via spectral regression,” in *Proc. 16th ACM Conf. Inf. Knowl. Manage.*, pp. 741-750, 2007.

- [CJS14] Y. Chen, A. Jalali, S. Sanghavi and H. Xu, “Clustering partially observed graphs via convex optimization,” *J. Mach. Learn. Res.*, vol. 15, pp. 2213-2238, 2014.
- [CK10] W. Cheney and D. Kincaid, *Linear algebra: Theory and applications*, Jones&Bartlett, 2010.
- [CLP13] G. Csurka, D. Larlus, F. Perronnin and F. Meylan, “What is a good evaluation measure for semantic segmentation?,” *BMVC*, vol. 27, pp. 1-32, 2013.
- [CLY13] H. Cheng, Z. Liu, L. Yang and X. Chen, “Sparse representation and learning in visual recognition: Theory and applications,” *Signal Process.*, vol. 93, pp. 1408-1425, 2013.
- [CM69] E. Cuthill, and J. McKee, “Reducing the bandwidth of sparse symmetric matrices,” in *Proc. 24th Nat. Conf.*, 1969.
- [CNM04] A. Clauset, M. E. J. Newman and C. Moore, “Finding community structure in very large networks,” *Phys. Rev. E*, vol. 70, pp. 066111, 2004.
- [CNW15] X. Chang, F. Nie, S. Wang, Y. Yang, X. Zhou and C. Zhang, “Compound rank- k projections for bilinear analysis,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, pp. 1502-1513, 2015.
- [Cou08] P. Courrieu, “Fast computation of Moore-Penrose inverse matrices.” Online-Edition: <https://arxiv.org/abs/0804.4809>, 2008.
- [CYY09] B. Cheng, J. Yang, S. Yan, Y. Fu and T. S. Huang, “Learning with ℓ^1 -graph for image analysis,” *IEEE Trans. Image Process.*, vol. 19, pp. 858-866, 2009.
- [DBE16] Y. Dar, A. M. Bruckstein, M. Elad and R. Giryes, “Postprocessing of compressed images via sequential denoising,” *IEEE Trans. Image Process.*, vol. 25, pp. 3044-3058, 2016.
- [DDF90] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer and R. Harshman, “Indexing by latent semantic analysis” *J. Am. Soc. Inf. Sci.*, vol. 41, pp. 391-407, 1990.
- [DH73] W. E. Donath and A. J. Hoffman, “Lower bounds for the partitioning of graphs,” *IBM J. Res. Dev.*, vol. 17, pp. 420-425, 1973.
- [DHZ01] C. H. Q. Ding, X. He, H. Zha, M. Gu and H. D. Simon, “A min-max cut algorithm for graph partitioning and data clustering,” in *Proc. IEEE Int. Conf. Data Min.*, pp. 107-114, 2001.

- [Don06] D. L. Donoho, “For most large underdetermined systems of linear equations the minimal ℓ^1 -norm solution is also the sparsest solution,” *Commun. Pure Appl. Math.*, vol. 59, pp. 797-829, 2006.
- [DPC19] Y. Ding, S. Pan and Y. Chong, “Robust spatial-spectral block diagonal structure representation with fuzzy class probability for hyperspectral image classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 58, pp. 1747-1762, 2019.
- [DS20] A. DePavia and S. Steinerberger, “Spectral Clustering Revisited: Information Hidden in the Fiedler Vector” 2020. [Online]. Available: <https://arxiv.org/abs/2003.09969>
- [DZ07] J. Ding and A. Zhou, “Eigenvalues of rank-one updated matrices with some applications,” *Appl. Math. Lett.*, vol. 20, pp. 1223-1226, 2007.
- [EKS96] M. Ester, H.-P. Kriegel, J. Sander and X. Xiaowei, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proc. 2nd Int. Conf. Knowl. Discovery Databases Data Min.*, pp. 226-231, 1996.
- [EPY97] D. Eppstein, M. S. Paterson and F. F. Yao, “On nearest-neighbor graphs,” *Discrete Comput. Geom.*, vol. 17, pp. 263-282, 1997.
- [EV12] E. Elhamifar and R. Vidal, “Block-sparse recovery via convex optimization,” *IEEE Trans. Signal Process.*, vol. 60, pp. 4094-4107, 2012.
- [EV13] E. Elhamifar and R. Vidal, “Sparse subspace clustering: Algorithm, theory, and applications,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, pp. 2765-2781, 2013.
- [FHT08] J. Friedman, T. Hastie and R. Tibshirani, “Sparse inverse covariance estimation with the graphical Lasso” *Biostat.*, vol. 9, pp. 432-441, 2008.
- [Fie73] M. Fiedler, “Algebraic connectivity of graphs,” *Czech. Math. J.*, vol. 23, pp. 298-305, 1973.
- [Fie75] M. Fiedler, “A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory,” *Czechoslovak Math. J.*, vol. 25, pp. 619-633, 1975.
- [Fie89] M. Fiedler, “Laplacian of graphs and algebraic connectivity,” *Banach Cent. Publ.*, vol. 1, pp. 57-70, 1989.

- [Fis36] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.
- [FJL20] W. Fan, R. Jin, M. Liu, P. Lu, X. Luo, R. Xu, Q. Yin, W. Yu and J. Zhou, “Application driven graph partitioning” in *Proc. 2020 ACM SIGMOD Int. Conf. Manage. Data*, pp. 1765-1779, 2020.
- [FLW21] L. Fan, G. Lu, Y. Wang and T. Liu, “Block Diagonal Sparse Subspace Clustering,” in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, pp. 1-6, 2021.
- [FLX14] J. Feng, Z. Lin, H. Xu and S. Yan, “Robust subspace segmentation with block-diagonal prior,” in *Proc. IEEE Conf. Comp. Vision Pattern Recognit.*, pp. 3818-3825, 2014.
- [FSC04] F. Fages, S. Soliman, and N. Chabrier-Rivier, “Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM,” *J. Biol. Phys. Chem.*, vol. 4, pp. 64-73, 2004.
- [Gal96] E.f. Galba, “Weighted singular decomposition and weighted pseudoinversion of matrices,” *Ukrainian Math. J.*, vol. 48, pp. 1618-1622, 1996.
- [GCC15] N. García-Pedrajas, J. A. R. Del Castillo and G. Cerruela-García, “A proposal for local k values for k -nearest neighbor rule,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, pp. 470-475, 2015.
- [GCV16] F. Grisoni, V. Consonni, M. Vighi, S. Villa and R. Todeschini, “Investigating the mechanisms of bioconcentration through QSAR classification trees,” *Environ. Int.*, vol. 88, pp. 198-205, 2016.
- [GD05] P. M. Gleiser and L. Danon, “Community structure in Jazz,” *Adv. Complex Syst.*, vol. 6, pp. 565-573, 2005.
- [GGK13] M. J. Gangeh, A. Ghodsi and M. S. Kamel, “Kernelized supervised dictionary learning,” *IEEE Trans. Signal Process.*, vol. 61, pp. 4753-4767, 2013.
- [GKR05] R. Goldenberg, R. Kimmel, E. Rivlin and M. Rudzsky, “Behavior classification by eigendecomposition of periodic motions,” *Pattern Recognit.*, vol. 38, pp. 1033-1043, 2005.
- [GN02] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” in *Proc. Natl. Acad. Sci.*, vol. 99, pp. 7821-7826, 2002.

- [GS88] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Networks*, vol. 1, pp. 75-89, 1988.
- [GZZ19] Y. Gao, H. Zhang and Y. Zhang, "Overlapping community detection based on conductance optimization in large-scale networks," *Physica A.*, vol. 522, pp. 69-79, 2019.
- [Han20] Y. Han, "Sorting Real Numbers in $O(n\sqrt{\log n})$ Time and Linear Space", *Algorithmica*, vol. 82, pp. 966-978, 2020.
- [HC04] G. Hamerly and E. Charles, "Learning the K in K-Means," in *Proc. 16th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, pp. 281-288, 2004.
- [HCL04] X. He, D. Cai, H. Liu and W. -Y. Ma, "Locality preserving indexing for document representation" in *Proc. of 27th Annu. Intl. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, pp. 96-103, 2004.
- [Hen07] B. Hendrickson, "Latent semantic analysis and Fiedler retrieval" *Linear Algebra Appl.*, vol. 421, pp. 345-355, 2007.
- [Hes04] J. P. Hespanha, "An efficient MATLAB algorithm for graph partitioning," Santa Barbara, CA, USA: University of California, 2004.
- [HG07] A. Hinneburg and H. -H. Gabriel, "Denclue 2.0: Fast clustering based on kernel density estimation," in *Proc. Int. Symp. Intell. Data Anal.*, pp. 70-80, 2007.
- [HGM14] A. Muro-de-la Herran, B. Garcia-Zapirain and A. Mendez-Zorrilla, "Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications," *Sensors*, vol. 14, no. 2, pp. 3362-3394, 2014.
- [HK17] E. Hancer and D. Karaboga, "A comprehensive survey of traditional, merge-split and evolutionary approaches proposed for determination of cluster number," *Swarm Evol. Comput.*, vol. 32, pp. 49-67, 2017.
- [HN04] X. He and P. Niyogi, "Locality preserving projections," *Adv. Neural Inf. Process. Syst.*, vol. 16, pp. 153-160, 2004.
- [HR09] P. J. Huber and E. M. Ronchetti, *Robust statistics*, John Wiley & Sons, 2009.

- [HRG18] L. He, N. Ray, Y. Guan and H. Zhang, “Fast large-scale spectral clustering via explicit feature mapping,” *IEEE Trans. Cybern.*, vol. 49, pp. 1058-1071, 2018.
- [HS98] T. Hastie and P. Y. Simard, “Metrics and models for handwritten character recognition,” *Stat. Sci.*, pp. 54-65, 1998.
- [HTF09] T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer, 2009.
- [Hul94] J. J. Hull, “A database for handwritten text recognition research,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, pp. 550-554, 1994.
- [HW08] J. M. Hofman and C. H. Wiggins, “Bayesian Approach to Network Modularity,” *Phys. Rev. Lett.*, vol. 100, pp. 258701, 2008.
- [HXZ20] E. Hancer, B. Xue and M. Zhang, “A survey on feature selection approaches for clustering,” *Artif. Intell. Rev.*, vol. 53, pp. 4519-4545, 2020.
- [HZZ16] Z. He, M. Zhang and H. Zhang, “Data-driven research on chemical features of Jingdezhen and Longquan celadon by energy dispersive X-ray fluorescence,” in *Ceramics Int.*, vol. 42, pp. 5123-5129, 2016.
- [JDX14] H. Jia, S. Ding, X. Xu and R. Nie, “The latest research progress on spectral clustering,” *Neural Comput. Appl.*, vol. 24, pp. 1477-1486, 2014.
- [JJ19] J. Jang and H. Jiang, “DBSCAN++: Towards fast and scalable density clustering,” in *Proc. Int. Conf. Mach. Learn.*, pp. 3019-3029, 2019.
- [JV19] R. Janani and S. Vijayarani, “Text document clustering using spectral clustering algorithm with particle swarm optimization,” *Expert Syst. Appl.*, vol. 134, pp. 192-200, 2019.
- [JZH22] H. Ji, Z. Zuo and Q. -L. Han, “A divisive hierarchical clustering approach to hyperspectral band selection,” *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1-12, 2022.
- [KFE12] R. Killick, P. Fearnhead and I. A. Eckley, “Optimal detection of changepoints with a linear computational cost,” *J. Am. Stat. Assoc.*, vol. 107, pp. 1590-1598, 2012.
- [KKV15] D. Koutra, U. Kang, J. Vreeken and C. Faloutsos, “Summarizing and understanding large graphs,” *Stat. Anal. Data Min. : ASA Data Sci. J.*, vol. 8, pp. 183-202, 2015.

- [KL12] A. Kalogeratos and A. Likas, “Dip-means: An incremental clustering method for estimating the number of clusters,” in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 2393-2401, 2012.
- [KLJ09] J. A. Kelner, J. R. Lee, R. James, G. N. Price and S. -H. Teng, “Higher eigenvalues of graphs,” in *Proc. 50th Annu. IEEE Symp. Found. Comput. Sci.*, pp. 735-744, 2009.
- [KM22] T. Koka and M. Muma, “Fast and Sample Accurate R-Peak Detection for Noisy ECG Using Visibility Graphs,” in *44th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, pp. 121-126, 2022.
- [KN11] B. Karrer and M. E. J. Newman, “Stochastic blockmodels and community structure in networks,” *Phys. Rev. E*, vol. 83, pp. 016107, 2011.
- [Koh90] T. Kohonen, “The self-organizing map,” in *Proc. IEEE*, vol. 78, pp. 1464-1480, 1990.
- [KSZ08] H. P. Kriegel, M. Schubert and A. Zimek, “Angle-based outlier detection in high-dimensional data,” in *Proc. 14th ACM SIGKDD Intl. Conf. Knowl. Discovery and Data Mining*, ACM, pp. 444-452, 2008.
- [KWC19] Z. Kang, L. Wen, W. Chen and Z. Xu, “Low-rank kernel learning for graph-based clustering,” *Knowledge-Based Syst.*, vol. 163, pp. 510-517, 2019.
- [KXF16] E. Kodirov, T. Xiang, Z. Fu and S. Gong, “Person Re-Identification by Unsupervised ℓ_1 Graph Learning,” in *Proc. Comput. Vision—ECCV 2016: 14th Eur. Conf.*, pp. 178-195, 2016.
- [KY19] Z. Kong and X. Yang, “Color image and multispectral image denoising using block diagonal representation,” *IEEE Trans. Image Process.*, vol. 28, pp. 4247-4259, 2019.
- [LAK98] M. Lyons, S. Akamatsu, M. Kamachi and J. Gyoba, “Coding facial expressions with gabor wavelets,” in *Proc. 3rd IEEE Int. Conf. Autom. Face Gesture Recognit.*, pp. 200-205, 1998.
- [LB14] P. Di Lorenzo and S. Barbarossa, “Distributed estimation and control of algebraic connectivity over random graphs” *IEEE Trans. Signal Process.*, vol. 62, pp. 5615-5628, 2014.
- [LDK06] K. D. Lafferty, A. P. Dobson and A. M. Kuris, “Parasites dominate food web links,” in *Proc. Natl. Acad. Sci.*, vol. 103, pp. 11211-11216, 2006.

- [LFL18] C. Lu, J. Feng, Z. Lin, T. Mei and S. Yan, “Subspace clustering by block diagonal representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, pp. 487-501, 2018.
- [LFR08] A. Lancichinetti, S. Fortunato and F. Radicchi, “Benchmark graphs for testing community detection algorithms,” *Phys. Rev. E*, vol. 78, pp. 1373-1396, 2008.
- [LG02] L. Lee, W. E. L. Grimson, “Gait analysis for recognition and classification,” in *Proc. of 5th IEEE Intl. Conf. Autom. Face Gesture Recog.*, pp. 155-162, 2002.
- [LH18] J. Liu and J. Han, *Data clustering*, Chapman and Hall/CRC, pp. 177-200, 2018.
- [LHN06] E. A. Leicht, P. Holme and M. E. J. Newman, “Vertex similarity in networks,” *Phys. Rev. E*, vol. 73, pp. 026120, 2006.
- [LKJ20] L. Liu, L. Kuang and Y. Ji, “Multimodal MRI brain tumor image segmentation using sparse subspace clustering algorithm,” *Comput. Math. Methods Med.*, 2020.
- [LLB08] L. Lacasa, B. Luque, F. Ballesteros, J. Luque and J. C. Nuno, “From time series to complex networks: The visibility graph,” *Proc. Natl. Acad. Sci.*, vol. 105, pp. 4972-4975, 2008.
- [LLO20] Z. Liu, Z. Lai, W. Ou, K. Zhang and R. Zheng, “Structured optimal graph based sparse feature extraction for semi-supervised learning” *Signal Process.*, vol. 170, pp. 107456, 2020.
- [LLY10] G. Liu, Z. Lin and Y. Yu, “Robust subspace segmentation by low-rank representation,” *Icml.*, vol. 1, pp. 8, 2010.
- [LLY12] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu and Y. Ma, “Robust recovery of subspace structures by low-rank representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, pp. 171-184, 2012.
- [LLZ15] C. -G. Li, Z. Lin, H. Zhang and J. Guo, “Learning semi-supervised representation towards a unified optimization framework for semi-supervised learning,” in *Proc. IEEE Conf. Comp. Vision*, pp. 2767-2775, 2015.
- [LM14] A. Louis and K. Makarychev, “Approximation algorithm for sparsest k-partitioning” in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms*, pp. 1244-1255, 2014.

- [LMZ12] C. -Y Lu, H. Min, Z. -Q. Zhao, L. Zhu, D. -S. Huang and S. Yan, “Robust and efficient subspace segmentation via least squares regression,” in *Proc. Eur. Conf. Comp. Vision*, pp. 347-360, 2012.
- [LNC18] Z. Li, F. Nie, X. Chang, L. Nie, H. Zhang and Y. Yang, “Rank-constrained spectral clustering with flexible embedding,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, pp. 6073-6082, 2018.
- [LP19] J. Liu, and D. P. Palomar, “Regularized robust estimation of mean and covariance matrix for incomplete data” *IEEE Signal Process.*, vol. 165, pp. 278-291, 2019.
- [LR99] T. Leighton and S. Rao, “Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms” *J. ACM*, vol. 46, pp. 787-832, 1999.
- [LSB03] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, “The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations,” *Behav. Ecol. Sociobiol.*, vol. 54, pp. 396-405, 2003.
- [LSW16] Q. Liu, Y. Sun, C. Wang, T. Liu and D. Tao, “Elastic net hypergraph learning for image clustering and semi-supervised classification,” *IEEE Trans. Image Process.*, vol. 26, pp. 452-463, 2016.
- [LT09] J. Lu and Y. -P. Tan, “Regularized locality preserving projections and its extensions for face recognition,” *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 40, pp. 958-963, 2009.
- [Lux07] U. Von Luxburg, “A tutorial on spectral clustering,” *Stat. Comput.*, vol. 17, pp. 395-416, 2007.
- [LW12] M. Lucińska and S. T. Wierzchoń, “Spectral clustering based on k -nearest neighbor graph,” in *Proc. Int. Conf. Comput. Inf. Syst. Ind. Manage.*, pp. 254-265, 2012.
- [LWS20] M. Liu, Y. Wang, J. Sun and Z. Ji, “Structured block diagonal representation for subspace clustering,” *Appl. Intell.*, vol. 50, pp. 2523-2536, 2020.
- [LY11] G. Liu and S. Yan, “Latent low-rank representation for subspace segmentation and feature extraction,” in *Proc. Int. Conf. Comp. Vision*, pp. 1615-1622, 2011.

- [LZX20] W. Liang, S. Zhou, J. Xiong, X. Liu, S. Wang, E. Zhu, Z. Cai, X. Xu, “Multi-view spectral clustering with high-order optimal neighborhood Laplacian matrix,” *IEEE Trans. Knowl. Data Eng.*, 2020.
- [MB06] N. Meinshausen and P. Bühlmann, “High-dimensional graphs and variable selection with the lasso,” *Ann. Stat.*, vol. 34, pp. 1436-1462, 2006.
- [MB10] N. Meinshausen and P. Bühlmann, “Stability selection,” *J. R. Stat. Soc. Ser. B* 72, vol. 72, pp. 417-473, 2010.
- [MDD18] S. S. Mullick, S. Datta and S. Das, “Adaptive learning-based k -nearest neighbor classifiers with resilience to class imbalance,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, pp. 5713-5725, 2018.
- [MDG21] N. Monath, K. A. Dubey, G. Guruganesh, M. Zaheer, A. Ahmed, A. McCallum, G. Mergen, M. Najork, M. Terzihan, B. Tjanaka, Y. Wang and W. Yuchen, “Scalable Hierarchical Agglomerative Clustering,” in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Min.*, pp. 1245-1255, 2021.
- [MH11] E. L. Martelot and C. Hankin, “Multi-scale community detection using stability as optimization criterion in a greedy algorithm,” in *Proc. Intl. Conf. Knowl. Discovery and Inf. Retrieval*, pp. 208-217, 2011.
- [MH12] R. Mazumder and T. Hastie, “The graphical Lasso: New insights and alternatives” *Electron. J. Stat.*, vol. 6, pp. 2125, 2012.
- [MMY19] R. A. Maronna, R. D. Martin, V. J. Yohai and M. Salibián-Barrera, *Robust statistics: theory and methods (with R)*, John Wiley & Sons, 2019.
- [MVO20] J. Miettinen, S. A. Vorobyov, E. Ollila, “Modelling graph errors: Towards robust graph signal processing,” Online-Edition: <https://arxiv.org/abs/1903.08398>, 2020.
- [NC11] M. C. V. Nascimento and A. C. P. L. F. De Carvalho, “Spectral methods for graph clustering-a survey,” *Eur. J. Oper. Res.*, vol. 211, pp. 221-231, 2011.
- [New03] M. E. J. Newman, “Random graphs as models of networks,” *Handb. Graphs Networks*, vol. 1, pp. 35-68, 2003.

- [New04] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Phys. Rev. E*, vol. 69, pp. 066133, 2004.
- [New06] M. E. J. Newman, “Modularity and community structure in networks,” in *Proc. Natl. Acad. Sci.*, vol. 103, pp. 8577-8582, 2006.
- [New06] M. E. J. Newman, “Finding community structure in networks using the eigenvectors of matrices,” *Phys. Rev. E*, vol. 74, pp. 036104, 2006.
- [New13] M. E. J. Newman, “Community detection and graph partitioning,” *Europhys. Lett.*, vol. 103, p. 28003, 2013.
- [New18] M. Newman, *Networks*, Oxford university press, 2018.
- [NH11] B. Nasihatkon and R. Hartley, “Graph connectivity in sparse subspace clustering,” in *Proc. CVPR 2011*, pp. 2137-2144, 2011.
- [NHG19] A. Nazi, W. Hang, A. Goldie, S. Ravi and A. Mirhoseini, “Gap: Generalizable approximate graph partitioning framework,” Online-Edition: <https://arxiv.org/abs/1903.00614>, 2019.
- [NJW01] A. Y. Ng, M. I. Jordan and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” *Adv. Process. Neural Inf. Syst.*, vol. 14, pp. 849-856, 2001.
- [NNM95] S. A. Nene, S. K. Nayar and H. Murase, “Columbia object image library (coil-20),” 1995.
- [NPC16] L. Naranjo, C. J. Perez, Y. Campos-Roca and J. Martin, “Addressing voice recording replications for Parkinson’s disease detection,” *Expert Syst. Appl.*, vol. 46, pp. 286-292, 2016.
- [NRG19] A. Nguyen, N. Roth, N. H. Ghassemi, J. Hannink, T. Seel, J. Klucken, H. Gassner and B. M. Eskofier, “Development and clinical validation of inertial sensor-based gait-clustering methods in Parkinson’s disease,” *J. NeuroEng. Rehabil.*, vol. 16, pp. 1-14, 2019.
- [NWD16] F. Nie, H. Wang, C. Deng, X. Gao, X. Li and H. Huang, “New ℓ_1 -norm relaxations and optimizations for graph clustering,” in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016.
- [OFK18] A. Ortega, P. Frossard, J. Kovačević, J.M.F. Moura and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” *Proc. IEEE*, vol. 106, pp. 808-828, 2018.

- [OP09] A. B. Owen and P. O. Perry, “Bi-cross-validation of the SVD and the nonnegative matrix factorization,” *Ann. Appl. Stat.*, vol. 3, pp. 564-594, 2009.
- [OS05] P. Orponen and S. E. Schaeffer, “Local clustering of large graphs by approximate Fiedler vectors,” in *Proc. Int. Work. Exp. Effic. Algorithms*, pp. 524-533, 2005.
- [OT14] E. Ollila, and D. E. Tyler, “Regularized M -estimators of scatter matrix” *IEEE Trans. Signal Process.*, vol. 62, pp. 6059-6070, 2014.
- [Ote05] M. Otero, “Application of a continuous wave radar for human gait recognition,” *Signal Process., Sens. Fusion, and Target Recog. XIV, Intl. Society for Optics and Photonics*, vol. 5809, pp. 538-548, 2005.
- [PEC19] B. Phipson, X. P. Er, A. N. Combes, T. A. Forbes, S. E. Howden, L. Zappia, H. -J. Yen, K. T. Lawlor, L. J. Hale, J. Sun, E. Wolvetang, M. Takasato¹, A. Oshlack and M. H. Little, “Evaluation of variability in human kidney organoids,” *Nat. Methods*, vol. 16, pp. 79-87, 2019.
- [PM00] D. Pelleg and A. Moore, “X-means: Extending K-means with efficient estimation of the number of clusters,” in *Proc. 17th Int. Conf. Mach. Learn. (ICML)*, pp. 727-734, 2000.
- [PRE11] I. Psorakis, S. Roberts, M. Ebdon and B. Sheldon, “Overlapping community detection using Bayesian non-negative matrix factorization,” *Phys. Rev. E*, vol. 83, p. 066114, 2011.
- [PS82] C. C. Paige and M. A. Saunders, “Algorithm 583 LSQR: Sparse linear equations and least squares problems,” *ACM Transactions on Mathematical Software*, vol. 8, pp. 195-209, 1982.
- [PWA15] C. Pradhan, M. Wuehr, F. Akrami, M. Neuhaeusser, S. Huth, T. Brandt, K. Jahn and R. Schniepp, “Automated classification of neurological disorders of gait using spatio-temporal gait parameters,” *J. Electromyography and Kinesiology*, vol. 25, pp. 413-422, 2015.
- [PYT15] X. Peng, Z. Yi and H. Tang, “Robust subspace clustering via thresholding ridge regression,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, 2015.
- [PZ18] S. Park and H. Zhao, “Spectral clustering based on learning similarity matrix,” *Bioinf.*, vol. 34, pp. 2069-2076, 2018.

- [QWZ21] Y. Qin, H. Wu, X. Zhang and G. Feng, "Semi-Supervised Structured Subspace Learning for Multi-View Clustering," *IEEE Trans. Image Process.*, vol. 31, pp. 1-14, 2021.
- [RA17] M. Rahmani and G. K. Atia, "Innovation pursuit: A new approach to subspace clustering," *IEEE Trans. Signal Process.*, vol. 65, pp. 6276-6291, 2017.
- [Ras99] C. Rasmussen, "The infinite Gaussian mixture model," *Adv. Neural Inf. Process. Syst.*, vol. 12, 1999.
- [RB90] P. J. Rousseeuw and G. W. Bassett, "The remedian: A robust averaging method for large data sets", *Journal of the American Statistical Association*, vol 85, pp. 97-104, 1990.
- [RB18] P. J. Rousseeuw and V. D. W. Bossche, "Detecting deviating data cells," *Technometrics*, vol. 60, pp. 135-145, 2018.
- [RL05] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*, John Wiley, vol. 589, 2005.
- [ROK12] S. A. Razavi, E. Ollila, V. Koivunen, "Robust greedy algorithms for compressed sensing," *Proc. of the 20th European Signal Process. Conf.*, pp. 969-973, 2012.
- [RS00] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Sci.*, vol. 290, pp. 2323-2326, 2000.
- [RSB11] A. R. Rocha Neto, R. Sousa, G. A. Barreto and J. S. Cardoso, "Diagnostic of pathology on the vertebral column with embedded reject option," in *Proc. Iberian Conf. Pattern Recognit. Image Anal.*, pp. 588-595, 2011.
- [RTV08] S. R. Rao, R. Tron, R. Vidal and Y. Ma, "Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, pp. 1-8, 2008.
- [SAZ19] A. -K. Seifert, M. Amin and A. M. Zoubir, "Toward unobtrusive in-home gait analysis based on radar micro-Doppler signatures," *IEEE Trans. Biomed. Eng.*, vol. 66, pp. 1-11, 2019.
- [SBH16] C. Schulz, S. K. Bayer, C. Hess, C. Steiger, M. Teichmann, J. Jacob, F. Bernardes-lima, R. Hangu and S. Hayrapetyan, "Course notes: Graph partitioning and graph clustering in theory and practice," *Karlsruhe Inst. Technol.*, 2016.

- [SCB14] S. Sobolevsky, R. Campari, A. Belyi and C. Ratti, “General optimization technique for high-quality community detection in complex networks,” *Phys. Rev. E*, vol. 90, pp. 012811, 2014.
- [Sch07] S. E. Schaeffer, “Graph clustering,” *Comput. Sci. Rev.*, vol. 1, pp. 27-64, 2007.
- [SD11] S. Sarkar and A. Dong, “Community detection in graphs using singular value decomposition,” *Phys. Rev. E*, vol. 83, pp. 046114, 2011.
- [SGY15] M. Stephen, C. Gu and H. Yang, “Visibility graph based time series analysis,” *PloS One*, vol. 10, pp. e0143015, 2015.
- [SH94] F. S. Samaria and A. C. Harter, “Parameterisation of a stochastic model for human face identification,” in *Proc. IEEE Int. Workshop Appl. Comput. Vision*, pp. 138-142, 1994.
- [SHR14] B. Y. Su, K. C. Ho, M. J. Rantz and M. Skubic, “Doppler radar fall activity detection using the wavelet transform,” *IEEE Trans. Biomed. Eng.*, vol. 62, pp. 865-875, 2014.
- [Sil00] J. R. Silvester, “Determinants of block matrices”, *Math. Gazz.*, vol 84, pp. 460-467, 2000.
- [SLL19] J. Sui, Z. Liu, L. Liu, A. Jung, T. Liu, B. Peng and X. Li, “Sparse subspace clustering for evolving data streams,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.(ICASSP)*, pp. 7455-7459, 2019.
- [SM00] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 888-905, 2000.
- [SM21] C. A. Schroth and M. Muma, “Robust M-estimation based bayesian cluster enumeration for real elliptically symmetric distributions,” *IEEE Trans. Signal Process.*, vol. 69, pp. 3525-3540, 2021.
- [SPG17] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding and C. -T. Lin, “A review of clustering techniques and developments,” *Neurocomput.*, vol. 267, pp. 664-681, 2017.
- [Spi12] D. Spielman, *Spectral graph theory*, CRC Press Boca Raton, 2012.
- [SR19] E. Schubert and P. J. Rousseeuw, “Faster k-medoids clustering: improving the PAM, CLARA, and CLARANS algorithms,” in *Proc. Int. Conf. Similarity Search Appl.*, pp. 171-187, 2019.

- [SSS19] S. Sarfraz, V. Sharma and R. Stiefelham, “Efficient parameter-free clustering using first neighbor relations,” in *Proc. of the IEEE Conf. Comp. Vision Pattern Recog.*, pp. 8934-8943, 2019.
- [ST96] D. A. Spielman and S. -Hua Teng, “Spectral partitioning works: Planar graphs and finite element meshes”, in *Proc. 37th Conf. Found. Comput. Sci.*, pp. 96-105, 1996.
- [ST07] D. A. Spielman and S. -H. Teng, “Spectral partitioning works: Planar graphs and finite element meshes,” in *Linear Algebra Appl.*, vol. 421, pp. 284-305, 2007.
- [Ste98] G. W. Stewart, *Matrix Algorithms: Volume I Basic Decompositions*, Society for Industrial and Applied Mathematics, 1998.
- [Ste01] G. W. Stewart, *Matrix Algorithms: Volume II Eigensystems*, Society for Industrial and Applied Mathematics, 2001.
- [Ste02] G. W. Stewart, “A Krylov–Schur algorithm for large eigenproblems”, *SIAM Journal on Matrix Analysis and Applications*, vol. 23, pp. 601-614, 2002.
- [SVB01] S. Shahbazpanahi, S. Valaee and M. H. Bastani, “Distributed source localization using ESPRIT algorithm,” *IEEE Trans. Signal Process.*, vol. 49, pp. 2169-2178, 2001.
- [SWH89] V. G. Sigilitto, S. P. Wing, L. V. Hutton and K. B. Baker, “Classification of radar returns from the ionosphere using neural networks,” *Johns Hopkins APL Tech. Dig.*, vol. 10, pp. 262–266, 1989.
- [SY20] K. P. Sinaga and M. -S. Yang, “Unsupervised K-means clustering algorithm,” *IEEE access*, vol. 8, pp. 80716-80727, 2020.
- [SZL19] D. Shi, L. Zhu, Y. Li, J. Li and X. Nie, “Robust structured graph clustering,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, pp. 4424-4436, 2019.
- [TD20] E. Tam and D. Dunson, “Fiedler regularization: Learning neural networks with graph sparsity” 2020. [Online]. Available: <https://arxiv.org/abs/2003.00992>
- [THW16] Q. Tao, X. Huang, S. Wang, X. Xi and L. Li, “Multiple Gaussian graphical estimation with jointly sparse penalty,” *Signal Process.*, vol. 128, pp. 88-97, 2016.

- [TMO23] A. Taştan, M. Muma, E. Ollila and A. M. Zoubir, “Sparsity-aware block diagonal representation for subspace clustering,” in *Proc. 30th European Signal Process. Conf. (submitted)*, 2023.
- [TMZ18] F. K. Teklehaymanot, M. Muma and A. M. Zoubir, “Bayesian cluster enumeration criterion for unsupervised learning,” *IEEE Trans. Signal Process.*, vol. 66, pp. 5392-5406, 2018.
- [TMZ20] A. Taştan, M. Muma and A. M. Zoubir, “An unsupervised approach for graph-based robust clustering of human gait signatures,” in *Proc. 2020 IEEE Radar Conf.*, pp. 1-6, 2020.
- [TMZ21] A. Taştan, M. Muma and A. M. Zoubir, “Sparsity-aware Robust Community Detection,” *Signal Process.*, vol. 187, pp. 108147, 2021.
- [TMZ21] F. K. Teklehaymanot, M. Muma and A. M. Zoubir, “Robust Bayesian cluster enumeration based on the t distribution,” *Signal Process.*, vol. 182, p. 107870, 2021.
- [TMZ21] A. Taştan, M. Muma and A. M. Zoubir, “Robust Spectral Clustering: A Locality Preserving Feature Mapping Based on M-estimation,” in *Proc. 29th European Signal Process. Conf.*, pp. 851-855, 2021.
- [TMZ22] A. Taştan, M. Muma and A. M. Zoubir, “Eigenvalue-Based Block Diagonal Representation and Application to p -Nearest Neighbor Graphs,” in *Proc. 30th European Signal Process. Conf.*, pp. 1761-1765, 2022.
- [TMZ22] A. Taştan, M. Muma and A. M. Zoubir, “Robust regularized locality preserving indexing for Fiedler vector estimation,” *Signal Process. (accepted)*, 2022.
- [TMZ23] A. Taştan, M. Muma and A. M. Zoubir, “Fast and robust sparsity-aware block diagonal representation,” *IEEE Trans. Signal Process. (submitted)*, 2023.
- [TSM18] F. K. Teklehaymanot, A. -K. Seifert, M. Muma, M. G. Amin and A. M. Zoubir, “Bayesian target enumeration and labeling using radar data of human gait,” in *Proc. 26th European Signal Process. Conf. (EUSIPCO)*, pp. 1342-1346, 2018.
- [TV17] M. C. Tsakiris and R. Vidal, “Filtrated algebraic subspace clustering,” *SIAM J. Imag. Sci.*, vol. 10, pp. 372-415, 2017.

- [TV17] M. C. Tsakiris and R. Vidal, “Algebraic clustering of affine subspaces,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, pp. 482-489, 2017.
- [TWS15] K. M. Tan, D. Witten and A. Shojaie, “The cluster graphical Lasso for improved estimation of Gaussian graphical models,” *Comput. Stat. Data Anal.*, vol. 85, pp. 23-36, 2015.
- [TWS19] T. Tian, J. Wan, Q. Song and Z. Wei, “Clustering single-cell RNA-seq data with a model-based deep learning approach,” *Nat. Mach. Intell.*, vol. 1, pp. 191-198, 2019.
- [UST09] T. Uno, M. Sugiyama and K. Tsuda, “Efficient construction of neighborhood graphs by the multiple sorting method,” Online-Edition: <https://arxiv.org/abs/0904.3151>, 2009.
- [WBH15] F. Wahid, R. K. Begg, C. J. Hass, S. Halgamuge and D. C. Ackland, “Classification of Parkinson’s disease gait using spatial-temporal gait features,” *IEEE J. Biomed. Health Informat.*, vol. 19, pp. 1794-1802, 2015.
- [Wes01] D. B. West, *Introduction to graph theory*, Prentice hall, 2001.
- [WHG15] F. Wu, Y. Hu, J. Gao, Y. Sun and Yin. B, “Ordered subspace clustering with block diagonal priors,” *IEEE Trans. Cybern.*, vol. 46, pp. 3209-3219, 2015.
- [WLW18] Z. Wang, Z. Li, R. Wang, F. Nie and X. Li, “Large graph clustering with simultaneous spectral embedding and discretization,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, pp. 4426-4440, 2020.
- [WM89] W. H. Wolberg and O. L. Mangasarian, “Multisurface method of pattern separation applied to breast cytology diagnosis,” in *Proc. Natl. Acad. Sci.*, vol. 87, pp. 9193-9196, 1989.
- [WMM10] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang and S. Yan, “Sparse representation for computer vision and pattern recognition,” *Proc. IEEE*, vol. 98, pp. 1031-1044, 2010.
- [WPM14] P. K. Wagner, S. M. Peres, R. C. B. Madeo, C. A. M. Lima and F. A. Freitas, “Gesture unit segmentation using spatial-temporal information and machine learning,” in *Proc. 27th Int. Flairs Conf.*, pp. 101-106, 2014.
- [WQD14] X. Wang, B. Qian and I. Davidson, “On constrained spectral clustering and its applications,” *Data Min. Knowl. Discovery*, vol. 28, pp. 1-30, 2014.

- [WSR14] F. Wang, M. Skubic, M. Rantz and P. E. Cuddihy, “Quantitative gait measurement with pulse-Doppler radar for passive in-home gait assessment,” *IEEE Trans. Biomed. Eng.*, vol. 61, pp. 2434-2443, 2014.
- [WYG08] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, pp. 210-227, 2008.
- [WZW17] J. Wang, K. Zhang, P. Wang, K. Madani and C. Sabourin, “Unsupervised band selection using block diagonal sparsity for hyperspectral image classification,” *IEEE Trans. Geosci. Remote Sens. Lett.*, vol. 14, pp. 2062-2066, 2017.
- [XG08] T. Xiang and S. Gong, “Spectral clustering with eigenvector selection,” *Pattern Recognit.*, vol. 41, pp. 1012-1029, 2008.
- [XGL17] X. Xie, X. Guo, G. Liu and J. Wang, “Implicit block diagonal low-rank representation,” *IEEE Trans. Image Process.*, vol. 27, pp. 477-489, 2017.
- [XGZ71] K. Xia, X. Gu and Y. Zhang, “Oriented grouping-constrained spectral clustering for medical imaging segmentation,” *Multimedia Syst.*, vol. 26, pp. 27-36, 2020.
- [XT15] D. Xu and Y. Tian, “A comprehensive survey of clustering algorithms,” *Ann. Data Sci.*, vol. 2, pp. 165-193, 2015.
- [XTX15] S. Xiao, M. Tan, D. Xu and Z.Y. Dong, “Robust kernel low-rank representation,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, pp. 2268-2281, 2015.
- [XWW21] C. Xing, M. Wang, Z. Wang, C. Duan and Y. Liu, “Diagonalized Low-Rank Learning for Hyperspectral Image Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1-12, 2021.
- [YAI19] M. K. Yöntem, K. Adem, T. İlhan and S. Kılıçarslan, “Divorce prediction using correlation based feature selection and artificial neural networks,” *J. Nevşehir Hacı Bektaş Veli University SBE*, vol. 9, pp. 259-273, 2019.
- [YCL20] C. Yan, X. Chang, M. Luo, Q. Zheng, X. Zhang, Z. Li and F. Nie, “Self-weighted robust LDA for multiclass classification with edge classes,” *ACM Trans. Intell. Syst. Technol.*, vol. 12, pp. 1-19, 2020.

- [YFG10] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa and R. Sukthankar, “Decentralized estimation and control of graph connectivity for mobile sensor networks” *Autom.*, vol. 46, pp. 390-396, 2010.
- [YHJ20] S. Yazdanparast, T. C. Havens and M. Jamalabdollahi, “Soft overlapping community detection in large-scale networks via fast fuzzy modularity maximization,” *IEEE Trans. Fuzzy Syst.*, vol. 29, pp. 1533-1543, 2020.
- [YL07] M. Yuan and Y. Lin, “Model selection and estimation in the Gaussian graphical model”, *Biometrika*, vol. 94, pp. 19-35, 2007.
- [YL15] J. J. Yang and J. Leskovec, “Defining and evaluating network communities based on groundtruth,” *Knowl. Inf. Syst.*, vol. 42, pp. 181-213, 2015.
- [YLR16] C. You, C. -G. Li, D. P. Robinson and R. Vidal, “Oracle based active set algorithm for scalable elastic net subspace clustering,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, pp. 3928-3937, 2016.
- [YTL06] W. Yu, X. Teng and C. Liu, “Face recognition using discriminant locality preserving projections,” in *Image Vision Comput.*, vol. 24, pp. 239-248, 2006.
- [YYZ19] X. Yang, H. Yang, F. Zhang, L. Zhang, X. Fan, Q. Ye and L. Fu, “Piecewise linear regression based on plane clustering,” *IEEE Access*, vol. 7, pp. 29845-29855, 2019.
- [Zac77] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *J. Anthropological Res.*, vol. 33, pp. 452-473, 1977.
- [ZCS19] R. Zhou, X. Chang, L. Shi, Y.-D. Shen, Y. Yang and F. Nie, “Person reidentification via multi-feature fusion with adaptive graph learning,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, pp. 1592-1601, 2019.
- [Zha71] C. T. Zahn, “Graph-theoretical methods for detecting and describing gestalt clusters,” *IEEE Trans. Comput.*, vol. 100, pp. 68-86, 1971.
- [Zhu08] X. Zhu, “Semi-supervised learning literature survey,” *Comput. Sci.*, vol. 37, pp. 63-67, 2008.

- [ZKC12] A. M. Zoubir, V. Koivunen, Y. Chakhchoukh and M. Muma, "Robust estimation in signal processing: A tutorial-style treatment of fundamental concepts," in *PIEEE Signal Process. Mag.*, vol. 29, pp. 61-80, 2012.
- [ZKO18] A. M. Zoubir, V. Koivunen, E. Ollila and M. Muma, *Robust statistics for signal processing*, Cambridge, 2018.
- [ZP04] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," *Adv. Neural Inf. Process. Syst.*, vol. 17, 2004.
- [ZPW07] Z. Zhang, P. Pouliquen, A. Waxman and A. G. Andreou, "Acoustic micro-Doppler gait signatures of humans and animals," in *41st Annual Conf. Inf. Sci. and Syst.*, pp. 627-630, 2007.
- [ZR15] M. E. Zhukovskii and A. M. Raigorodskii, "Random graphs: models and asymptotic characteristics," *Russ. Math. Surv.*, vol. 70, pp. 33, 2015.
- [ZXS17] Z. Zhang, Y. Xu, L. Shao and J. Yang, "Discriminative block diagonal representation learning for image recognition," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, pp. 3111-3125, 2017.
- [ZZL18] X. Zhu, S. Zhang, Y. Li, J. Zhang, L. Yang and Y. Fang, "Low-rank sparse subspace for spectral clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 31, pp. 1532-1543, 2018.
- [ZZL18] Y. Zhu, C. Zhu and X. Li, "Improved principal component analysis and linear regression classification for face recognition," *Signal Process.*, vol. 145, pp. 175-182, 2018.
- [ZZP17] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. Conf. Comput. Vision Pattern Recognit.*, pp. 633-641, 2017.

Erklärungen laut Promotionsordnung

§ 8 ABS. 1 LIT. C PROMO

Ich versichere hiermit, dass die elektronische Version meiner Dissertation mit der schriftlichen Version übereinstimmt.

§ 8 ABS. 1 LIT. D PROMO

Ich versichere hiermit, dass zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde. In diesem Fall sind nähere Angaben über Zeitpunkt, Hochschule, Dissertationsthema und Ergebnis dieses Versuchs mitzuteilen.

§ 9 ABS. 1 PROMO

Ich versichere hiermit, dass die vorliegende Dissertation selbstständig und nur unter Verwendung der angegebenen Quellen verfasst wurde.

§ 9 ABS. 2 PROMO

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 27.03.2023

Aylin Taştan

Curriculum Vitae

Name: Aylin Tařtan
Date of birth: 31.03.1994
Place of birth: Ankara, Turkey

EDUCATION

08/2016 - 11/2018 Gazi University, Turkey
Electrical and Electronics Engineering
Master of Science (M.Sc.)
09/2012 - 06/2016 Gazi University, Turkey
Electrical and Electronics Engineering
Bachelor of Science (B.Sc.)
09/2008 - 06/2012 Dr. Binnaz - Rıdvan Ege Anatolian Highschool, Turkey

WORK EXPERIENCE

since 04/2019 Research Associate
Signal Processing Group
Technische Universitat Darmstadt, Germany
04/2022 - 05/2022 Visiting Ph.D. Student
Department of Information and Communications
Engineering
Aalto University, Finland
10/2016 - 04/2018 Research Associate
Department of Electrical and Electronics Engineering
Bařkent University, Turkey