Sampling and Subspace Methods for Learning Sparse Group Structures in Computer Vision

2018

Maryam Jaberi
University of Central Florida

SAMPLING AND SUBSPACE METHODS FOR LEARNING SPARSE GROUP STRUCTURES IN COMPUTER VISION

by

MARYAM JABERI
M.S. University of Nevada Reno, 2012

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Sciences
at the University of Central Florida
Orlando, Florida

Spring Term
2018

Major Professor: Hassan Foroosh

ABSTRACT

The unprecedented growth of data in volume and dimension has led to an increased number of computationally-demanding and data-driven decision-making methods in many disciplines, such as computer vision, genomics, finance, etc. Research on big data aims to understand and describe trends in massive volumes of high-dimensional data. High volume and dimension are the determining factors in both computational and time complexity of algorithms. The challenge grows when the data are formed of the union of group-structures of different dimensions embedded in a high-dimensional ambient space. To address the problem of high volume, we propose a sampling method referred to as the Sparse Withdrawal of Inliers in a First Trial (SWIFT), which determines the smallest sample size in one grab so that all group-structures are adequately represented and discovered with high probability. The key features of SWIFT are: (i) sparsity, which is independent of the population size; (ii) no prior knowledge of the distribution of data, or the number of underlying group-structures; and (iii) robustness in the presence of an overwhelming number of outliers. We report a comprehensive study of the proposed sampling method in terms of accuracy, functionality, and effectiveness in reducing the computational cost in various applications of computer vision. In the second part of this dissertation, we study dimensionality reduction for multi-structural data. We propose a probabilistic subspace clustering method that unifies soft- and hard-clustering in a single framework. This is achieved by introducing a delayed association of uncertain points to subspaces of lower dimensions based on a confidence measure. Delayed association yields higher accuracy in clustering subspaces that have ambiguities, i.e. due to intersections and high-level of outliers/noise, and hence leads to more accurate self-representation of underlying subspaces. Altogether, this dissertation addresses the key theoretical and practically issues of size and dimension in big data analysis.

To my mom and dad

Thank you for all your support, love, and guidance.

ACKNOWLEDGMENTS

Over the past six years, I have received support and encouragement from numerous individuals. First, I would like to thank my doctoral adviser, Dr. Hassan Foroosh, for his invaluable insights and much-needed support. His guidance has made this a thoughtful and rewarding journey. I would also like to thank my doctoral co-adviser, Dr. Marianna Pensky, for her friendly guidance, thought provoking suggestions, and many valuable contributions. My sincere appreciation is extended to the members of my dissertation committee, Dr. Guo-Jun Qi and Dr. Boqing Gong. Their contribution was crucial to a timely completion of this dissertation. I owe many thanks to Jussi Doherty for his encouragement and hours of proofreading. My most heartfelt gratitude is for my parents and my sisters, whose love and guidance are with me in all my pursuits.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Understanding the underlying patterns and trends in data is an essential step in a wide range of application areas including computer vision, machine learning, and pattern recognition, etc. In recent years, the unprecedented growth in data in terms of size and dimension has made the need for fast and accurate methods of detecting patterns in data even more apparent. Sophisticated tools of machine learning, mathematics, and statistics are thus needed to tackle these hard problems of data mining and knowledge acquisition that are overwhelmingly affected by the sheer amount of data. Research on big data analysis aims to understand and describe massive volumes of high-dimensional data. High volume is the primary characteristic of big data, where the large scale is a dominant factor in determining the computational complexity. An important and effective strategy in analyzing high-volume data is sampling, whereby a smaller subset of data is used to estimate the characteristics of its entire population. A second important factor contributing to the complexity of big data is the high dimension of analysis space. Handling high-dimensional data is computationally expensive. Thus, dimensionality reduction is a fundamental step in the uncovering of group structures in the data. Sparse sampling and dimensionality reduction become even more challenging when the data is formed of the union of multiple group structures of different dimensions embedded in a high-dimensional ambient space, which is a common problem encountered in many applications of computer vision and machine learning. In this dissertation, we studied this issue from two different angles: (1) Reducing the size of data by one-time grab sparse sampling and (2) Looking for the compact representation of high dimensional data by exploiting their structure when data is formed from a union of multiple high dimensional structures.

The first part of this dissertation focuses on the sparse sampling of big data. One problem in big data is the time complexity of detecting and understanding the underlying patterns. We propose a sparse sampling method that can be used as the front-end to any method of processing data and detecting structures. Our goal in this study is to reduce the number of points needed so that all structures are still represented adequately and can be discovered with high probability, despite a substantial ratio of outliers. We impose no constraints on the distribution of points. Thus, the samples are taken uniformly, i.e. requiring no prior knowledge of the distribution of data. Moreover, the proposed method can handle structures with different dimensions.

Another issue with a wide range of applications in computer vision, machine learning, and pattern recognition is the discovery of clustering subspaces in high dimensional data. In the second part of this dissertation, we study the behavior of high dimensional data when it is formed from a union of multiple subspaces. The idea behind subspace clustering algorithms is that, when a subset of high dimensional data belongs to a cluster, the points in the cluster lie in a low dimensional subspace. Therefore, each subspace can be represented by a small set of (learned) bases. This becomes more difficult when data are noisy or have ambiguity regarding the correct cluster or when the population size is very large. In this dissertation, we first study the effect of using sparse sampling to reduce the time complexity in subspace clustering. We then propose a probabilistic-based subspace clustering method to improve the accuracy of final clusters.

1.1 Sampling in Multiple Structures Estimation

Estimating and clustering the underlying structures in data are among the most fundamental problems in computer vision, machine learning, and data analytics. However, the unprecedented growth in data with a high ratio of outliers has created a greater demand for faster and more accurate methods [100]. A popular approach explored in the literature to tackle these problems of size and dimensionality is to estimate the characteristics of the entire data population using only sampled subsets of the data. Methods like RANdom SAmpling Consensus (RANSAC) [24], use sampling to find a single structure in the data. In most applications, however, multiple instances of a model (structure) exist simultaneously. Examples include multiple independently moving objects in a video, multiple planes in a scene, or multiple instances of the same face in a database under varying lighting condition. One of the main challenges in such multi-structured data is a high percentage of outliers in the population due to: (i) gross outliers, which are comprised of points that do not belong to any model instance, and (ii) pseudo-outliers, which are points that are inliers to one model instance (structure), but effectively act as outliers to all other model instances in the population [10]. Existing sampling techniques can be broadly divided into two groups: (i) Random iterative (multi-grab) sampling methods that sequentially grab multiple subsets of data in order to fit model instances, and (ii) One-time grab sampling methods that generate all possible hypotheses or model instances from a single subset of data. More details on these two groups of methods are discussed in the next section. However, it is worth noting that a major issue that is often overlooked in one-grab sampling is the question of sufficient statistics, i.e. the optimal sample size to guarantee that all underlying structures or model instances are discovered. In simple terms, if not enough samples are taken, then one may miss some model instances. If too many samples are taken, then the benefits of sampling for reducing the computational cost is diminished or may be lost.

The first part of this dissertation focuses on this latter problem in one-grab sampling. We provide tight upper and lower bounds on the sample size required for discovering all underlying structures. We provide strong theoretical guarantees and confirm them via simulations and experiments. In particular, we verify that our method leads to a much smaller sample size than state-of-the-art methods in the literature [33, 110]. In addition, our experiments on real data demonstrate that the proposed solution reduces computational cost without compromising accuracy. More specifically, we answer the following question:

"*Given a large population of points N with C embedded structures and gross outliers, what is the minimum number of points r to be selected randomly in one grab in order to make sure with probability P that at least ε points are selected on each structure, where ε is the number of degrees* *of freedom of each structure.*"

The answer to this question is significant because of the following reasons: (i) We will show that even under a large number of pseudo-outliers and gross outliers, r is extremely small (i.e. the one-grab sample size is a sparse subset of the population), hence the name, Sparse Withdrawal of Inliers in a First Trial (SWIFT); (ii) Although P is an intricate function of r (difficult to invert), we prove that it is a non-decreasing function, and hence r can be mathematically approximated and found by a simple one-dimensional search, regardless of the dimensionality of the problem; (iii) The sample size r is very slowly growing with the number of embedded structures C, keeping the one-grab sample set sparse even under an overwhelming number of pseudo-outliers; (iv) The method does not assume any prior knowledge about the distribution of data; (v) The sparsity of the sampled subset implies a significant reduction in computation.

1.2 Subspace Clustering

In a large variety of applications, data is presented in a high dimensional space but naturally forms clusters of low dimensional subspaces. Recovering the low-dimensional presentation of data is an important step in understanding the data, estimating the structures and reducing the computational and space complexity of algorithms. Moreover, recovering the low-dimensional structure can help reduce noise in the data. Early methods such as Principal Component Analysis (PCA) [38] try to find a low-dimensional presentation of data when it is drawn from a single high-dimensional space. In a large variety of applications, however, data is derived from a mixture of multiple subspaces in a high dimensional ambient space. In this case, we are interested in finding the best fit for each subspace. In video processing, for instance, recognizing independent motions in dynamic scenes is an essential process for many applications including person and object tracking, gesture recognition, video compression, and motion analysis. These motion trajectories are usually represented by high-dimensional vectors. Yet, they can span low-dimensional linear manifolds [101]. Texture/background modeling and face/image classification [76, 104] are used in many applications, such as scene and object recognition. It is shown that, under some conditions, these images also lie on low-dimensional linear subspaces [32]. High-dimensional data presents a distinct challenge in clustering algorithms. Particularly, traditional similarity measures in clustering use simple distance-based techniques between points to segment data. This, however, is an unsuitable measurement for high dimensional data.

Subspace clustering algorithms are designed to discover clusters in a mixture of high-dimensional vectors drawn from multiple probability distributions. Since data points are unlabeled, we do not know in advance to which subspace they belong. Subspace clustering is defined to simultaneously cluster these data into multiple subspaces and find a low-dimensional subspace approximating all the points in a cluster. The idea is that, when a subset of high dimensional data belongs to a cluster, the points in the cluster lie in a low-dimensional subspace. Therefore, each subspace can be represented by a small set of (learned) bases. Formally, given a set of N points $x_1, ..., x_N \in \mathbb{R}^n$ lying from C subspaces $\mathbb{R}^n$, $\{S_k\}_{k=1}^{C}$ with dimensions $\{d_k \ll n\}_{k=1}^{C}$, subspace clustering algorithms try to divide points into C subspaces and represent points of each subspace with a small set of bases. In this part of the dissertation, we study the improvement of subspace clustering from two angles: (1) Reducing the time and space complexity of subspace clustering using one-time grab sampling; (2) Improving the accuracy of clustering by introducing a joint optimization scheme that incorporates soft and hard clustering techniques in assigning points to clusters.

1.2.1 Scalable Subspace Clustering

A main group of methods in subspace analysis, includes the entire population to recover the low-dimension presentation of each subspace. These methods can find the subspaces with a high accuracy. However, it is computationally expensive when we have a very large dataset. An efficient solution to reduce the time complexity of these algorithms is to start with a subset of points sampled from the dataset [58, 93]. In this thesis, we study the effective of sparse sampling on estimation the coefficient matrix Z and the clustering the data points. Our motivation derives from the fact that subspaces are in low dimension. This assumption indicates that each data point can be represented using a few other points in the same subspace. Thus, a small number of sampled data points can be used to represent the entire population with a high accuracy.

1.2.2 Probabilistic Sparse Subspace Clustering

Early subspace clustering methods stop after dividing points into separate subspaces. The shortcomings of these methods are that the results are final and cannot be improved after clustering is carried out. This deficiency is an incentive to introduce alternating optimization algorithm, where clusters can be improved iteratively. In this part of this dissertation, we introduce an alternating method of subspace clustering. Our method incorporates hard and soft clustering and attempts to include the advantages of both by dividing points into two groups based on an estimated reliability of the cluster assignment for each point x_i, $i = 1, \cdots, N$. Points with high reliability are treated as hard clustered points that can help to get a better representation of subspaces. Low reliable points use a soft clustering approach and have the advantages therein. we delay association of their points with any cluster. At each iteration, subsets of points are associated with clusters, leading to get a better representation of subspaces. Also, it allows a delayed point to be drawn closer to the correct subspace before the final assignments is made. One advantage of this algorithm is that we effectively combine the advantages of both hard and soft clustering, leading to more accurate representation of the points in subspaces, and hence more accurate final results, since certain points are hard-clustered, whereas for uncertain points continuous values are used for re-weighting the representation in the next iteration, similar to soft clustering.

1.3 Dissertation Organization

This dissertation is organized as follows: Chapter 2 studies the related works for sparse sampling and subspace clustering in multi-structured environments. In Chapter 3, we describe our proposed method in one-time grab sparse sampling. In Chapter 4, we introduced our scalable subspace clustering method in handling large scale high-dimensional data points. Chapter 5 proposes a new subspace clustering method using a joint optimization technique. We conclude and discuss future works in Chapter 6

CHAPTER 2: LITERATURE REVIEW

2.1 Sampling in Multiple Structures Estimation

Studies on multi-model estimation can roughly be divided into two groups of iterative based approaches and one-grab sampling based methods. In this section, we review existing methods in both categories and analyze the challenges in each of them.

Iterative Multi-grab Sampling Methods: These methods focus on iteratively sampling subsets of points for finding consensus sets that yield the underlying model instances [66, 102]. Greedy methods such as RANSAC or RANSAC-like methods [24, 81, 90, 99] focus on sequentially detecting structures and estimating their parameters. In order to detect each of the structures, many subsets of ε-*tuples* are sampled randomly until a set consisting of only inliers is determined. Here, ε represents the number of degrees of freedom of the model instances and ε-*tuples* is a subset of ε points. Multi-RANSAC [110], on the other hand, attempts to find all model instances in parallel, but it assumes that the underlying structures do not intersect, which is an impractical limitation for most applications. Iterative multi-grab methods are generally suboptimal for multi-structure data (e.g. when multiple model instances co-exist in data), since the stopping criterion is usually non-trivial, and inaccurate initial fitting can significantly affect the detection of the model instances [95]. These sequential methods also assume the outliers are distributed uniformly, which is violated when multiple model instances are present, forming thus pseudo-outliers that are not distributed uniformly. This issue is discussed in [70], where it is shown that clustered pseudo-outliers are more difficult to handle than uniformly distributed gross outliers. More recent studies [34, 43, 56, 92, 51] attempt to use regularization for better model fitting, while handling multiple intersecting structures. These methods are also iterative. However, instead of using a greedy sequential approach, they resort to a constraint optimization process to find the underlying model instances that can optimally represent the entire data. A major drawback of these methods is that at many iterations one may end up testing unnecessary model hypotheses. Moreover, the method is in part supervised, since the number of model instances is assumed to be known *a priori*, which in practice may not be feasible.

One-grab Sampling Methods: These methods strive to find the best segmentation of the entire data by estimating a set of putative model instances from a sampled subset of data. Starting with a set of putative models, they attempt to determine those that best fit the entire population. In [19] a set of randomly sampled points are used to grow the models that best fit the data. Similarly, the methods in [14, 70] use a subset of points to accelerate the model fitting process, but provide no guaranty as to how big the sample set should be. The method proposed in [109] achieves optimal model selection by measuring a residual error based on histogram distribution of every point in the data for all possible predicted models from a sampled subset of the population. The methods in [79] and [55] also start with a set of initial models derived from randomly selected points and merge them to obtain the best segmentation of the entire data. In a similar manner, the Multi-Bernoulli SAmple Consensus (MBSAC) method proposed in [33] grabs a subset of ε-*tuples* and uses a multi-Bernoulli filtering approach [91] in order to detect all the model instances simultaneously. This method provides some guaranties on the number of required ε-*tuples* to be sampled, and determines the optimal model instances by removing the models with low probabilities. Random Cluster Models (RCM), proposed in [65], includes a conditional random sampling of possible model hypotheses from initially clustered points in a weighted graph. This method along with many others like [91, 84, 100] use some prior knowledge in order to sample points and select the initial hypotheses. One-grab sampling methods handle multi-model fitting problems well and are not affected by the perils of iterative multi-grab sampling. However, they mostly suffer from poor computational efficiency. In order to detect all the models in the data, these methods either include the entire population [3, 71], or sample a subset of population with no guaranty on the optimal sample size, i.e. use heuristics. This often leads to either failing to detect some valid model instances, or taking too many samples leading to too many hypotheses in their search for the optimal segmentation of data. This becomes, in particular, problematic when dealing with "big data" in a high-dimensional space.

To tackle the computational complexity of these approaches, it can be very useful to accelerate the clustering process by sampling a sufficient number of points from the population. Among the studies that also consider the sampling process, the primary goal has been statistical robustness, i.e. maximizing inlier selection for improving the breakdown point. The common thread between most existing methods in terms of the sampling strategy is the idea of sampling based on maximizing some probability of selecting inlier points. This is either achieved by assuming some prior information (e.g. in the form of local neighborhood correlation) [84, 79], or by dividing the sampling set into subsets that are used successively to improve the inlier selections [79, 109]. The problem with this group of methods is that they either need some prior knowledge of the distribution of data or require pre-processing steps in order to obtain some information about the clusters. Few studies attempt to generalize and accelerate the process of fitting models by uniformly selecting points and without any prior process on data. Some Approaches like multi-RANSAC [90] focus on iteratively sampling subsets of ε points (called ε-*tuples*) and find the models to fit most data. In these methods, in order to detect each of the structures, many sets of ε-*tuples* are sampled randomly until a set consisting of only inliers is determined. With selected probability P, the number of ε-*tuples* required to be sampled to get at least one set of all inlier is:

$$r_{\text{RANSAC}} = \frac{log(1-P)}{log(1-\varrho^{\varepsilon})}, \tag{2.1}$$

Where ϱ is the probability of selecting an inlier point, ϱ^{ε} is the probability that all ε points in the sampled subset are inliers and r is the number of ε-*tuples* required to be sampled. This process needs to be repeated for each structure in the population until all the structures are detected and the stopping criterion is met. Basically, these sampling methods model the problem using a multinomial pmf, which implies sampling with replacement. To detect C models in a multi-structure data it is required to sample $C \times r_{\text{RANSAC}}$ points which is too big to be able to reduce the computation time. MBSAC in [33] proposed a sampling technique and use a multi-Bernoulli filtering approach to detect multiple structures simultaneously. The sampling part of this method employs a multinomial distribution model to find the required number of ε-*tuples* to be selected. The idea is that if C is the maximum possible number of structures in the population, then the total number of ε-*tuples* required to detect all the structures will be given by:

$$r_{\text{MBSAC}} = \frac{log(1 - P^{\frac{1}{C}})}{log(1 - \varrho^{\varepsilon})} \tag{2.2}$$

This approach rather than sampling a collection of points in one grab, is characterized by sampling a collection of sampling sets of size ε. The minimum sampling set ε-*tuples* is typically equal to the number of degrees of freedom of each model instance in the population, e.g. for lines in 2D we would sample point pairs. This has the advantage of making the subsequent clustering step a simpler process. Nonetheless, sampling ε-*tuples* imposes limitations on the problem, in the sense that one cannot investigate the cases in which multiple structures with different dimensions exist. Moreover, since they need to make sure that at least one ε-*tuple* consists only of inliers, it is required to sample points with replacement where it makes the sampling strategy to select a point repeatedly.

A natural solution to avoid excessive oversampling and the computational cost is by finding an accurate method of determining the required sample size, which is the focus of this thesis. In particular, we generalize the one-grab sampling, since a tight choice of sample size also allows for sampling without any prior assumptions, while guaranteeing that all model instances can be discovered. The key is to determine the size of the one-grab sampled set, so that all the structures are still represented adequately and can be discovered with high probability, despite a substantial ratio of outliers. This turns out to be the solution to a complex nonlinear equation with no analytic closed form answer. We solve the problem by formulating it in terms of either a multinomial or a hypergeometric pmf and bounding the solution to reduce it to a binary search problem. We impose no constraints on the distribution of points. Thus the samples are taken uniformly, i.e. requiring no prior knowledge of the distribution of data. Moreover, the proposed method can handle structures with different dimensions.

2.2 Subspace Clustering

Several methods are proposed in subspace clustering including algebraic methods [4, 87, 17], iterative methods [6, 83, 108], statistical methods [78, 54, 103], and spectral clustering-based methods [4, 48, 9, 36, 67]. The main drawbacks of the first group of methods are in the handling of noise and outliers, as well as the increase of time complexity with the number of subspaces and their dimensions. Iterative methods detect subspaces sequentially in a repetitive manner. In these methods the number of subspaces and their dimensionalities should be known but this prior knowledge is hardly available in most practical applications. They also suffer from sensitivity to noise and dependency on initialization. Statistical methods approach the problem using the properties of data/noise distribution. The main weakness of these techniques is that the complexity increases quickly with the number of subspaces. Also, they require some prior knowledge about dimensions and number of subspaces. Spectral clustering based (or graph-based) method consist two separate steps of forming a similarity/affinity matrix that describes the pairwise similarity between data points; and obtaining the final segmentation results using graph-based clustering such as spectral clustering (SC) techniques. The main differences of the existing spectral clustering methods lie in the ways of constructing the similarity matrix. These methods use either local information such as the angle between pairwise points [39, 101] or global information such as global optimization approaches [21, 49] to form a similarity matrix.

The main idea of spectral subspace clustering is that to recover the subspace structures from the data each data point can be represented as a linear combination of the bases in a dictionary. Particularly, points in subspaces are self-representative. Thus, when subspaces are independent and noiseless, by having sufficient number of points in each subspace, any point in a subspace can be represented as a linear combination of other points in that subspace. Given a matrix $X \in \mathbb{R}^{n\times N}$, with columns drawn from a union of C independent linear subspaces of $\mathbb{R}^n$, $\{S_k\}_{k=1}^{C}$ with dimensions $\{d_k \ll n\}_{k=1}^{C}$, any data point x_i can be represented as $x_i = X_{s_{\hat{k}}} z_i$, where $x_i \in S_k$, $X_{s_{\hat{k}}}$ are all the data points in S_k except for x_i, and z_i is a coefficient column vector. In a general case, when the population is a union of points from multiple subspaces, x_i is represented as $x_i = X_{\hat{i}} Z_i$ and can be recovered as a sparse solution of the following optimization problem.

$$min||Z_i||_\ell \quad \text{subject to} \quad x_i = X_{\hat{i}} Z_i, \tag{2.3}$$

where $X_{\hat{i}}$ includes all the points in the population of size N except x_i. It is expected that the optimal solution would include non-zero coefficients corresponding to the columns of $X_{\hat{i}}$ that are in the same subspace as x_i and zero for remaining points. To find the sparse representation of all data point $i = 1, .., N$ in a matrix form, the self-representation matrix Z can be obtained by the following optimization.

$$min||Z||_\ell \quad \text{subject to} \quad X = XZ, \tag{2.4}$$

where $Z \in \mathbb{R}^{N \times N}$ is the matrix that column i corresponds to the sparse representation of point x_i. Considering there are outliers and noise in real world data where $X = X_0 + E_0$, X is obtained by corrupting error-free X_0 using E_0 which includes bounded noise or sparse outlier entries. In this case, the optimization problem can be written as follows

$$\min_{Z} \|Z\|_{\ell} + \lambda \|E\|_{\ell'} \quad \text{subject to} \quad E = X - XZ, \tag{2.5}$$

In the literature, the different choices of $\|.\|_{\ell}$ and $\|.\|_{\ell'}$ are studied in which they usually enforce sparsity or rank minimization on the representative coefficients matrix Z by $\|.\|_{\ell}$ [21, 49, 53] and enforce error/noise minimization in $\|.\|_{\ell'}$ [75, 94]. Give the recovered coefficient matrix Z, pairwise similarities of points can be computed using equation (2.6) which is a symmetric and non-negative matrix.

$$\bar{Z} = \frac{1}{2}(|Z| + |Z^T|). \tag{2.6}$$

A graph-based clustering method, can then be applied to the similarity matrix to find the clusters. Graph-based clustering such as spectral clustering characterizes data points as nodes in a weighted graph. Pairwise dissimilarity weighs the edges between nodes. The key idea is to partition the nodes in to several sets with the minimum sum of edge weights between each set. Clustering algorithm such as Normalized-cut [73] gets number of subspaces C as the input to the algorithm. A modified version of this algorithm in proposed in [57] in which it handles an arbitrary number of clusters. However, the performance of these clustering methods depend highly on the accuracy of similarity matrix, and that is the motivation for a lot of studies in this area.

One limitation of classical subspace clustering algorithms is that the self-representation matrix Z is learned from shallow data that may not capture complex hidden structures. Another issue is that they heavily rely on linearity assumptions which may not always be true. In fact, in a lot of applications, data can be modeled more accurately by nonlinear manifolds. Kernel-base methods, such as [31, 60, 97, 107] attempt to address this deficiency. However, the performance of these methods highly depends on the choice of the kernel function. Very recent studies explore the idea of using neural networks [45, 98, 29, 61, 106, 41] to reduce the data dimension and to improve the handling of nonlinearity. Auto-encoder neural networks are unsupervised learning algorithms consisting two parts. The encoder attempts to map the input data to a compact representation. The decoder uses this representation to accurately reconstruct the data. In the context of computer vision, auto-encoders are used in representation learning of unlabeled data [89, 68, 1]. These approaches use deep learning as a preprocessing step to learn the low-dimensional latent space representation of data before segmenting them into subspaces. For clustering purposes, the learned low-dimensional representation of points must be tied with the clustering process to derive an accurate clustering result [18, 63, 105]. The method in [63] uses auto-encoders to find the representation of data and clusters the latent-space features using k-means. However, they search for the self-representation matrix Z using the entire population in the ambient dimension n which can be very time consuming and may include a high ratio of noise as it depends on the linearity of input data. The method in [61] tried to remove this dependency by defining initial cluster kernels where they learn both data representation and modify the cluster centers jointly. However, the choice of the kernels can still be problematic. In general, these methods suffer from two main disadvantages: First, they go through a more complex and time consuming process; second, they need large training sets to learn network parameters. Making the training unsupervised (label-free) would result in substantial loss of accuracy.

2.2.1 Scalable Subspace Clustering

The key idea in subspace clustering is to find an effective self-representation of points in similarity matrix $\bar{Z}$. One issue with these method is the computational complexity of the process in dealing with large-scale datasets. Methods in the literature have been tried to improve the performance of subspace clustering, by enforcing different constraints on the coefficients [52], or proposing sampling/rescaling techniques [50, 100, 77]. Sampling is a natural method to rescale the data and make the process more efficient. A group of methods in the literature use some pre-processing techniques to reduce the data size. Proposed method in [100] uses k-means clustering to approximate cluster centers as an initial step and prior to subspace clustering process. In a similar manner [74] proposed a bottom-up approach to assign points to cluster centers that are estimated in a pre-processing step. Authors in [11] proposed Landmark-based Spectral Clustering (LSC) algorithm where a set of landmarks (bases) are selected using k-means clustering and the rest of the points are assigned to the landmarks by a pairwise distance measurement. In [64] the authors propose to cluster a small sampled subset of the original data and then classify the rest of the data based on the learned groups. In this approach the subset of points can be sampled uniformly at random or by other techniques such as k-means clustering. These methods attempt to reduce the computational complexity of subspace clustering by providing sampling techniques but provide no guarantee as to how big the sample set should be.

The basic idea in [64, 62] is that by having enough sample points from each subspace S_k any point in the subspace (even out-of-sampled data) can be represented as the linear combination of sampled points in that subspace. In another word, having a dataset $X \in \mathbb{R}^{n\times N}$ with columns drawn from a union of C independent linear subspaces, if we sample enough columns in a subset $\chi \in \mathbb{R}^{n\times\aleph}$ with $\aleph$ points, then we can find the coefficient matrix Z for in-sample points χ using optimization (2.5) and form the clusters. Later out-of-sample points χ^* projects over χ using the following equation:

$$Z^* = \left(\chi\chi^T + \beta\mathbf{I}\right).\chi^T\chi^*, \quad \text{where} \quad \chi \cap \chi^* = \varnothing, \tag{2.7}$$

where β is the rigid regression parameter $\beta = 1e - 6$. Equation (2.7) assign each out-of-sample points $x_i^* \in \chi^*$ to the best subspace S_k by finding the minimum normalized residual $\arg\min_k r_k(x_i^*)$ for all subspaces $k = 1, .., C$, using the following formula:

$$r_k(x_i^*) = \frac{\|x_i^* - \chi\xi_j(Z_i^*)\|_2}{\|\xi_j(Z_i^*)\|_2}, \tag{2.8}$$

where $\xi_k(Z_i^*)$ is found by keeping all the elements of the vector $\bar{Z}_i^*$ that are associated with subspace S_k, and setting the remaining elements to zero. This method, similar to other methods mentioned earlier, cluster points more efficiently. However, they either require some pre-processing steps [100, 74] or provide no theoretical guaranty on the sample size [64]. The first group can slow down the process or establish an irreversible initialization. The second group can fail to detect all the correct subspace or be unsuccessful to reduce the computational complexity enough.

In this dissertation, we study the effect of one-time grab sampling strategy on finding the required number of point in the selected subset in order to detect all the subspaces with a high accuracy and efficiency.

2.2.2 *Probabilistic Sparse Subspace Clustering*

Early spectral subspace clustering methods include two separate steps of computing a similarity matrix and finding clusters using a spectral clustering algorithm. In this process, the similarity matrix discloses the connectivity between points. In the clustering step, points are divided into separate groups using the given similarity. The shortcomings of these methods are that the results are final and cannot be improved after clustering is carried out. This deficiency is an incentive to combine the two steps into an alternating optimization algorithm, where both the similarity matrix and clusters can be improved. The authors of [28, 23, 47] developed unified iterative frameworks for updating the low-rank matrix Z using clustering results and subsequently finding the clusters using this new version of Z. The idea is that both sparse similarity matrix and clusters depend on each other. Thus, an alternating method can be used in the spectral clustering step to remove noise from the similarity matrix, resulting in a more accurate estimator of the similarity matrix. This leads to more accurate clusters in the spectral clustering step. The method proposed in [47] defines an additional matrix $Q \in \{0,1\}^{N \times C}$, as the clustering matrix, such that $q_{ij} = 1$ if vector i belongs to cluster j, and $q_{ij} = 0$ otherwise. This method uses an approach similar to (2.5) and defines an objective function as follows:

$$\min_{Z,Q} \left(\|Z\|_{1,Q} + \lambda \|E\|_{\ell'} \right) \quad \text{subject to} \quad E = X - XZ, \quad \text{diag}(Z) = 0, \tag{2.9}$$

where $\|Z\|_{\ell}$ in equation 2.5 is replaced with $\|Z\|_{1,Q}$ and depends on both the sparsity of Z and the clustering matrix Q obtained in the previous step. This term is defined as follows:

$$\|Z\|_{1,Q} = \|Z\|_1 + \alpha \|Z\|_Q \quad \text{where} \quad \|Z\|_Q = \sum_{i,j} |Z_{i,j}| \left(\frac{1}{2} \left\| q^{(i)} - q^{(j)} \right\|^2 \right), \tag{2.10}$$

where $\alpha > 0$ and $q^{(i)}$ and $q^{(j)}$ are rows i and j of matrix Q. The two matrices Q and Z are related. If the elements $z_{i,j} \neq 0$ or $z_{j,i} \neq 0$ in Z are relatively large, then it is likely that both points i and

j are from the same subspace and, therefore, rows i and j of matrix Q should be identical. One would expect that $q^{(i)} = q^{(j)}$ whenever $\bar{Z}_{ij}$ is large and vice versa. To find the two matrices of Z and Q, authors proposed an iterative optimization algorithm that alternates between solving for (Z, E) given the clustering matrix Q using convex optimization, and solving for Q given (Z, E), using spectral clustering. In other words, when Q is known, equation (2.9) reduces to the problem of finding (Z, E):

$$\min_{Z,E} \|Z\|_{1,Q} + \lambda \|E\|_{\ell'} . \tag{2.11}$$

Note that the norm $\|Z\|_{1,Q}$ places higher penalty on values of $|Z_{i,j}|$ if vectors x_i and x_j are in the different clusters. Similarly, given (Z, E), matrix Q is the solution of the following optimization problem:

$$\min_{Q} . \quad \|Z\|_{1,Q} \quad Q \in \mathcal{Q} \tag{2.12}$$

where $\mathcal{Q} = \{Q \in \{0, 1\} : Q\mathbf{1} = \mathbf{1} \quad and \quad \text{rank}(Q) = C\}$ [47] means that in each row of Q, only one element is equal to one and all others are equal to $zero$ and $\text{rank}(Q) = C$ where C is the number of subspaces in the population. Since optimization for formula (2.12) is a combinatorial optimization, in the literature, graph-based clustering approaches (e.g. spectral clustering) is used to find the binary segmentation matrix Q.

The problem of this approach is that element of matrix Q are either zero or one. That can remove useful information in Z, e.g. strong coefficients z_{ij} when i and j are clustered in two separate subspaces using k-means. An extended version of this approach (called soft S^{3}C) is proposed in [47] where Q includes continuous real values obtained by keeping eigenvectors associated with C smallest eigenvalues of the computed Laplacian matrix. Soft clustering has the advantage of including continuous real-values for re-weighting the representation matrix in the next iteration. This is useful in capturing more information from previous iterations, especially when there are ambiguities. However, soft clustering has the disadvantage of removing less noise from the similarity matrix compared to hard clustering. Some recent studies explore the idea of using neural networks [61] to reduce the data dimension and to improve handling of nonlinearity. These methods suffer from two main disadvantages: First, they go through a more complex and time consuming process; second, they need large training sets to learn network parameters. Making the training unsupervised (label-free) would result in substantial loss of accuracy.

In this dissertation, we introduce an alternative method of subspace clustering. We estimate the reliability of the cluster assignment for each point x_i, $i = 1, \cdots, N$. If the reliability is low, we call the point "*uncertain*", and delay its association with any cluster. Remaining points at that iteration are considered "*certain*", and clustered right away. This helps improve the accuracy of updating the elements of the similarity matrix Z in the next iteration. At each iteration, subsets of points are associated with clusters, leading to a accurate self-representation matrix Z. Also, it allows a delayed point to be drawn closer to the correct subspace before the final assignments is made. Two main advantages are: (i) We effectively combine the advantages of both hard and soft clustering, leading to more accurate similarity matrix Z and more accurate final results, since certain points are hard-clustered, whereas for uncertain points, continuous values are used for re-weighting the representation matrix in the next iteration. (ii) This process of delayed association lends itself to the possibility of using an incremental spectral clustering, which in turn leads to a huge reduction in complexity and computational time.

2.3 Summary

In this chapter, we briefly presented commonly used approaches for big data analysis in computer vision and pattern recognition applications using sparse sampling and dimension reduction. We first reviewed recent approaches in structure detection in multi-structural environments. These methods detect data structures and their segmentation in high volumes of data by sampling a subset of points. Furthermore, we reported recently introduced methods in subspace clustering in which they reduce the dimension of the data in multi-structural environments and cluster them. We studied various algorithms in this area, including their strengths and weaknesses. In the next three chapters, we describe our proposed methods of reducing high volume and high-dimensionality of big data and their application in computer vision.

CHAPTER 3: SPARSE ONE GRAB SAMPLING WITH GUARANTEES

3.1 Method Overview

In this chapter, we propose a solution to the problem of estimating multiple structures in a large population of data points by sampling a sparse subset of the data in one grab. The key is to determine the size of the one-grab sampled set, so that all the structures are still represented adequately and can be discovered with high probability, despite a substantial ratio of outliers. This turns out to be the solution to a complex non-linear equation with no analytic closed form answer. We solve the problem by formulating it in terms of either a multinomial or a hypergeometric pmf and bounding the solution to reduce it to a binary search problem. We impose no constraints on the distribution of points. Thus, the samples are taken uniformly, i.e. requiring no prior knowledge of the distribution of data. Moreover, the proposed method can handle structures with different dimensions.

Sparse Withdrawal of Inliers in a First Trial (SWIFT) [35] is a one-grab sampling method that we propose in order to select a random subset of data with no prior assumptions about the distribution of points, and with the guarantee that the estimated optimal sample size is both tightly bounded and statistically sufficient to determine all the underlying model instances in the data. Taking a one-grab sample of r points from a population of size N may be viewed as sampling r points one point at a time without replacement. As is well known, the probability of whether a randomly drawn point has some specified feature is determined by the hypergeometric pmf [37, 69]. Also, if the population size N is much larger than r, then this probability is closely approximated by a multinomial pmf [37, 69]. Intuitively, we can see why this is true, because multinomial pmf models sampling with replacement, and when $N \gg r$, the chance of drawing the same sample point after replacement would be extremely negligible, i.e. multinomial would asymptotically approach the hypergeometric. We will show that, for most practical applications, the condition $N \gg r$ is readily satisfied, i.e. our tight bounds of the estimated sample size yields the desired sparsity. This is in contrast to other methods described earlier that rely on heuristics. Sampling with replacement modeled by multinomial pmf is easier to handle mathematically, due to the independence of events leading to simpler approximations. In this chapter, we study both the hypergeometric model, which is the true mathematical model for our problem, and the multinomial model that closely approximates our problem. We also analyze and compare the accuracy of both models in section 3.6.

3.2 Definitions and Notations

We consider the situation where a population of N points is comprised of C classes with sizes $\theta_1, ..., \theta_C$ where $\sum_i^C \theta_i = N$. In a real situation neither the number of classes nor their sizes are known, so we propose a sampling algorithm that does not assume this knowledge. In particular, our algorithm requires three input parameters: the minimum size of a sample set per structure ε, the minimum model size θ, and the probability $1 - \delta$ of grabbing at least ε points from each structure. The value ε here is no smaller than the number of degrees of freedom of each model instance, e.g., $\varepsilon = 2$ for a line, and $\varepsilon = 3$ for a circle. The minimum model size, θ, is the minimum number of inlier points necessary to accept a candidate model, so that if the class size is smaller than θ, then we are not interested in extracting it. The sampling is then carried out by grabbing r points at random from the population without replacement. Then, there are d_i points from each of C classes and $\sum_i^C d_i = r$. The novelty of our approach is that we provide the value of $r = r(\varepsilon, \theta, \delta)$ such that $d_i \geq \varepsilon$ for each of the classes $i = 1, \cdots, C$.

To derive the SWIFT sampling scheme, we start by assuming the worst-case scenario, where all model instances in the population are presumed to be of size θ. This implies that all outliers are pseudo-outliers and that the maximum number of possible model instances C that can be potentially present in the population is given by

$$C = \left\lceil \frac{N}{\theta} \right\rceil . \tag{3.1}$$

Where $\lceil\rceil$ rounds the fraction to the nearest upper integer, and N is the population size. It is well known that with this one-time grab, the vector $(d_1, \cdots, d_C)$ follows multivariate hypergeometric distribution, so derivation of SWIFT in this case does not require any additional assumptions.

3.3 Modeling SWIFT by Hypergeometric Distribution

We have a population of N points comprising of underlying C model instances. Suppose now we select a subset of r points sampled at random in a one-time-grab, with d_i points coming from the i^{th} instance. Then, the pmf of the vector $(d_1, \cdots, d_C)$ follows the multivariate hypergeometric distribution (sampling without replacement) [37, 69]:

$$P(d_1 = x_1, \cdots, d_C = x_C) = \frac{\prod_{i=1}^{C} \binom{\theta_i}{x_i}}{\binom{N}{r}}, \tag{3.2}$$

where N is the population size, C is the number of classes and $\theta_1, \ldots, \theta_C$ are their respective sizes, with $\sum_{i=1}^{C} \theta_i = N$, $\sum_{i=1}^{C} x_i = r$ and $0 \le x_i \le \theta_i,\ i = 1, \ldots, C$. Equation (3.2) expresses the probability of a given sample set in terms of r. However, our goal in SWIFT sampling is to solve the inverse problem of finding r for a given probability. For this purpose, we recall that we are dealing with the symmetric case of $\theta_1 = \theta_2 = \cdots = \theta_C = \theta$ and the worst case scenario, where for a given N, the maximum possible classes is $C = \frac{N}{\theta}$. Therefore, $N = C\theta$ and equation (3.2) becomes:

$$P(d_1 = x_1, \cdots, d_C = x_C) = \frac{\prod_{i=1}^{C} \binom{\theta}{x_i}}{\binom{C\theta}{r}}. \tag{3.3}$$

The objective of the method can then be expressed as follows: Find r such that, for a given value $\delta > 0$, the probability of selecting at least ε points in each of C model instances is at least $1 - \delta$, that is

$$P\left(\cap_{i=1}^{C}(d_i \geq \varepsilon)\right) \geq 1 - \delta \quad \text{provided} \quad \sum_{i=1}^{C} d_i = r \tag{3.4}$$

The solution to the above problem can be determined by finding the lower and upper bounds for the tail probabilities of the multivariate hypergeometric pmf (see, e.g. [13, 8, 42]) and use them for derivation of the value of r.

3.3.1 Lower Bound

We find a lower bound for the probability in the left-hand side of the inequality (3.4) and use it for finding the lower bound for r. Note that $P(d_i \geq \varepsilon) = 1 - P(d_i \leq \varepsilon - 1)$ and, therefore,

$$P\left[\cap_{i=1}^{C}(d_i \geq \varepsilon)\right] = 1 - P\left[\cup_{i=1}^{C}(d_i \leq \varepsilon - 1)\right] \geq 1 - \sum_{i=1}^{C} P(d_i \leq \varepsilon - 1) \tag{3.5}$$

Using the above inequality, we can prove the following theorem:

Theorem 1 *Let $(C-1)\theta \geq r$. Then, one has*

$$P(\cap_{i=1}^{C}(d_i \geq \varepsilon)) \geq 1 - \sum_{i=1}^{C} P(d_i \leq \varepsilon - 1) = 1 - C\Delta, \tag{3.6}$$

where

$$\Delta = P(d_1 \leq \varepsilon - 1) \leq P(d_1 = 0) \sum_{k=0}^{\varepsilon-1} \binom{r}{k} \left(\frac{\theta}{N - r - \theta + k}\right)^k, \tag{3.7}$$

and

$$P(d_1 = 0) = \prod_{j=0}^{\theta-1} \frac{N-r-j}{N-j} \leq \left(1 - \frac{r}{N}\right)^{\theta} \leq e^{\frac{-r}{C}}. \tag{3.8}$$

The proof of this and later theorems are placed in the appendix.

Setting $C\Delta = \delta$ and solving (3.7) for r with $\Delta = \frac{\delta}{C}$, we obtain the necessary lower bound on the SWIFT sampling size. The derived inequality for $\Delta(r)$ in (3.7) is a non-increasing function on r. This sets the statement in equation (3.6) as a non-decreasing function of r when C is reasonably small. Given N and $(\varepsilon, \theta, \delta)$, we can simply find r by using a binary search through all possible values of r between $C \times \varepsilon$ and N. For $P(d_1 = 0)$, one can either use the exact formula or its upper bound in equation (3.8). In our numerical studies, we used the exact expression for better accuracy. Algorithm 1 shows the detailed steps to find the sample size r. Note that, since we are using the lower bounds for the multivariate hypergeometric pmf, Theorem 1 provides the bound on the sample size r. In order to show that this lower bound is tight, in the next section, we study also the upper bound for r.

Algorithm 1 Estimating sampling size r by hypergeometric pmf

```
Input: $N$ and $(\varepsilon, \theta, \delta)$
 1: Set $C := \lceil \frac{N}{\theta} \rceil$
       $r_{\text{Min}} := C \times \varepsilon$
       $r_{\text{Max}} := N$
       $\Delta := 0$
    Do Binary search  {since $\Delta(r)$ is non-increasing on $r$ }
 2: while $r_{\text{Min}} < r_{\text{Max}}$ do
 3:     $r := \frac{1}{2}(r_{\text{Min}} + r_{\text{Max}})$
 4:     Find $\Delta$ using inequities (3.6)-(3.8)
 5:     if $C\Delta > \delta$ then
 6:         $r_{\text{Min}} := r$
 7:     else
 8:         $r_{\text{Max}} := r$
 9:     end if
10: end while
11: return $r$
```

3.3.2 Upper Bound

In this section, we derive an upper bound for the tail probabilities of the hypergeometric pmf, which will set an upper bound on the sample size r for SWIFT. In particular, the following statement is true:

Theorem 2 *If $(C - 2)\theta \geq r$, then*

$$P(\cap_{i=1}^{C}(d_i \geq \varepsilon)) \leq 1 - C\Delta + \frac{C(C-1)}{2}\Delta', \tag{3.9}$$

where Δ *is defined in* (3.7)*,*

$$\Delta' = P\left[(d_1 \leq \varepsilon - 1) \cap (d_2 \leq \varepsilon - 1)\right] \leq P_0 \times \sum_{k=0}^{\varepsilon-1} \sum_{l=0}^{\varepsilon-1} \binom{r}{k,\ l} \left[\frac{\theta}{(C-2)\theta - r}\right]^{k+l}, \qquad (3.10)$$

and

$$P_0 = P(d_1 = 0, d_2 = 0) \leq \left(1 - \frac{r}{C\theta}\right)^{2\theta} \leq e^{\frac{-2r}{C}}. \qquad (3.11)$$

By combining (3.6) and (3.9), the probability of choosing at least ε points in each model instances can be bounded above and below as:

$$1 - C\Delta \leq P(\cap_{k=1}^{C}(d_i \geq \varepsilon)) \leq 1 - C\Delta + \frac{C(C-1)}{2}\Delta'. \qquad (3.12)$$

In section 3.6, we demonstrate that the bounds derived above are indeed very tight, providing accurate estimates for r. The described estimation for the upper-bound is used to show the accuracy and tightness of estimated sample size r with respect to the ground-truth.

3.4 Modeling SWIFT by Multinomial Distribution

Although SWIFT is a "one-grab" sampling method, and hence is exactly modeled by the multivariate hypergeometric pmf, the estimation of r requires solution of nonlinear inequalities, which can be time consuming when both N and C are large. However, as mentioned earlier, when $N \gg r$ the hypergeometric pmf can be accurately approximated by the multinomial pmf, which basically implies that sampling with replacement approaches the sampling without replacement when $N \gg r$.

Again, this is due to the fact that for $N \gg r$, the probability of grabbing any sample point more than once becomes extremely negligible. Since choosing r as small as possible is one of the goals of SWIFT, the assumption of $N \gg r$ is justified. This motivates us in this section to study the approximation of the tail probabilities based also on a multinomial pmf. An important outcome of the study in this section is that it eliminates the need for searching for r when $N \gg r$ (see Algorithm 2).

Suppose a set of r points is selected at random with replacement from a population of $N \gg r$ points comprised of C classes with sizes $\theta_1, ..., \theta_C$, so that $\sum_i^C \theta_i = N$. Then, the pmf of d_i is given by:

$$P(d_i = x_i) = \binom{r}{x_i} p_i^{x_i} (1 - p_i)^{r - x_i}, \tag{3.13}$$

where $\sum_i^C x_i = r$, $0 \leq x_i \leq \theta_i$ and $p_i = \frac{\theta_i}{N}$. Using equation ((3.5)) and recalling that $C\theta = N$, $\theta_i = \theta$ and $p_i = \frac{1}{C}$ for $i = 1, \ldots, C$, we obtain that the inequality ((3.6)) still holds in this case, but with a different value of Δ:

$$\Delta = P(d_1 \leq \varepsilon - 1) = \sum_{k=0}^{\varepsilon-1} \binom{r}{k} \frac{(C-1)^{r-k}}{C^r} = Bin\left(r, \frac{1}{C}\right), \tag{3.14}$$

where $\sum_{i=1}^C d_i = r \geq \varepsilon \times C$ and $Bin(r, \frac{1}{C})$ is the binomial pmf. Note that r is the only unknown variable in Equation (3.14) and that the right-hand side of this equation is a non-decreasing function of r when C is relatively small. Thus, similar to section 3.3.1, one can find the value of r by using a binary search through the possible values. Moreover, by applying Bernstein inequality for the tail probability of the binomial distribution: for any $t > 0$

$$P\left(Bin\left(r, \frac{1}{C}\right) < \frac{r}{C} - t\right) \leq \exp\left(-\frac{t^2C^2}{2r(C-1) + 4C^2t/3}\right). \tag{3.15}$$

We can find an explicit lower bound on r. In particular, the following statements hold.

Theorem 3 *Let N be large, so that the multinomial approximation of the hypergeometric distribution holds. If*

$$r \geq \frac{14}{3} C \ln\left(\frac{C}{\delta}\right) + 2C(\varepsilon - 1), \tag{3.16}$$

then

$$P(\cap_{i=1}^{C}(d_i \geq \varepsilon)) \geq 1 - \delta. \tag{3.17}$$

Proposition 1 *Based on the Central Limit Theorem [7], if the total sample size r is relatively large, (say, r is an order of magnitude bigger than C), then the binomial distribution for the sample size d_i in the i^{th} model instance can be approximated by the normal distribution: $Bin(r, \frac{1}{C}) \approx \mathcal{N}\left(\frac{r}{C}, \frac{r(C-1)}{C^2}\right)$. Thus, the sampling size r can be obtained by:*

$$r \geq \left(A^2(C-1) + 2C(\varepsilon - 1)\right), \tag{3.18}$$

where A is:

$$A \equiv \max\left(1, \sqrt{2\ln\left(\frac{C}{\sqrt{2\pi}\delta}\right)}\right). \tag{3.19}$$

Based on what is described in this section, the multinomial distribution estimation can be used to calculate a lower-bound for r when the condition in inequity (3.18) is satisfied. Otherwise, one needs to use equation (3.14) or use the hypergeometric estimation of r. In section (3.6), we will evaluate the tightness of the inequality (3.18). Algorithm 2 shows the steps for finding r using the multinomial model.

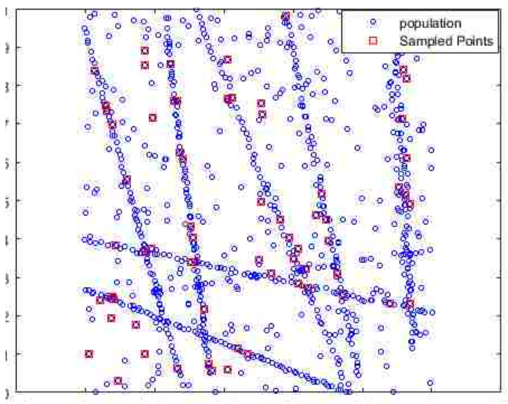
Algorithm 2 Estimating sampling size r by multinomial pmf

```
Input: $N$ and $(\varepsilon, \theta, \delta)$
1: Set $r$ according to formula (3.16) or (3.18)
2: if $N \gg r$ then
3:     return $r$
4: else
5:     Find $r$ using algorithm1 using formula (3.14) instead of (3.6)-(3.8)
6:     return $r$
7: end if
```

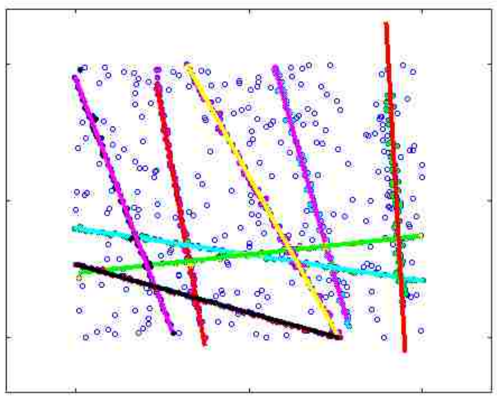
3.5 Clustering and Parameter Estimation

Once a SWIFT subset of a population of points is selected, the sampled points are used to estimate the model parameters. We can then detect the valid model instances in the population, where valid means a model size larger than θ. Different clustering method may be used as the back-end to our sampling step, e.g. [15, 79, 14, 34]. In our experiments below, we used mean-shift [12, 16] (also used later in section 3.7), which is a non-parametric unsupervised clustering method that does not require a prior knowledge of the number of clusters nor any constraints on the distribution of the clusters.

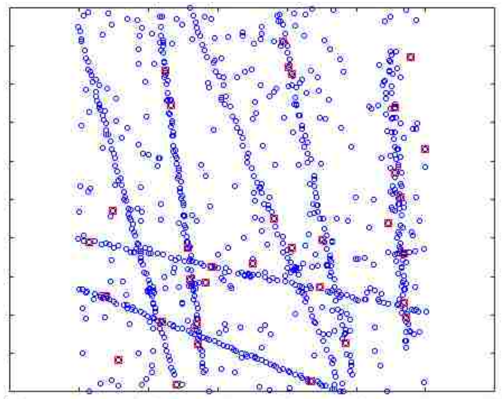
To demonstrate immediately how SWIFT is useful in guaranteeing an accurate estimate of the required sample size in order to find all model instances from a sparse subset of data, we run a test on a simple synthetic data. In this experiment, we show that an accurate sample size r can yield sufficient number of points to detect all the model instances. On the other hand, one is likely to fail in finding all the model instances, when we do not have a guaranty on sample size. This example includes 2D lines in a population size of 850 points and is illustrated in Figure 3.1. The points form 8 noisy lines in 2D, crossing randomly in the presence of 50% gross outliers. In Figure 3.1a we used SWIFT to calculate the sample size r. The selected sample size is $r = 66$ which is computed with input parameters $r(\varepsilon = 2, \theta = 80, 1 - \delta = 0.95)$. Using this sample size, r points are uniformly sampled in a one-time grab, as shown with small red boxes in Figure 3.1a. This subset generates 2145 hypotheses without any preprocessing for finding neighboring points (as in [14]) nor having any prior knowledge regarding the distribution of the data. Figure 3.1b shows the clusters formed from the sampled subset and the subsequently detected inliers. As shown, a sufficient number of points are selected and all the models (lines) are detected accurately. On the other hand, Figures 3.1c and 3.1d show how the same process for detecting model instances can fail, when the sample size is insufficient. In this example, we used the same data as in the previous experiment. The sample subset is 33, which is selected uniformly at random and generates 528 hypotheses. As illustrated, two out of eight model instances are not detected.



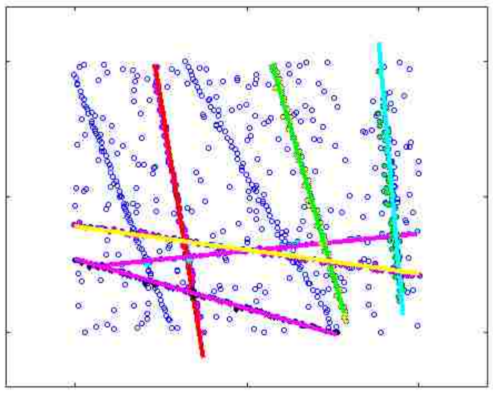
(a) Points and the sampled subset



(b) Detected Lines



(c) Points and the sampled subset



(d) Detected Lines

Figure 3.1: The population includes 8 lines and 50% gross outliers. (a): shows the data points and sampled subset when the size of the sample set is calculated using SWIFT. (b) shows the detected lines using SWIFT samples and mean-shift. (c) and (d) show the underestimated sample size and the detected lines, failing to find all the instances in the data.

3.6 Experimental Evaluation

In this section, we evaluate the proposed sparse one-grab sampling method, and investigate the effect of different input parameters.

3.6.1 Accuracy of SWIFT sample size

As mentioned earlier, due to the non-decreasing property of the derived equations in section 3.1, the SWIFT sample size r can be estimated by a simple search, with the time complexity of $O(log(N))$. The success of the proposed SWIFT sampling highly depends on the accuracy of the estimated values for input parameters. In essence, by following the worst-case scenario, we are treating the problem as if there were no gross outliers in the population. Moreover, the parameter θ is chosen to be equal to the smallest possible size for a valid model instance. These two assumptions, plus the fact that the value of the probability P is in practice chosen as high, ensure that the computed sample size r can guarantee with high probability at least ε points on every structure. Below, we experimentally evaluate the accuracy of the estimated sample size r and the derived bounds in (3.6), (3.12) and (3.14). For this purpose, we used a statistical simulation of sampling without replacement to generate a "ground-truth" data set for different sample sizes with the computed probabilities. The ground-truth simulation was averaged over 1000 independent trials.

Accuracy of Estimated Sample Size: In sections 3.3 and 3.4, we introduced two different solutions for the lower bound of the sample size r, based on hypergeometric and multinomial pmf models. The defined $\delta(r)$ in both inequity (3.7) and the equality (3.14) are non-increasing with respect to r. Using binary search through all possible values of r, one can find the best sample size r. We validated the accuracy of these estimates against ground-truth. For this purpose, we chose different population sizes with different embedded model instances. The ground-truth and estimated values of r given by equations (3.7) and (3.14) are plotted as a function of the probability $1-\delta$ in Figure 3.2. These plots present the average values of r over 200 independent trials for population sizes of $N \in \{10^3, 10^4, 10^5\}$. As expected, our approximations of the lower bound estimation of hypergeometric distribution and multinomial distribution follow the ground-truth closely. The multinomial distribution can overestimate the sample size when the number of classes grow.

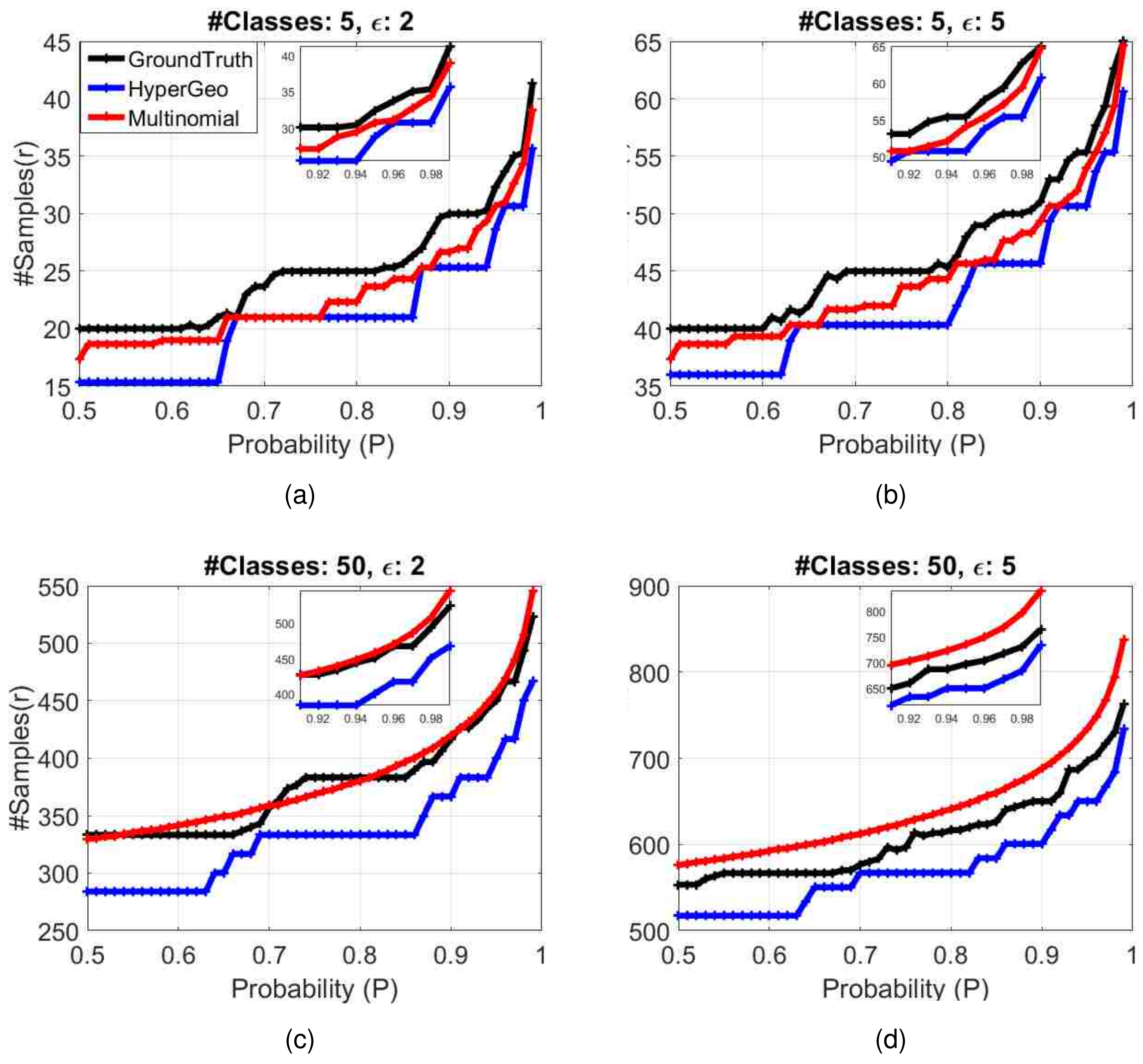
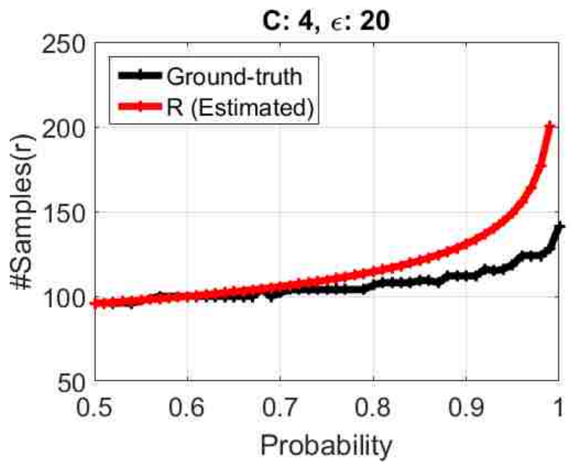
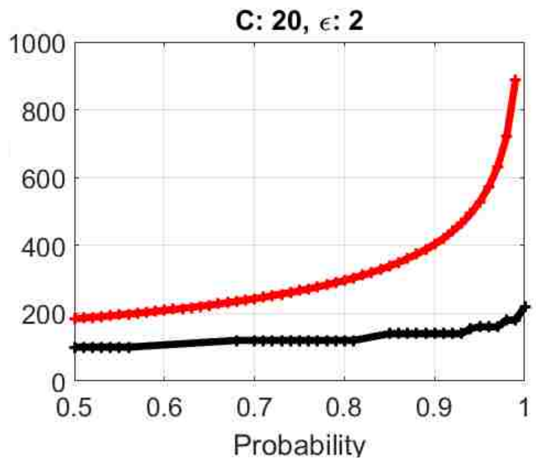


Figure 3.2: Comparison of estimated r averaged over 200 independent trials versus ground-truth of r when $N \in \{10^3, 10^4, 10^5\}$, $\varepsilon \in \{2, 5\}$ and $C \in \{5, 50\}$.

In addition to the estimated upper/lower bounds, in section 3.4, we introduced an approximation for sample size r under the condition of $r \gg C$. In this part, we examine the accuracy of the estimate sample size in (3.18). In Figure 3.3, the result of the estimated values of r given by (3.18) and the ground-truth are plotted against different desired probability values $1-\delta$ and when $N \in \{10^3, 10^4, 10^5\}$. In Figure 3.3a, the computed value of r satisfies the condition of $r \gg C$. Thus, the estimated value r (using (3.18)) closely follows the ground-truth in most parts and overestimates the sample size when probability $1-\delta$ is close to one. However, the experiment in Figure 3.3b violated the condition of $r \gg C$. Therefore, the estimated sample size r using (3.18) overestimated the sample size r and it is not close enough to the ground-truth.



(a) $r \gg C$



(b) $r \not\gg C$

Figure 3.3: Examine the average accuracy of estimated r using multinomial estimation in (3.18). Here $N = 10^5$ and from left to right $\varepsilon = \{20, 2\}$ and $C \in \{4, 20\}$. The condition of $r \gg C$ is satisfied in (a) and violated in (b)

Tightness of Upper/Lower Bounds: We evaluated the tightness of both upper and lower bounds given by the inequalities in (3.6) and (3.9). For this purpose, we examined different population sizes with different numbers of model instances. The results are presented in Figure 3.4. These graphs illustrate the average results over 200 independent trials computed for population sizes of $N = \{10^2, 10^3, 10^4\}$. Results demonstrate that both the lower bound and the upper bound approximations tightly follow the ground-truth of $1 - \delta$. Of course, a conservative estimate of r would be given by (3.6). However, the proof of tightness of these bounds indicates that we would not drastically overestimate r using (3.6).

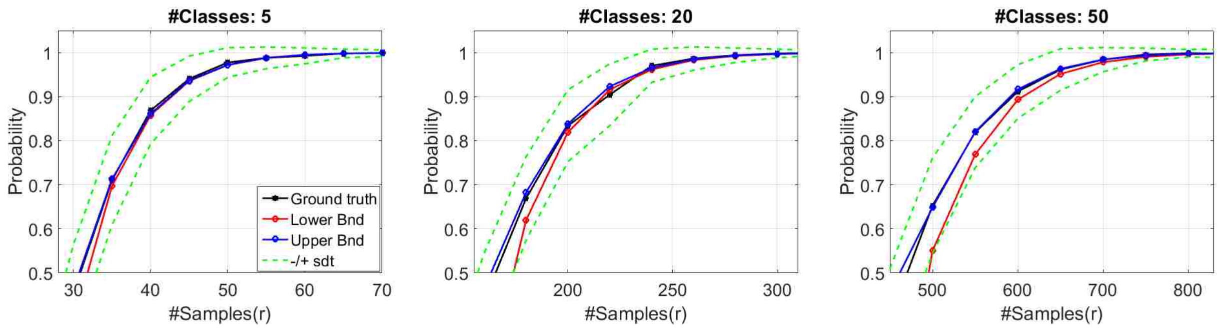


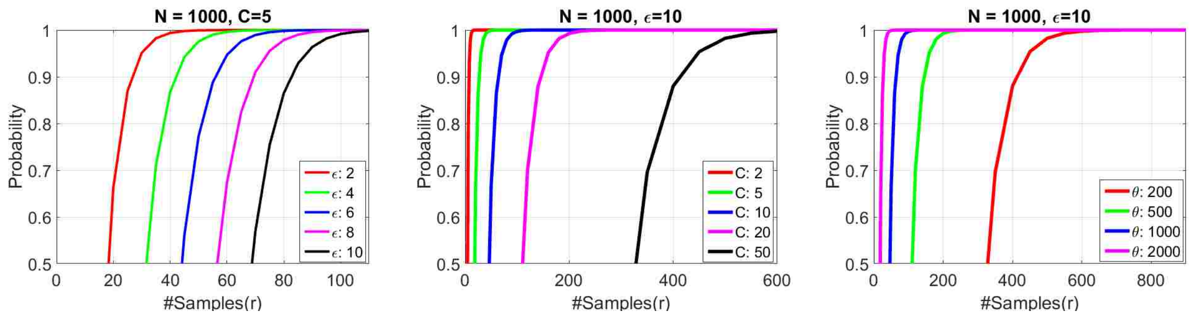
Figure 3.4: Estimated values of lower bound and upper bound of r for $1 - \delta$ averaged over 200 independent trials versus the ground-truth when $N = 10^4$ and $\varepsilon = 4$. From left to right: $C \in \{5, 20, 50\}$.

3.6.2 Role of Different Parameters

In this section, we investigate the effect of changing each of the parameters in (3.6) on estimating the value of r.

The Role of ε: One of the parameters in estimating r is the minimum sample set ε per structure to be withdrawn by SWIFT. If the population size N is fixed, then the growth in ε leads to an increase in the number of required samples r for a given probability $1 - \delta$. This behavior is illustrated in Figure 3.5a in which the population size, N, and the number of model instances C are kept constant. However, we can see that the growth in r as ε increases is independent of probability δ.

Number of Model Instances C and Model Size θ: In a constant population size N, increasing the number of model instances forces the method to grab more points in order to guarantee, with probability $1 - \delta$, minimum ε points on each model instance. This behavior is shown in Figure 3.5b. Note that the relation between θ and C is defined based on equation (3.1). When N is constant, increasing C is equivalent to decreasing the value of θ. Therefore, a similar behavior is observed for θ in Figure 3.5c. A very important observation is that, when the number of classes C is fixed, increasing the population size N does not affect the number of required sampled points r, i.e. *the level of sparseness is independent of the population size.* In other words, r remains small regardless of the population size N, which is a desirable sparseness property and justifies the multinomial approximation. On the other hand, in equation (3.1), we see that adding more gross outliers increases the worst case estimate for C in equation (3.6). In the next section, we study the possible applications that can use SWIFT as the front end. Later, we compare SWIFT with the state of the art method proposed in [33].



(a) changing ε (b) changing C (c) changing θ

Figure 3.5: The effect of changing different parameters on the sample size r. (a): Increasing the value of ε forces the method to select more samples in order to remain with the same probability. (b) and (c): increasing C or decreasing θ forces the method to grab more samples to reach the same probability.

3.7 Application Examples

As a generic unsupervised sparse sampling method, SWIFT can be used in virtually any scenario where multiple structures need to be detected in a large population of points. Here, a structure could be in a physical space (e.g. planar surfaces or other 3D structures), or in some abstract feature space (e.g. the space of all fundamental matrices, all homographies in some configuration of scene/camera motion, or subspaces formed in some high dimensional spaces). Below, we give some examples.

3.7.1 Detecting 2D lines

In this experiment, we consider detecting 2D lines. This is a classical model detection and it is used in this section to study the effect of SWIFT sampling on accuracy and time complexity of detected models. We show that sample sizes smaller than the one given by SWIFT can fail to detect all model instances and decrease the accuracy in model detection. Also, oversampling does not increase the likelihood of detection and would increase the computational time.

We generated a dataset of up to eight noisy lines that intersect each other. An example of this dataset has been shown in Figure 3.1. We detected the model instances by sampling a subset of points in a one-time grab sampling. Using this subset, we generated all the valid hypotheses and found their inliers. Finally, we applied mean-shift to discover the lines. To show that SWIFT sampling can guarantee a sufficient number of points and high accuracy, we ran three separate experiments: (i) using SWIFT to estimate the optimal sample size, (ii) an underestimated sample size, and (iii) an overestimated sample size. Table 3.1 shows the percentage of points sampled, the accuracy, and the time cost of the examined scenarios. As shown in the table, SWIFT sampling has similar accuracy compared with the result of the overestimated sample size. The underestimated sample size is faster but has lower accuracy. The time complexity using SWIFT sample size is an order of magnitude better than the over-estimated case. The data in this dataset is contaminated with Gaussian noise and the results are shown in the presence of (i) no gross outliers and (ii) 50% gross outliers.

Table 3.1: Performance comparison among 3 different scenarios of (i) SWIFT sample size (ii) underestimated sample size (iii) overestimated sample size.

Gross Outlier: 0%

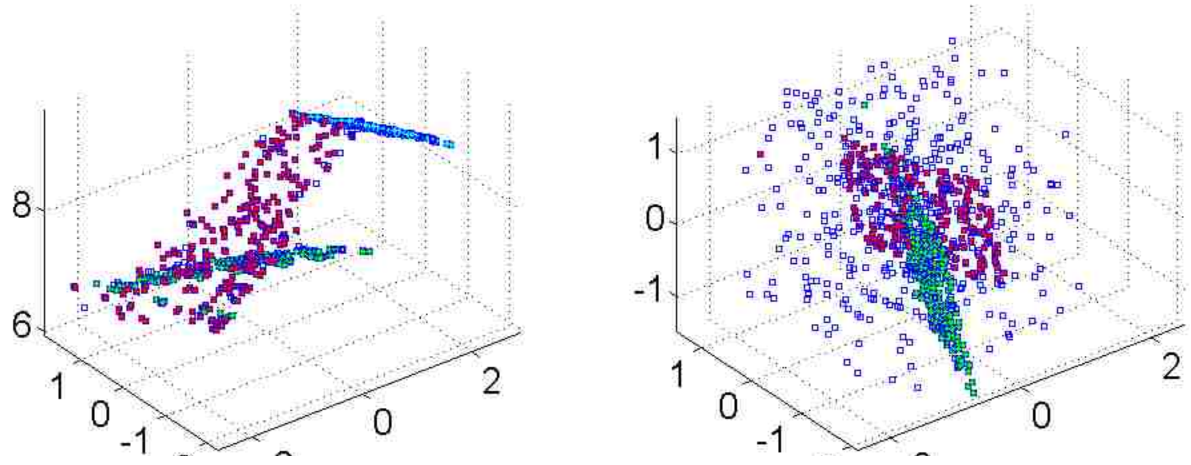
#Classes	2			4			6			8		
	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)
Underestimated	2.06	0.68	0.02	2.26	0.77	0.03	2.16	0.76	0.03	2.59	0.88	0.05
Overestimated	25.4	**1.00**	0.08	26.6	**1.00**	0.36	25.9	**1.00**	0.99	31.2	0.98	3.24
SWIFT (r)	**8.45**	**1.00**	**0.05**	**8.86**	**1.00**	**0.08**	**8.65**	**1.00**	**0.14**	**10.4**	**0.99**	**0.35**

Gross Outlier: 50%

#Classes	2			4			6			8		
	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)	%Sample	%Acc	Time(s)
Underestimated	2.19	0.96	0.03	2.60	0.91	0.05	2.56	0.92	0.08	2.58	0.87	0.10
Overestimated	26.2	**1.00**	0.87	31.2	**1.00**	2.68	30.4	0.99	6.49	30.6	0.98	11.9
SWIFT (r)	**8.74**	**1.00**	**0.14**	**10.4**	**1.00**	**0.33**	**10.1**	**1.00**	**0.69**	**10.2**	**0.98**	**1.15**

3.7.2 Detecting 3D Planes

Plane detection is a prerequisite for various computer vision tasks. In this experiment, we investigated the accuracy of our sparse sampling method for detecting planes in 3D space. In the first step, we considered synthetic models as illustrated in Figure 3.6. We examined two different scenarios. The first used a set of 3D points from Castelvecchio dataset [79] with three planes and no gross outliers (Figure 3.6a). Using SWIFT sampling just 5.6% of the data is used to detect planes in this figure. In Figure 3.6b, we used a synthetic dataset, with two planes and 50% gross outliers. Using SWIFT, 1.4% of data was selected for the detection step. As it is shown in Figure 3.6, the planes are detected correctly and gross outliers are not included in the models. Note also that the estimated sample sizes for both cases were similar despite different population sizes.



(a) 3 planes (b) 2 planes with 50% gross outliers

Figure 3.6: Using SWIFT to detect 3D Planes when $1 - \delta = 0.9$ (a): Data is from Castelvecchio dataset [79] where $r = 42$ and $N = 754$. (b): Blue points are outliers that are not grouped in any model when $r = 43$ and $N = 3000$.

In the second experiment, we examined real cases where images are collected with Kinect. Generally, the point clouds generated with Kinect include a huge number of points while the number of valid model instances in the scene is small. (i.e. N and θ are large relative to C). In these particular cases, the procedure of finding inliers for all the candidate model instances is a time consuming process since the total number of points $N = 167,028$ is extremely large. In order to overcome this problem, the SWIFT method was applied in two levels. In the first level, the value of r was computed to sample the minimum required number of points to instantiate each model candidate (which is $\varepsilon_1 = 3$ for detecting planes). Thus, the sample size r is used to instantiate all the model candidates. In the second level, we set the value of ε to a bigger number (e.g. $\varepsilon_2 = 100$) and sampled a subset of points from the population with a guaranty of selecting at least $\varepsilon_2 = 100$ points in each plane. The second subset of points was then used, instead of the entire population, as the group of points from which we selected the inliers for each model candidate. This example shows that SWIFT can be inherently implemented also in a multi-level setting. Figure 3.7 shows the point cloud data from Kinect used to accurately detect three planes in the scene using the SWIFT algorithm. By filtering the points with $depth = 0$ the total number of points in the cloud was $N = 167,028$ and $\theta = 30,000$. Setting $1 - \delta = 0.9$, the size of sampled points in the first level when $\varepsilon_1 = 3$ is $r = 43$ and in the second level when $\varepsilon_2 = 100$ is $r = 722$.

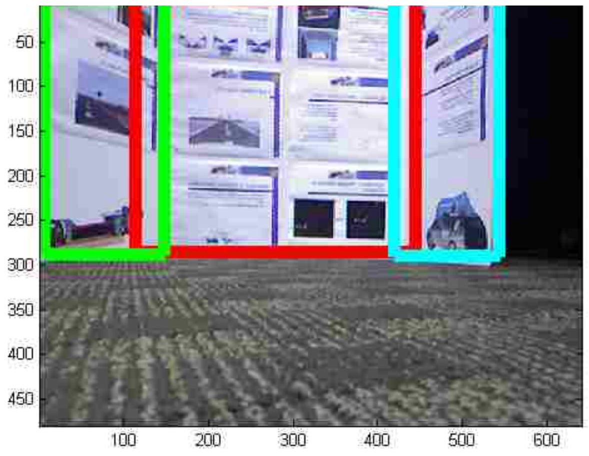
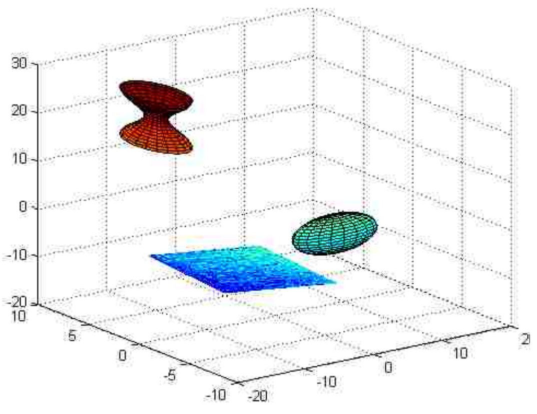


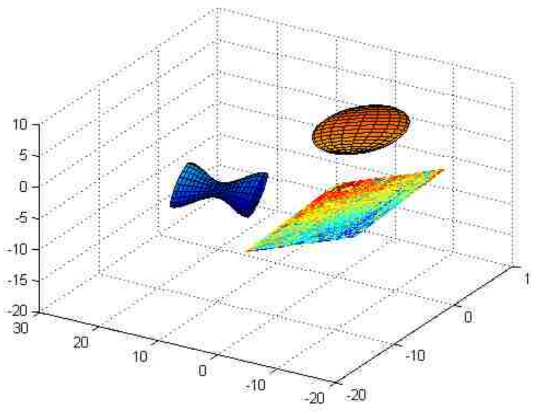
Figure 3.7: Detecting planes in 3D point cloud data collected using a Kinect.

3.7.3 Multibody Structure from Motion

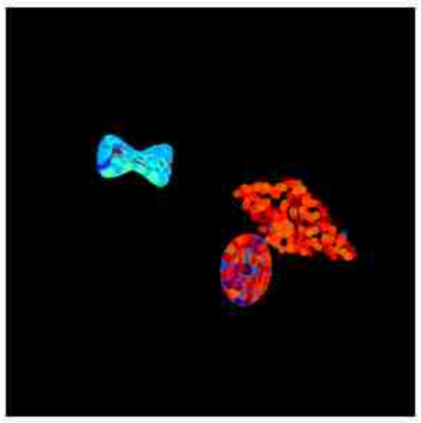
Estimating motion models in a video sequence is a classical problem in computer vision. This problem gets more complicated in dynamic cases when multiple rigid objects move independently in a 3D scene [88, 25]. Multibody structure from motion refers to the problem when there are several views of a 3D scene and the motions, structures, and camera calibration are unknown. Recent studies in this area suggest various solutions to this problem [88, 72]. In this section, we used the method in [72] but used SWIFT in the sampling step. The first assumption in [72] is that the number of independent motions in the scene and their parameters are unknown, where each motion may either be estimated with a homography or a fundamental matrix. Thus, to start the process a set of 2-D point correspondences is required. Then a fixed number of point sets are randomly sampled to generate candidate homographies and fundamental matrices, using the constraints that the minimum required correspondences for a homography is 4 and for fundamental matrix is 7. A shortcoming of the method in [72], however, is that one must specify the number of samples in order to ensure detecting all the motions in the scene. To investigate the application of SWIFT in this problem, we generated 100 synthetic scenes each containing three 3D-objects (not necessarily planar) and a single moving camera. For each synthetic scene, an initial 300×300 image was created. The 3D-objects and the camera were moved randomly and independently and then a new 300×300 image was taken. An example of 3D-objects and their 2D projections are shown in Figure 3.8. Using the images, point correspondences were generated by selecting 50 random points from each object. Assuming that points at close proximity are likely to belong to the same object, the sampling strategy explained in [72] divided images heuristically into 9 overlapping areas and sampled points locally. In our experiment, to exploit the proximity constraint, we applied a simple image segmentation algorithm to divide the image into separate clusters. Later, we show that the accuracy of image segmentation does not affect the final results.



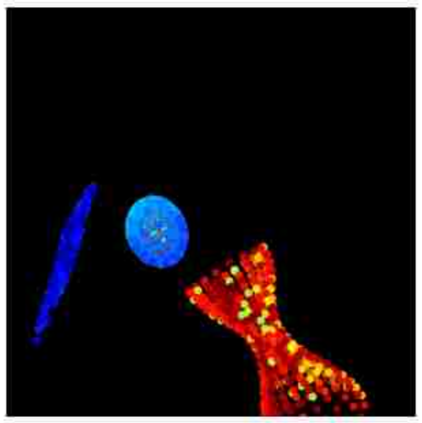
(a) First scene



(b) Second scene



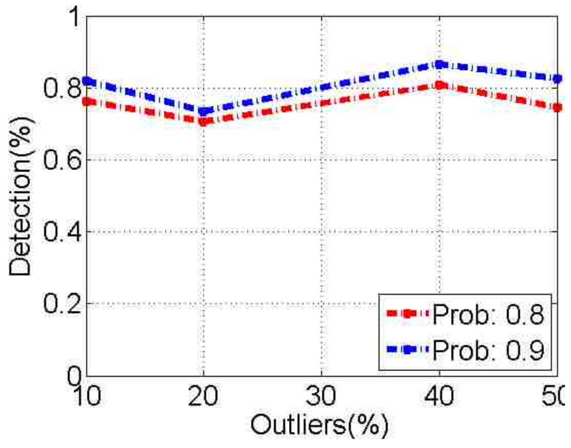
(c) 2D projection of 3.8a



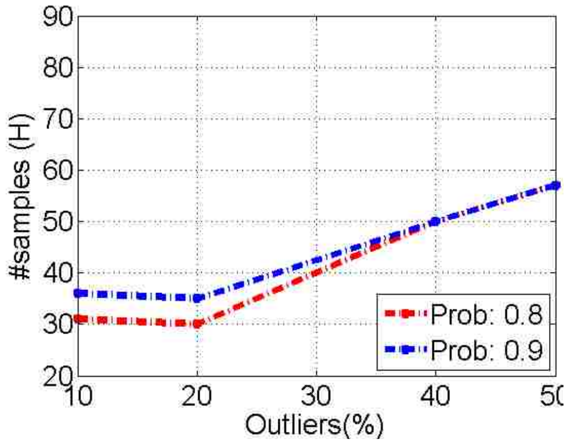
(d) 2D projection of 3.8b

Figure 3.8: Synthetic data for multibody structure from motion. The outliers are added after moving objects and computing the 2D projections. These outliers are not shown in this figure. (a): The 3D objects that are not necessarily planar. (b): 3D objects are moved randomly and independently. (c): The image is taken from (a). (d): The image is taken from (b) after moving the camera.

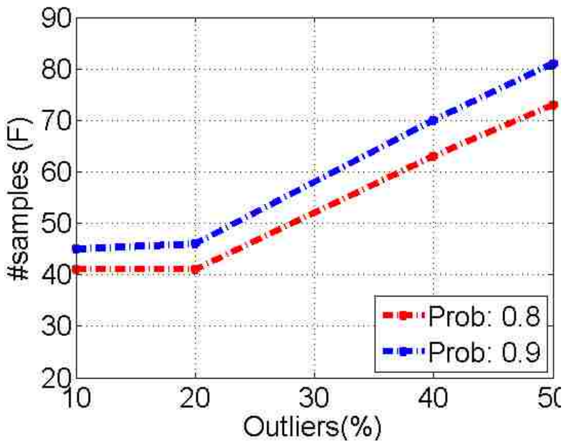
The proposed method in [72] samples a batch of ε-tuples to detect fundamental matrices and the homographies. Since the degree of freedom of fundamental matrices and the homographies are different, separate sets should be sampled for detecting each of the matrices. In SWIFT, however, r points are sampled in a one-time grab. We can employ the sampled set r to find both homographies and fundamental matrices, i.e. since ε for a fundamental matrix is greater that ε for a homography, we can sample r as the number of points required for finding all fundamental matrices. A subset of r is sufficient to find the homographies. In the first experiment, the effect of changing the accuracy of image segmentation was studied. In this experiment, based on the chosen value of θ, the sample size r was computed and grabbed from the total population of $N = 200$ points, with 50 points (25%) of gross outliers added to the correspondences. Using the sample set of r points, the number of inliers selected in each segment is computed. Figure 3.10 shows the number of inliers per segment as the accuracy of image segmentation is increasing with $\varepsilon_h = 4$ and $\varepsilon_f = 7$ for the homographies and fundamental matrices, respectively. As can see from this experiment, a key advantage of using SWIFT sampling is that the required number of samples to maintain a certain level of accuracy with a given probability $1 - \delta$ can be calculated. Therefore, the accuracy of the results can be maintained stable as illustrated in Figure 3.9. In fact, as the number of gross outliers is growing, the size of population N is also increasing. Since, the other parameters θ, δ and ε are fixed, increasing N leads to selecting a more accurate number of points r. Figures 3.9, (a) to (d) demonstrate the idea of automatic adaption of r and stability of SWIFT sampling in terms of accuracy of results.



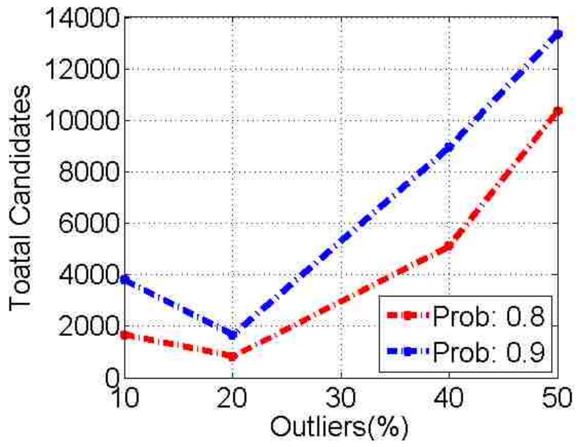
(a) Detected



(b) r: Homographies



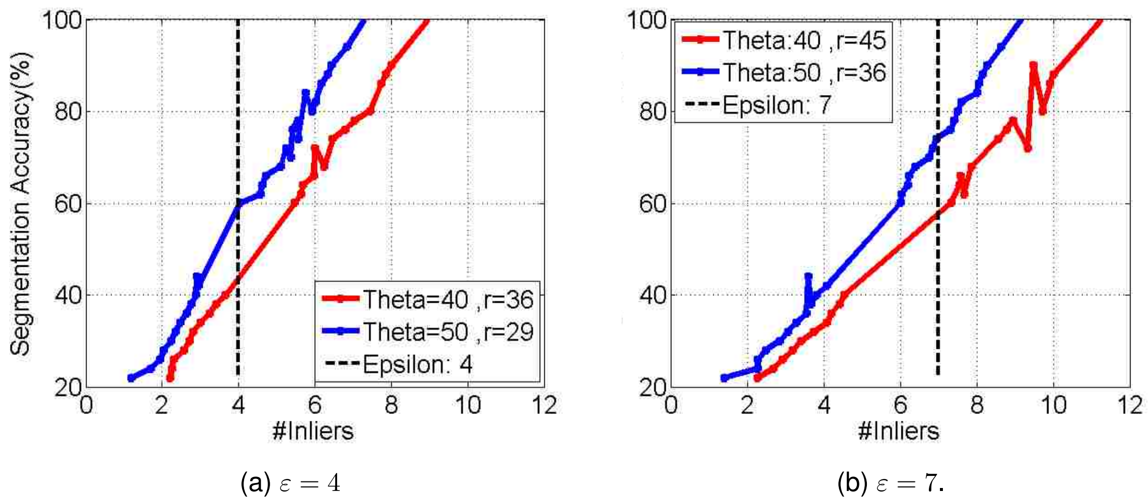
(c) r: Fundamental Mat



(d) #Candidates.

Figure 3.9: The effect of changing percentage of outliers on required sampled points and accuracy of multibody structure form motion in [72]. The results are in the presence of 3 independent motions and when the image segmentation has an average accuracy of 65% and $\theta = 40$. (a): $\%$ of Detected models. (b), (c): Values of r when $\varepsilon = \{4, 7\}$ respectively. (d): initial candidates for both homographies and fundamental matrices.

In Figure 3.10a, for input parameters ($\varepsilon = 4$, $\theta = \{40, 50\}$, $1 - \delta = 0.9$), the computed SWIFT sample size are $r = \{42, 32\}$ for mutlinominal model and $r = \{36, 29\}$ for hypergeometric model. In (b) for ($\varepsilon = 7$, $\theta = \{40, 50\}$, $1 - \delta = 0.9$), the computed SWIFT sample size $r = \{63, 49\}$ and $r = \{45, 36\}$ for the mutlinominal and hypergeometric models respectively.



(a) $\varepsilon = 4$ (b) $\varepsilon = 7$.

Figure 3.10: The effect of image segmentation accuracy on the average number of inliers sampled in each model instance when we have three motions.

To examine the method on a real case, we used the image data in [72] that include three motions and $\%25$ gross outliers. Using the SWIFT algorithm, we calculated the sample points for both homographies and fundamental matrices. First, we used SWIFT to sample a sufficient number of points. Assuming that points at close proximity belong to one motion, we applied a basic image segmentation in order to cluster objects of the scene (Figure 3.11a). Then the initial candidate homographies and fundamental matrices were calculated using the method in [72], with segmentation used as a proximity constraint. In this example, the average number of initial candidates (homographies and fundamental matrices) generated over 20 trials, were $13,073$. Basically, the estimated SWIFT sample size r guarantees with probability $1 - \delta = 0.9$ that the group of candidates includes all the existing motions in the scene.

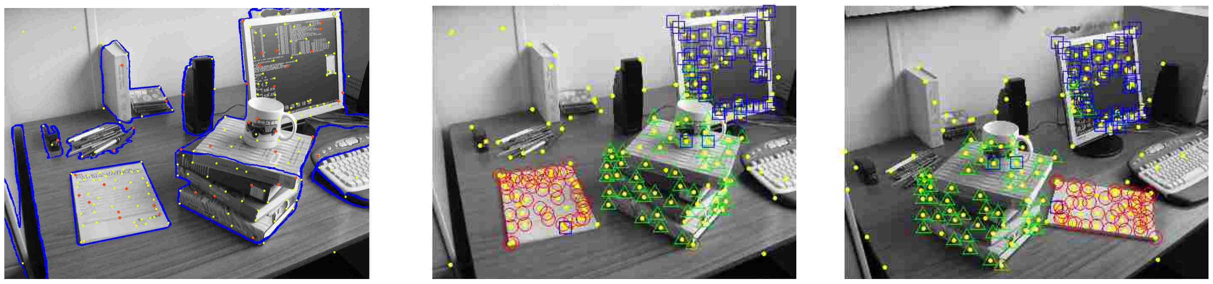

(a) Segmented (b) Left Image (c) Right Image

Figure 3.11: Detecting multibody structure from motion when the segmentation accuracy is 80% and $N = 200$ includes 25% gross outliers. For fundamental matrices, the sample sizes are $r = 63$ and $r = 50$ for multinomial and hypergeometric models, respectively. (a): segmented image. Yellow dots are the correspondences, and red stars are the sampled points using SWIFT. (b),(c): images with 3 objects moved independently and the detected motions using the method in [72].

3.8 Discussion and Conclusion

This chapter introduces a sparse one-time grab random sampling method, which together with an unsupervised clustering method, can be used to simultaneously detect multiple structures or subspaces in a large population of high-dimensional data with an overwhelming percentage of outliers. When the data is overly large, a popular approach is to sample a subset of the data that remains representative of the whole population. SWIFT provides a generic solution to a well-known question: How big should the sampled "subset" be in order to remain "representative" of the whole population? Application examples are numerous, and some were discussed in this chapter. For instance, one case is when a large set of points are tracked across many frames by a moving camera observing several independently moving objects in the scene - a problem known as multi-body structure from motion in computer vision.

We proved that one-time grab random sampling can be accurately modeled using either a hypergeometric or a multinomial pmf (i.e. as either sampling without replacement or with replacement under some constraints). The hypergeometric pmf is the exact mathematical model for one-grab sampling, since it is equivalent to sampling without replacement. We found a non-decreasing relation for estimating the sample size for the hypergeometric model, where one can find the sample size by using a binary search in a time complexity of $O(\log N)$. We carefully derived tight upper bound and lower bound for this model and illustrated experimentally the accuracy. However, when N is too large, finding the sample size can be time consuming. Thus, we also derived an approximation of the random sample size using a multinomial model of pmf (i.e. sampling with replacement assuming $N \gg r$). This approximation can be obtained in $O(1)$ and does not require a searching process. As it is shown, this approximation is not as tight as the estimated sample size through the search process. We carefully studied the behavior and accuracy of these models, providing a practical method of selecting the minimum sample size that guarantees with some probability the detection of all model instances (e.g. subspaces) in the data. In addition to accuracy, one desirable behavior of SWIFT sampling is “sparseness”, which we have shown is independent of the population size, making our solution important for processing and analysis of “big data”. The problem involves the solution to a hard inverse problem, which we solved by finding tight bounds to the solution.

An important lesson learned in this study is that our solution makes one-grab sampling methods competitive with multi-grab methods. Popular multi-grab methods such as RANSAC or its variations have long established the answer to their sampling questions such as sample size (per grab) and the number of sampling iterations. Our study now reveals that the main disadvantage of multi-grab methods is that they have to guarantee sufficient statistic at each iteration, which over a large number of iterations leads to a huge number of samples. By settling the unanswered sampling question of sample size with sufficient statistic for one-grab methods, we show that this class of methods have the huge advantage of guaranteeing a sparse subset of data can solve the usual data mining problems for “big data” without resorting to heuristics. To demonstrate these points, we compared the sample sizes computed by SWIFT against the number of points sampled in other existing methods. As mentioned earlier, sampling methods such as [33], sample a batch of ε-tuples, with the total number of sampled points as ε times the number of ε-tuples. This is, in some ways, similar to multi-RANSAC. Figure 3.12 compares the number of sampled points in SWIFT with the method in [33], and the sequential-RANSAC [90]. The values for sequential-RANSAC are computed using the number of required sampled points to detect each model instance multiplied by the maximum number of instances C [33]. As illustrated, the sample size obtained by SWIFT sampling is significantly smaller than the values in the other two methods.

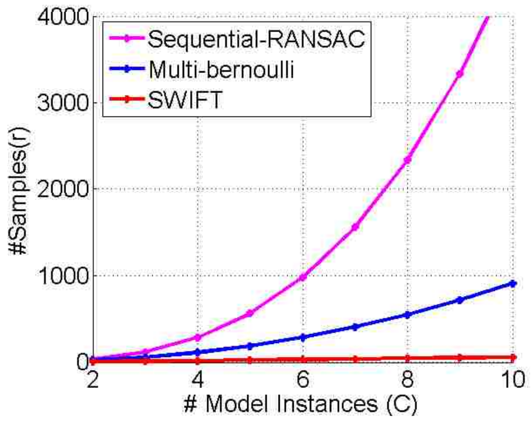


Figure 3.12: Comparing the averaged number of samples r over 200 trials in sequential RANSAC, the proposed method (MBSAC) in [33], and SWIFT when $N = \{10^2, 10^3, 10^4\}$, $\varepsilon = 2$ and $P = 0.9$.

CHAPTER 4: SCALABLE SUBSPACE CLUSTERING BY SPARSE SAMPLING

4.1 Method Overview

The main idea in graph-based subspace clustering is that data is self-representative. Thus, by having sufficient number of points in each subspace, any point in a subspace can be represented as a linear combination of other points in that subspace. In other words, given $\{x_i \in \mathbb{R}^n\}_{i=1}^{N}$ defines a collection of N data points in an ambient space of dimension n drawn from a union of C independent linear subspaces $\{S_k\}_{k=1}^{C}$ with dimension $\{d_k \ll n\}_{k=1}^{C}$. If $x_i \in S_k$ and subspace S_k includes sufficient number of points, x_i can be represented by a linear combination of other points in the subspace S_k. Sufficient number of points in each subspace means that there are enough points to determine the learned bases of each subspace. Thus, in subspace S_k of dimension d_k with N_k points, it is necessary for the ratio of $\rho_k = \frac{N_k}{d_k}$ to be greater than one in order to determine the bases of the subspaces. This ratio indicates the number of samples per dimension in subspace S_k. In the literature [75, 40], this is called *sampling density* of the subspace. If ratio ρ_k is smaller than one, then there is an insufficient number of points to span all the directions of subspace S_k, and therefore subspace clustering algorithms will fail to detect the subspace correctly. On the other hand, high sampling density ρ can create unnecessary overhead in which it increases the computational complexity of subspace clustering. Authors in [75] study the effects of sampling density of subspace and the affinity between subspaces in the method introduced in [21]. They show that the accuracy of subspace clustering depends on decreasing the affinity between subspaces as well as the increasing the sampling density in each subspace. However, they show that: (i) when subspace dimensions are fixed and the sampling density ρ becomes exponentially large, the accuracy of subspace clustering decreases; (ii) under some conditions, increasing the number of inliers of subspaces does not noticeably change the accuracy of the subspace clustering algorithm. This confirms that, in those cases, oversampling can slow down the clustering process without improving the accuracy of the final result. In this chapter, we study the effect of using SWIFT sampling in reducing the computational complexity of graph-based subspace clustering.

4.2 SWIFT and Sparse Subspace Clustering

Graph-based subspace clustering algorithms, e.g. [21, 49], can estimate clusters with high accuracy. However, it is computationally expensive when we have a very large dataset. An efficient solution to reduce the time complexity of the subspace clustering algorithm, such as the SSC algorithm [21], would start with a subset of points sampled from the population [58, 93]. The process of finding sparse coefficients and clustering points can be done in two steps. First, find sparse coefficients for a sample set of r points. Second, locate inliers for each cluster. As mentioned in the literature review in Chapter 2, although this idea was explored in [64], their choice of size for the sampled subset was rather ad-hoc. In particular, SSC requires that each subspace of dimension d_j have at least $d_j + 1$ points for the method to work. This is a natural setting for SWIFT, since it guarantees to select, with high probability, a minimum subset of points required to discover all subspaces. Furthermore, SWIFT provides a tight estimate to ensure low time-complexity. Once the subspaces are estimated in the sampled subset, they serve as the initial solution to clustering the whole population of data. The proposed SWIFT-SSC method is summarized in algorithm 3.

To be able to use SWIFT to estimate the sample size r, we need to study the choice of SWIFT input parameters $(\varepsilon, \theta, \delta)$. Similar to experiments in the previous chapter, the value of probability $1 - \delta$ must be close to one in order to compute sampling size r accurately. The value of ε should be large enough to ensure that we sample a sufficient number of points from each subspace. By having C subspaces of size $d_1, ..., d_C$, parameter ε needs to be set to the largest possible dimension of subspace in the population $\max_k(d_k)$. In most subspace clustering applications, it is possible to determine the largest dimension of subspaces. For instance, the set of face images under all possible illumination conditions can be well approximated by a 9-dimensional linear subspace [44]. Similarly, handwritten digits with some variations lie on 12-dimensional subspaces [30]. Value of subspace size θ can also be calculated having the number of subspace C. In most of graph-based subspace clustering algorithms, e.g. SSC [21], the number of subspaces in the population is given as the input parameter. This knowledge can be used here to find value of θ using formula (3.1). The effect of choosing these two parameter ε and θ is studied in section 4.3.

Algorithm 3 Subspace clustering using SWIFT

```
Input: High dimensional data points $X \in \mathbb{R}^{n\times N}$ and the number of desired clusters $(\varepsilon, \theta, \delta)$
1: Set $\varepsilon = \max_k(d_k)$,
       $\theta = \lfloor \frac{N}{C} \rfloor$
2: SWIFT Sampling Find sample size $r$ using SWIFT sampling proposed in Chapter 3
3: Form $\chi \in \mathbb{R}^{n\times r}$ by Sampling $r$ columns of dataset $X$ uniformly at random
4: Cluster sampled points : Find pairwise similarity matrix for $\chi$ sample subset and cluster
   them using (2.5).
5: Normalized residual: Find normalized residual of out-of-sample points $\chi^*$ in every cluster
   using equation (2.8)
       $r_k(x_i^*) = \frac{\|x_i^* - \chi\xi_j(Z_i^*)\|_2}{\|\xi_j(Z_i^*)\|_2}$
6: Find inliers : Determine the best cluster for out-of-sample points $\chi^*$ using $\arg\min_k r_k(x_i^*)$
```

4.3 Experimental Evaluation

In this section, we study different scenarios for subspace clustering and the effect of using sparse sampling (SWIFT) on the accuracy and time complexity of the final outcome. Later, we show the results as applied to face clustering as an example of real application.

4.3.1 Density of Subspaces

As stated previously, the study in [75] demonstrates that improving the accuracy of SSC depends on increasing the number of inliers in each subspace as well as decreasing the affinity between subspaces. They show that, under some conditions, increasing the number of inliers of subspaces does not noticeably improve the accuracy of the subspace algorithm. It is demonstrated that one gets a stable level of accuracy when the ratio $\rho \approx 3$. In this experiment, we show that SWIFT sampling can be used to sample sufficient points to detect all the subspaces with a high level of accuracy. Using the study [75], we can estimate the parameter ε. We set $\varepsilon = \rho d + 1$ and show that when $\rho \in \{3, 4\}$, we sample sufficient inliers from each subspace to cluster them with high accuracy. Oversampling increases the time complexity without noticeable improvement in accuracy.

Below, we show that SWIFT sampling attains the optimal number of inliers for each subspace. The data in this experiment included 3 subspaces of dimension $d = 20$ in $\mathbb{R}^{40}$ with 500 points in each subspace. The angles between the subspaces were $\frac{\pi}{3}$, which caused the subspaces to overlap. The results in [75] show that the accuracy of SSC did not improve significantly after $\rho_j = \frac{N_j}{d_j} = 3.25$. In this experiment, to demonstrate the effect of SWIFT sampling, we varied the ratio ρ in the range $\rho \in [1, 7]$. Ground-truth is defined as sampling $\rho \times d$ points from each subspace. SWIFT sample size is computed when $N = 4500$ and SWIFT input parameters are $\varepsilon = \rho d + 1$, $C = 3$, and $1 - \delta = 0.9$. As illustrated in Figure 4.1, the accuracy is roughly stable when $\rho \in [3, 4]$ and the estimated SWIFT sample size are very close to the ground truth. We show SWIFT sampling for both the hypergeometric pmf (3.6)-(3.8) and multinomial pmf (3.14) models are estimated very close to the ground truth.

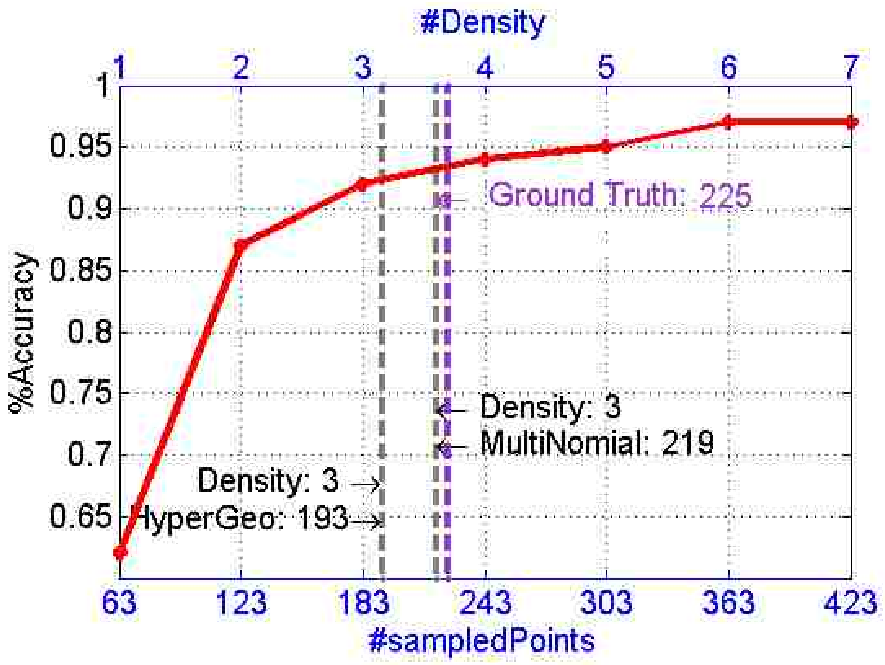


Figure 4.1: The effect of changing the ratio of inliers (ρ) on accuracy of clustering. The estimated SWIFT sample size when $1-\delta = 0.9$ is shown in gray dotted lines. When $\rho = 3$, the actual size of the sampled population needs to be 225 (ground truth), the calculated value by SWIFT sampling using hypergeometric and multinomial pmf models are 193 and 219, respectively. These results were averaged over 100 trials.

4.3.2 Subspaces with Different Dimensions

In the case of subspaces with different dimensions d_j, we are still able to compute the required sample size using the SWIFT algorithm. We need to make sure that a sufficient number of points, $\rho d_j + 1$, are sampled from each subspace. This means that we need to make sure a subspace with the largest dimension, $d_m = \max_j\{d_j\}$, has enough points to be detected. Thus, in the SWIFT algorithm, we can set $\varepsilon = \rho d_m + 1$, which forces (with the probability $1-\delta$) the sampling process to select ε points from each of the subspaces, even those with lower dimensions. As it turns out, this still gives us a very sparse subset of the points that can be used to detect the subspaces with a very high probability and accuracy. To show the sparsity of the computed sample size in SWIFT, we generated a dataset of 4 subspaces. The subspaces have dimensions $d_j = \{2, 5, 10, 15\}$, with each containing 2500 points in an ambient space of dimension 50. In order to compute the sample size r, we need to set $\varepsilon = \rho \times d_{max} + 1$, where $d_{max} = 15$ is the maximum dimension of the subspaces. Table 4.1 shows the values of the computed sample sizes using SWIFT and compares them with manual sampling of points, setting $\varepsilon = \rho \times \{2, 5, 10, 15\}$. However, a manual sampling would be only possible if one can accurately distinguish the subspaces to sample the required number of points from each of them. Since, in practice, we do not have such prior knowledge about the subspaces, using SWIFT is the most optimal option. SWIFT selects points uniformly with sufficient number of points in each subspace, even though it adds a small overhead to some subspaces. As presented in Table 4.1, the computed value of the SWIFT sample size is still very sparse compared to the population size. The ground truth of the optimal sample size using statistical simulation is also added to this table for comparison.

Table 4.1: Comparing the size of SWIFT sampling, the manually sampled points, and the ground-truth. In the SWIFT algorithm $d_{max} = 15$, and $P = 0.9$.

Population Size	ε	**Density**	**#Manually Sampled**	**#HyperGeo**	**#Multinomial**	**Ground Truth**
10000	15	1	36	93	93	96
	30	2	68	157	164	167
	45	3	100	221	232	234
	60	4	132	277	299	300
	75	5	164	332	365	366
	90	6	196	381	432	431

4.3.3 Face Clustering With Illumination and Pose Variation

Face clustering is one of the many applications in subspace clustering. Face image clustering techniques try to cluster images of the same subject under varying lighting conditions in one group. Studies in [44, 2, 32] show that a set of images of an object under varying illumination lies in a low-dimensional linear subspace of the image space of up to nine dimensions. This can be used to determine the minimum sample set ε in the SWIFT method. The Extended Yale B Database [26] is a facial dataset widely used in subspace clustering literature [21, 64, 96] and contains $2,414$ frontal face images of 38 human subjects taken under approximately 64 different illumination conditions. Figure 4.2 shows some examples of images in this dataset. In this experiment, we used the SWIFT algorithm to study the sample size required to cluster face images using [20].

Figure 4.2: Example of Images in the Extended Yale B Database.

We used subsets of 5 different subjects from the dataset. Each subject includes 64 images ($\theta = 64$). By changing the ratio of the inliers (ρ) in the subspaces, we examined the accuracy of the subspace clustering algorithm. The graph in Figure 4.3 shows the accuracy of the clustering algorithm as a function of ρ. In addition, the calculated sampled size using the SWIFT algorithm is shown for $\rho = 3$. As we can see in this figure, both multinomial and hypergeometric SWIFT estimations give answers that are very close to the ground truth. The multinomial pmf model overestimates the sample size, while the hypergeometric model underestimates it, but both are close to the ground-truth sample size.

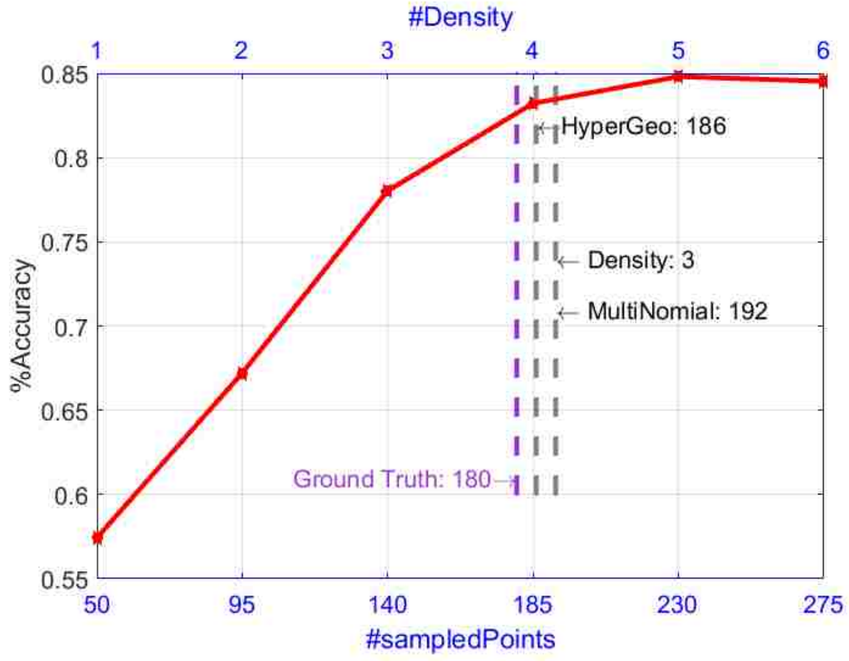


Figure 4.3: Accuracy of clustering face images. Computed SWIFT sample sizes using both multinomial and hypergeometric estimations are shown in gray lines. In this experiment, $N = 320$, $P = 0.9$, and $C = 5$. Results are averaged over 100 images.

4.4 Discussion and Conclusion

In this chapter, we proposed a method for analysis large-scale and the out-of-sample clustering problems for high dimensional points. We studied the effect of one-time grab sampling strategies on finding the required number of high dimensional points in the selected subset in order to detect all the subspaces with a high accuracy and efficiency. We solve an important problem of large scale subspace clustering in high-dimensional "big data" for finding structures, patterns and lower dimension of subspaces. The key question of how many samples one should take to ensure with some probability that all subspaces are adequately represented in the sampled subset has been answered in this chapter.

CHAPTER 5: PROBABILISTIC SPARSE SUBSPACE CLUSTERING USING DELAYED ASSOCIATION

5.1 Method Overview

In this chapter, we introduce a joint probabilistic based subspace clustering method where we estimate the reliability of the cluster assignment for each point before assigning a point to a subspace. We group the data points into two groups of certain and uncertain in which we use hard and soft clustering assignments for these two groups respectively. We try to improve the subspaces assignments by alternating between searching for a better self-representative matrix Z and a better subspace clustering result. It is shown that engaging both soft and hard clustering is more suitable in handling noise/outliers removing ambiguity caused by intersection between subspaces. Similar to the methods described in the literature review in Chapter 2, the method proposed in this chapter is based on a joint optimization approach that converges to an optimal solution for self-representation of clusters in high dimensional data. The novelty of our approach is that we delay the association of a point to a cluster at a given iteration, when such association is *uncertain*, as described next. This has an advantage of converging to a more accurate solution when the subspaces have high overlap (intersections), or when data is contaminated with outliers/noise. We describe assignments of points to C clusters by a soft subspace segmentation matrix $\Phi \in [0,1]^{N\times C}$ where elements ϕ_{ik} represent the probability of point i belonging to subspace k, so that

$$\sum_{k=1}^{C} \phi_{ik} = 1, \quad i = 1, \ldots, N. \tag{5.1}$$

In particular, $\phi_{ik} = 1$ when point i is confidently assigned to cluster k, and $\phi_{ik} = 0$ when point i is confidently excluded from cluster k. Thus, we can define the *association matrix* A as:

$$A = \Phi\Phi^T. \tag{5.2}$$

Elements a_{ij} of matrix $A \in [0, 1]^{N\times N}$ indicate the strength of the connection between points i and j in the dataset. If Φ were a clustering matrix with entries of only zero or one, then one would have $a_{ij} = 1$ if points i and j lie in the same class and $a_{ij} = 0$ otherwise. In the soft clustering case $\Phi \in [0, 1]$ and a_{ij} represents the probability of connection of two points if they are segmented in one cluster.

The sparse similarity matrix $\bar{Z}$ indicates the connection between each point and all other points in the dataset. On the other hand, the association matrix A represents the relationship between clustered points. Hence, these two matrices are related to each other and present similar information about points in the dataset. The association matrix A can be used to denoise the coefficient matrix Z and get a better representation of points in their own subspace, while taking into account the coefficient matrix Z can lead to a more accurate recovery of the association matrix A. The following equation uses a similar concept as equation (2.5) and formulates the connection between Z and the newly introduced matrix A:

$$\min_{Z,E,A} \left(\lambda_0 \|Z\|_1 + \frac{1}{2}\|E\|_F^2 + \lambda_1 \|(\mathbf{1} - A) * Z\|_F^2 \right) \tag{5.3}$$
$$\text{subject to} \quad E = X - XZ, \quad z_{ii} = 0, \quad A = \Phi\Phi^T, \quad \text{rank}(\Phi) = C.$$

Where $\mathbf{1}^{N\times N}$ is the matrix with all unit elements, $E = XZ - X$ is the error in the calculated representation of each point, $*$ is the Hadamard product, and λ_0 and λ_1 are positive regularization (trade off) parameters. The constraint $z_{ii} = 0$ forces the main diagonal to be close to zero to avoid a degenerated solution. The first and second terms in (5.3), similar to equation (2.5), enforce the sparsity and small errors between points and their linear representations, respectively. The last term $\|.\|_F^2$ in this equation imposes connectivity between points in the same subspace and removes connectivity of points in different subspaces. Indeed, the entries a_{ij} of matrix A represent the probabilities of points i and j being in the same cluster and have been used in representation of each other. Thus, when two points x_i and x_j are segmented in two different subspaces then the elements $a_{ij} = 0$ and $a_{ji} = 0$ and term $\|(1 - A) * Z\|_F^2$ forces the respective entries z_{ij} and z_{ji} of matrix Z to be small. In other words, $\|(1 - A) * Z\|_F^2$ term adds an extra error weight to the optimization formula when points are not clustered in the same subspace but have fairly strong connectivity in the coefficient matrix Z.

Our proposed method jointly searches for a sparse self-representation matrix Z that satisfies $E = X - XZ$ and the soft subspace segmentation matrix Φ that satisfies rank$(\Phi) = C$. In order to identify the subspace clusters and mitigate/eliminate noise and outliers from the coefficient matrix Z and the association matrix A, we alternate between finding the association matrix A and the coefficient matrix Z. For a given matrix A, the objective function (5.3) is convex in $\{Z, E\}$ which can be solved efficiently using the alternating direction method of multipliers (ADMM) [5]. Given $\{Z, E\}$, we estimate A using spectral clustering. A main novelty of our approach is in the second step, where given the matrix $\{Z, E\}$, we generate the association matrix $A \in [0, 1]^{N\times N}$.

5.2 Joint Minimization Framework for Subspace Clustering

As stated previously, the coefficient matrix Z and the association matrix A are computed via a joint optimization approach: (i) Given the coefficient matrix Z, and error matrix E we approximate the association matrix A by spectral clustering; (ii) Given the association matrix A, we minimize (5.3) with respect to the coefficient matrix Z. We repeat this process until convergence.

5.2.1 Delayed Association Representation

Updating Φ and A: Given the similarity matrix $\bar{Z}$ and the error matrix E, our objective is to find the soft subspace segmentation matrix Φ and the association matrix A. The solution for determining Φ as the probability of assigning points to C clusters can be defined as a pairwise data clustering problem [85, 73]. One can find hard clusters by applying a spectral clustering algorithm such as normalized cuts. The spectral clustering algorithm divides points into C separate subspaces, using similarity matrix $\bar{Z}$, where each element represents the connection between points. The clusters can be obtained by applying the K-means algorithm to eigenvectors associated with C smallest eigenvalues of a Laplacian matrix driven from the similarity matrix $\bar{Z}$. Given the hard clusters obtained from the spectral clustering algorithm, we need to determine the likelihood of a point x_i belonging to each subspace $\{S_k\}_{k=1}^{C}$, for the purpose of computing Φ. We define:

$$\left\|\xi_{s_k}(\bar{Z}_i)\right\|_1, \tag{5.4}$$

as the *degree of association* of each point x_i with the subspace $\{S_k\}_{k=1}^{C}$, where $\bar{Z}_i$ is the i column of similarity matrix $\bar{Z}$ corresponding to point x_i, and $\xi_{s_k}(\bar{Z}_i)$ is found by keeping all the elements of the vector $\bar{Z}_i$ that are associated with subspace S_k, and setting the remaining elements to zero. Each point $x_i, i = 1, \cdots, N$, is assumed to be more likely to be associated with a subspace S_k if it has a higher degree of association to the subspace S_k. This probability is defined as:

$$p_{ik} = \frac{\left\|\xi_{s_k}(\bar{Z}_i)\right\|_1}{\left\|(\bar{Z}_i)\right\|_1} \quad k = 1, \cdots, C. \tag{5.5}$$

Using this formula, we build the association probability matrix $P \in [0,1]^{N\times C}$ with elements p_{ik} being the probabilities of assigning point x_i, $i = 1, \cdots, N$ to the subspace S_k, $k = 1, \cdots C$, where $\sum_{k=1}^{C} p_{ik} = 1$. The soft subspace segmentation matrix Φ is determined using the computed association probability matrix P as described below. For each i, we denote $k_i = \arg\max_{1\le k\le C} \{p_{ik}\}$ and divide points into *certain* and *uncertain* using the delayed association parameter Ω.

$$\phi_{ij} = \begin{cases} 1, & \text{if } j = k_i \quad \text{and} \quad p_{ik_i} \ge \Omega \\ 0, & \text{if } j \ne k_i \quad \text{and} \quad p_{ik_i} \ge \Omega \\ p_{ij}, & \text{if } p_{ik_i} < \Omega \end{cases} \tag{5.6}$$

The delayed association parameter Ω is calculated by finding the average affinity between points of a subspace using the following equation:

$$\Omega = 1 - \frac{\sum_{i\ne j} M_{ij}}{(C-1)\sum_{i=j} M_{ij}} \quad \text{where} \quad M = P^T P \tag{5.7}$$

Note that matrix $M \in \mathbb{R}^{C\times C}$ demonstrates the affinity between points in subspaces. The main diagonal of this matrix shows the correlation between points of a subspace and off-diagonal entries showing the similarity of points in different subspaces [41]. Thus, when the similarity matrix $\bar{Z}$ turns into a block diagonal matrix, the probabilities p_{ik} will be pushed through zero or one and the defined matrix M turns into the identity matrix with low affinity between points of different subspaces and strong connectivity among points of a subspace. We use matrix M to find the delayed association parameter Ω in each iteration. Based on the definition of Ω in equation (5.7) when there is a high ambiguity between clustered points, matrix M turns into a matrix containing all similar entries and $\Omega \approx 0$. This allows more points to be grouped as *uncertain* and give them the chance to find a better representation before being assigned to a subspace. On the other hand, when there is a low affinity between points of different subspaces, M turns into the identity matrix and $\Omega \approx 1$, which allows more points to be grouped as *certain*.

Given an assignment matrix Φ, we form the association matrix $A \in [0,1]^{N \times N}$, as $A = \Phi\Phi^T$, which is a symmetric matrix. For any pair of points x_i and x_j marked as *certain*, $a_{ij} \in \{0,1\}$ shows if point x_j is assigned to the same subspace as x_i ($a_{ij} = 1$) or are segmented in different subspaces ($a_{ij} = a_{ji} = 0$). For *uncertain* points $a_{ij} \in [0,1]$ represents the probability of assigning x_i and x_j to a same subspace. The rationale behind the method is that, in the sparse similarity matrix $\bar{Z}$, each column i includes the coefficients associated with other points used to represent the point x_i. These coefficients indicate the connection between a point x_i and all other points. When a point is marked as *certain* in the association matrix A, we discard coefficients from other clusters by setting $a_{ij} = 0$, even when the values are large. By setting $a_{ij} = 1$ for all $i, j \in S_k$, we also improve the connection between points within the same subspace. For an *uncertain* point, however, there is an ambiguity regarding the correct cluster. Using the definition in formula (5.6), we preserve all strong connections, regardless of the cluster to which it is assigned in the spectral clustering step. We include all the strongly connected points while updating the coefficient matrix Z in the next iteration. This approach helps us improve the connections between points and reduce noise in the next iteration. This process is summarized in algorithm 4.

Algorithm 4 Finding soft segmentation matrix Φ

```
Input: Similarity matrix $\bar{Z}^{(t)}$ and hard clustering assignments
1: Compute Association Probability $P^{(t)}$ using equation (5.5)
2: Compute Delayed Association Parameter $\Omega^{(t)}$ using equation (5.7)
3: for $i \in \{1, .., N\}$ do
4:     Set $k_i = \arg\max_{1 \leq k \leq C} \{p_{ik}^{(t)}\}$.
5:     if $p_{ik_i}^{(t)} \geq \Omega^{(t)}$ then {Mark $i$ as "certain"}
6:         for $j \in \{1, .., C\}$ do
7:             if $j = k_i$ then
8:                 $\phi_{ij}^{(t)} = 1$
9:             else
10:                $\phi_{ij}^{(t)} = 0$
11:            end if
12:        end for
13:    else {Mark $i$ as "uncertain"}
14:        $\phi_{ik}^{(t)} = p_{ik}^{(t)} \quad \forall k \in \{1, .., C\}$
15:    end if
16: end for
17: Set $A = \left(\Phi^{(t)}\right)\left(\Phi^{(t)}\right)^T$
18: return $A$
```

5.2.2 Subspace Sparse Representation

Updating Z and E: Given the soft subspace segmentation matrix Φ and association matrix A, we update the coefficient matrix Z and error matrix E in the next step by solving the optimization equation (5.3) with respect to $\{Z, E\}$

$$\min_{Z,E} \left(\lambda_0 \|Z\|_1 + \frac{1}{2} \|E\|_F^2 + \lambda_1 \|(\mathbf{1} - A) * Z\|_F^2 \right) \tag{5.8}$$
$$\text{subject to} \quad E = X - XZ, \quad z_{ii} = 0$$

The first term in this equation formulates the sparsity of the self-representative model. The last term adds another error weight to the optimization formula when points are not in the same cluster but have strong pairwise connectivity. We solve this optimization equation using the Alternating Direction Method of Multipliers (ADMM) [5]. Alternating between updating matrices $\{Z, E\}$, and matrices $\{\Phi, A\}$, as explained, helps us remove small values in the sparse coefficient matrix Z and obtain a better pairwise representation of points with less noise/outliers, and hence a more accurate clustering result.

In the initial step of the proposed method, we set $A = \mathbf{1}^{N \times N}$. This is equivalent to removing the third term in the equation (5.3), which converts it to equation (2.5). We compute the coefficient matrix Z and error E using (2.5). Then, using spectral clustering [73], we divide the input data X into C clusters. By defining association degrees (5.5), we form the probability matrix P and compute the delayed parameter Ω as in formula (5.7). In the soft subspace segmentation matrix Φ, we divide the points into two groups of *certain* and *uncertain* points as in (5.6) and form the association matrix A. In each iteration, *certain* points give a better representation of points in their assigned subspace. *Uncertain* points get a chance of estimating a better representation before being assigned to any cluster. We alternate between optimizing with respect to $\{\Phi, A\}$ and $\{Z, E\}$. The proposed joint optimization method is summarized in algorithm 5. Given sparse similarity matrix $\bar{Z}$, we need to find the segmentation of points in each iteration. Previous methods (e.g. [47]) use normalized cuts and recompute the solution from scratch, with a time complexity of $O(N^{3/2})$ in the best case [27]. Our delayed association of points allows us to resort to an incremental spectral clustering algorithm [59] at a substantially lower computational cost, since the computed eigenvectors are updated when there are changes in the similarity matrix $\bar{Z}$. The running time of this clustering approach is close to $O(N)$ when every column of the coefficient matrix Z is sparse. In our approach, when *certain* points do not have any coefficients that connect them to *uncertain* points, they do not need to be updated in spectral clustering. Incremental clustering is applied to *uncertain* points and all other points that are connected to *uncertain* points in $\bar{Z}$. As a result of updating only *uncertain* points in the clustering matrix, the time complexity is drastically reduced. In section 5.4, we empirically show that the number of *uncertain* points, denoted as $\kappa(\Phi^{(t)})$, generally decreases, which implies that the cost of incremental updating itself is reducing at each iteration. In this notation $\Phi^{(t)}$ is the soft clustering matrix in iteration t.

Algorithm 5 Probabilistic Sparse Subspace Clustering Using Delayed Association

```
Input: $X \in \mathbb{R}^{n \times N}$, number of subspace $C$
1: Initialization: $A^{(0)} = \mathbf{1}^{N \times N}$
2: repeat
3:     Update $Z^{(t)}$: $\min_{Z,E} \left( \lambda_0 \|Z\|_1 + \frac{1}{2} \|E\|_F^2 + \lambda_1 \|(\mathbf{1} - A) * Z\|_F^2 \right)$
4:     Set $\bar{Z}^{(t)} = \frac{1}{2} \left[ Z^{(t)} + (Z^T)^{(t)} \right]$
5:     Find clusters by incremental spectral clustering[59] or re-initialize by spectral clustering. [73]
6:     Update $\Phi^{(t)}$ using Algorithm 4
7:     Set $A^{(t)} = (\Phi^{(t)})(\Phi^{(t)})^T$
8: until convergence
```

5.3 Parameter Analysis and Stopping Criterion

5.3.1 Synthetic Data

As pointed out earlier, we expect that a delayed probabilistic association in subspace clustering is better suited to handle subspaces with large intersections and overlaps, i.e. those with many points that belong to more than one subspace. This is of practical interest, since in most real data there is either ambiguity due to noise or significant similarity of points in different clusters, due to nested subspaces. In this section, generate a synthetic data to experimentally study the role of different parameters in the proposed method.

To generate our dataset, we used the method described in [75]. We examine the effect of intersection between subspaces when each subspace $\{S_j\}_{j=1}^{C}$ has a true dimension of $d = 10$ in $\mathbb{R}^n$ with an ambient dimension $n = 200$, and an intersection dimension of s (i.e. sharing s basis vectors). The first subspace S_1 of dimension d is generated uniformly at random. To generate each of the remaining subspaces for each subspace $\{S_j\}_{j=2}^{C}$, we generated two sets of basis: (i) the intersection basis $S_j^{(1)}$ of dimension s where $s < d$. (ii) the disjoint basis $S_j^{(2)}$ with dimension $d - s$. Then, the basis for each of the remaining subspaces (S_j, $j = 1..C$) are formed as $S_j = S_j^{(1)} \cup S_j^{(2)}$. We generated three different models by varying the number of subspaces C from two to four and samples $N_j = 100$ points uniformly at random from each subspace. We generated 20 instances from each of these models and changed the ratio of the intersection between subspaces in the range $\frac{s}{d} = \{40\%, .., 90\%\}$.

5.3.2 *Role of Delayed Association Matrix*

The assignment matrix Φ splits the points into two groups of *certain* and *uncertain* and uses the combination of both hard and soft clustering in constructing association matrix A. The third term, $\|(1-A)*Z\|_F^2$ in equation (5.8), impacts the search for the sparse self-representation matrix Z by bringing the effect of association between points in learning Z. On the other hand, the self-representation matrix Z is used for estimating the association between points given in A. More specifically, this joint optimization framework, learns the self-representation matrix and corrects soft clustering results Φ in an alternating manner. In each iteration:

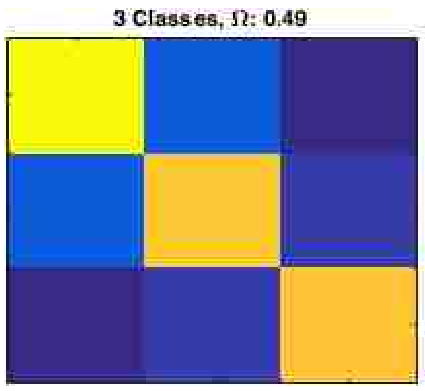
(i) For each pair of points x_i and x_j grouped as certain, when they are not segmented in the same subspace, the term $\|(1-A)*Z\|_F^2$ in (5.8) helps to sparsify column matrix Z by pushing coefficients z_{ij} and z_{ji} closer to zero.

(ii) For each pair of points x_i grouped as uncertain, term $\|(1-A)*Z\|_F^2$ in (5.8) keeps the probability of using other data points in representing x_i. This provides an opportunity to search for the best subspace based on the probability of including other points in the representation of x_i. In other words, the association of points to any subspace is delayed until later by giving the entire dataset an opportunity to participate in the representation of the uncertain point x_i.

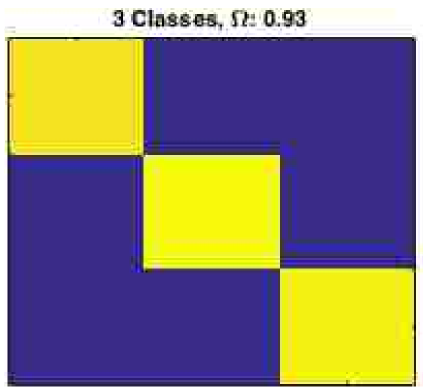
In each iteration, certain points help to give a better understanding of subspaces. This can help to obtain a better representation of uncertain points and identify their subspace more accurately. As it is illustrated in the experimental results, as the iteration continues, more points are grouped as certain which helps to obtain a better self-representation matrix Z.

5.3.3 *Role of Delayed Association Parameter*

The delayed association parameter Ω controls the ratio of uncertainty in matrices Φ and A using the similarity between points of the same subspace and affinity between different subspaces coded in matrix M. In early iterations, the coefficient matrix Z is not influenced significantly by the segmentation results in A. That means that Z is more ambiguous in terms of correct subspace, which results in a more uncertain computed probability matrix P. This high ambiguity between clustered points turns matrix M into a matrix with similar entries and consequently a smaller delayed association parameter Ω. A smaller Ω groups more points as uncertain where it delays the assignment of points to a subspace. As we move towards a more accurate self-representative matrix Z, affinity matrix M moves closer to the identity matrix which causes the delayed association parameter Ω to be closer to one and allows more points to be grouped as certain. This strategy allows uncertain points to be drawn closer to the correct subspaces before final assignments are made. Figure 5.1 illustrates matrix M for 3 classes. In Figure 5.1a there is a high ambiguity between clustered points. Thus, matrix M has similar entries. Matrix M in Figure 5.1b is close to the identity matrix and shows a case with low affinity between points of different subspaces and strong connectivity among points of a subspace.



(a)



(b)

Figure 5.1: Affinity between different subspaces coded in matrix M when there are 3 subspaces.

Figure 5.2 shows the result of changes in Ω over 10 iterations for 3 different cases of 2, 3 and 4 classes and 50% intersection. As it is shown, in all these graphs, the value of Ω increases as iterations continue. This increase shows that the off-diagonal entries on matrix M are being reduced. This means the affinity between points of different subspaces decreased.

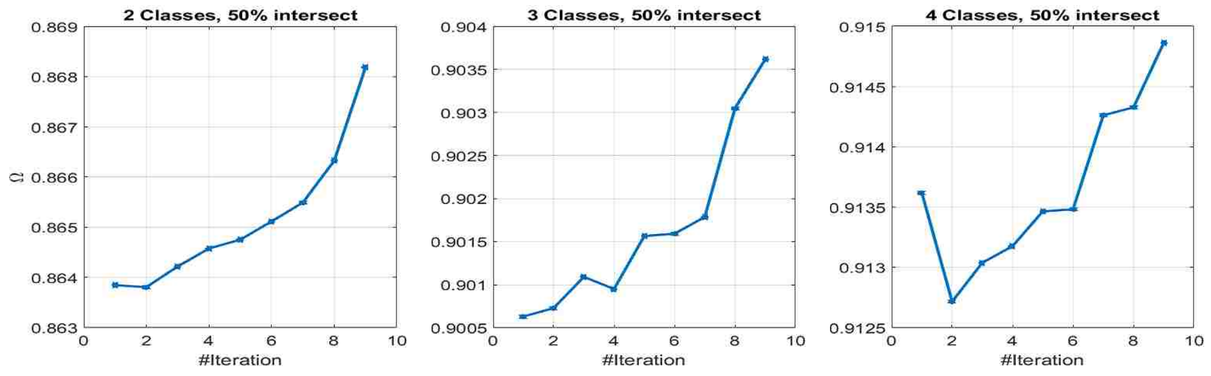


Figure 5.2: Updates in Ω over 10 iterations for 3 different cases of 2, 3 and 4 classes and 50% intersection.

As the percentage of intersection and number of classes increases, the ratio of points that stay in the intersection of subspaces grows. That increases the ambiguity of assigning points to clusters. In these cases, the delayed Association Parameter Ω might decrease which means that there a lot of ambiguity between different subspaces and more points remain as uncertain. Figure 5.3 shows 3 different cases of 2, 3 and 4 classes and 70% intersection. As illustrated, the delayed Association Parameter Ω decreases when the data includes 4 classes with 70% intersection.

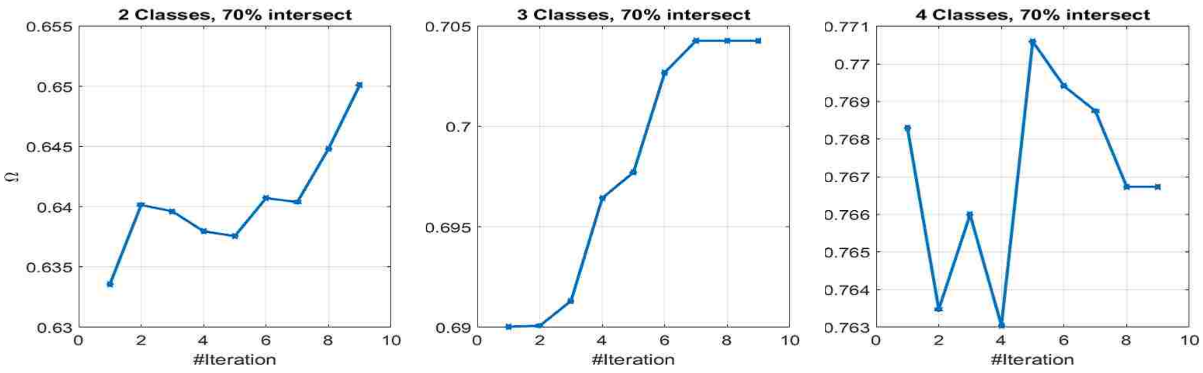


Figure 5.3: Updates in Ω over 10 iterations for 3 different cases of 2, 3 and 4 classes and 70% intersection.

5.3.4 Role of Regression Parameters

The two terms $\lambda_0 \|Z\|_1$ and $\lambda_1 \|(\mathbf{1} - A) * Z\|_F^2$ in formula (5.8) are participating in the search for the most accurate self-representation matrix Z, The ratio of parameters λ_0 and λ_1 in (5.8) implies how much the association matrix A impacts constructing the representational matrix Z. Particularly, $\lambda_0 \ll \lambda_1$ increases the effect of the segmentation results from the previous iteration (matrix A) on constructing the coefficient matrix Z. On the other hand, when $\lambda_0 \gg \lambda_1$ the effect of association matrix A will be very small and the optimization formula (5.8) approaches (2.5). In this dissertation, we choose the value of λ_0 and λ_1 empirically. We change the ratio of $\frac{\lambda_0}{\lambda_1}$ between 0.1 to 10 based on the percentage of intersection or noise/outliers in the dataset.

5.3.5 Stopping Criterion

The proposed optimization framework in 5.3 is not convex with respect to A. Thus, we use spectral clustering techniques to estimate the subspace segmentation and association matrix A. This can cause the method to converge to a global or local optimum answer. Therefore, we need to add a stopping criterion of maximum iteration t_{max} to avoid infinite loops in algorithm 5.

Another stopping criterion is defined as a decrease in the number of *uncertain* points. In the experimental results section 5.4, we show that the number of uncertain points $\kappa(\Phi^{(t)})$ generally decreases in each iteration. Thus, we can stop algorithm 5 when the number of uncertain points does not decrease.

$$\kappa\left(\Phi^{(t)}\right) - \kappa\left(\Phi^{(t-1)}\right) \leq 0, \tag{5.9}$$

where $\kappa\left(\Phi^{(t)}\right)$ is the number of *uncertain* points in iteration (t) and $\kappa\left(\Phi^{(t-1)}\right)$ is the number of *uncertain* points in iteration $(t-1)$. An example of updates in association matrix A and similarity matrix $\bar{Z}$ over 2 sequential steps is illustrated in Figure 5.4. The data includes 3 subspaces and 70% intersection. As it is shown, $\bar{Z}$ became more sparse and close to block-diagonal in the second iteration. association matrix A also could derive a better representation of clusters.

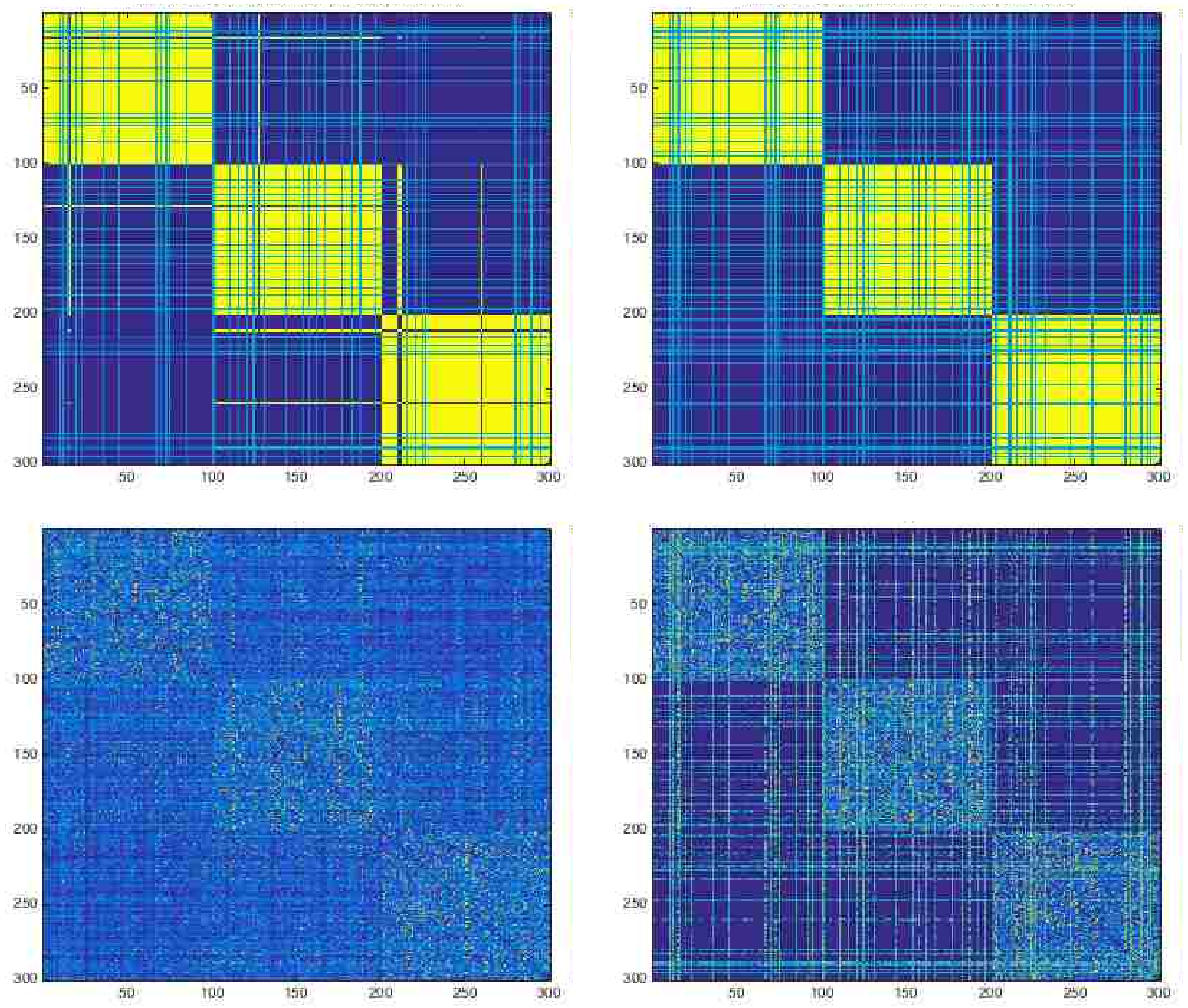


Figure 5.4: Updates in A (top) and $\bar{Z}$ (bottom) over two consecutive iterations. Data includes 3 subjects and 70% intersection.

5.4 Experimental Results

5.4.1 Error Metrics

In order to evaluate the accuracy of the proposed subspace clustering method, we used different metrics. The first metric we used is a direct measure of the misclassification error of sparse subspace clustering results. This is defined as:

$$Err = \frac{\#\text{Misclassified Points}}{N} \tag{5.10}$$

This is the total number of incorrectly clustered points over the total number of points in the population. A second metric (equation (5.11)) measures the misclassification error of subspace clustering using only the points that are not on the intersection of subspaces, i.e. with a high probability of belonging to only one subspace.

$$Err_{\hat{\kappa}} = \frac{\#\text{Misclassified Points}_{\hat{\kappa}}}{N_{\hat{\kappa}}} \tag{5.11}$$

where $\#\text{Misclassified Points}_{\hat{\kappa}}$ is the number of points that are not on the intersection of subspaces and are clustered incorrectly, and $N_{\hat{\kappa}}$ is the total number of points that are not on subspace intersections. Another metric used is the subspace sparse recovery error [75]. This metric computes the error of representing each point in its final cluster according to the coefficient matrix Z. The columns of z_i determine all the coefficients to self-represent the point x_i using all the other points in the dataset. Using the result of the clustering algorithm, each column z_i gets divided into C classes $\delta_{s_1}(z_i), \delta_{s_2}(z_i), ..., \delta_{s_C}(z_i)$, where each of the $\delta_{s_k}(z_i)$ are the coefficients of representing x_i in cluster k. Assuming m is the correct cluster for point x_i, and that $\delta_{s_m}(z_i)$ are the corresponding coefficients to reconstruct the point, the average subspace sparse recovery error can be defined as:

$$SSR = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{\|\delta_{s_m}(Z_i)\|_1}{\|Z_i\|_1} \tag{5.12}$$

In the following sections, we study the accuracy of the proposed method for both synthetic and real datasets, and compare the results with state-of-the-art algorithms. In our experiments, the value of $t_{\max} = 10$ and the initial value of the ratio of $\frac{\lambda_0}{\lambda_1} = 10$.

5.4.2 Synthetic Data

In this section, we experimentally study the accuracy of the proposed method using the generated synthetic data explained in section 5.3.1.

Clustering Accuracy: In the first experiment, we examined the convergence of the algorithm, i.e. the clustering misclassification versus the number of iterations. We show the result for 50% and 90% intersections and for two to four clusters. The average number of iterations t before reaching the condition in algorithm 5 is $t = 5$. The complete summary of average misclassification error over 20 independent trials is shown in table 5.1. As shown in this table, we compare the misclassification error of the proposed method with SSC [20] as a baseline and S^{3}C [47] as the state-of-the-art. The proposed delayed association method consistently outperforms both the baseline and the state-of-the-art methods.

Table 5.1: Intersection %misclassification as a function of subspaces intersection and number of subspaces.

#Subspace	2					3					4				
%intersect	50%	60%	70%	80%	90%	50%	60%	70%	80%	90%	50%	60%	70%	80%	90%
SSC	0.10	0.50	2.68	8.0	33.8	0.13	1.20	3.98	13.3	42.0	0.31	1.30	6.19	18.4	50.9
S^{3}C	**0.03**	0.40	2.38	7.63	30.9	0.15	1.10	3.98	13.3	42.0	0.30	1.23	5.94	18.1	50.5
Prob SSC	**0.03**	**0.25**	**1.58**	**7.25**	**24.9**	**0.08**	**0.65**	**3.23**	**11.7**	**34.8**	**0.11**	**0.66**	**4.41**	**14.4**	**42.4**

Additionally, table 5.2 shows a complete summary of average representation errors (equation (5.12)) on the same experiment. As seen in this table, our method computes a sparser matrix Z compared with SSC. However, in some cases, it may compute a less sparse matrix Z compared with S^{3}C due to delayed association. When there is more ambiguity in the clusters, our method keeps more points in the *uncertain* group, making the coefficient matrix Z temporarily less sparse compared with S^{3}C. However, this helps keep the classification error lower (see Table 5.1).

Table 5.2: Intersection %SSR error as a function of subspaces intersection and number of subspaces.

#Subspace	2					3					4				
%intersect	50%	60%	70%	80%	90%	50%	60%	70%	80%	90%	50%	60%	70%	80%	90%
SSC	0.09	0.17	0.20	0.29	0.38	0.12	0.18	0.28	0.36	0.53	0.13	0.26	0.28	0.49	0.60
S^{3}C	**0.04**	**0.10**	**0.14**	**0.21**	0.39	0.08	**0.09**	**0.22**	**0.27**	0.48	**0.05**	**0.16**	**0.17**	**0.42**	**0.57**
Prob SSC	**0.04**	**0.10**	0.17	0.25	**0.36**	**0.04**	**0.09**	**0.22**	0.32	**0.46**	0.06	0.19	0.19	0.44	**0.57**

Increasing the intersection between subspaces, rises the ambiguities in finding the correct segment for points. In this experiment, we examine the misclassification error of proposed method as the percentage of intersection between subspaces increases. Figure 5.5 shows the average misclassification error (equation (5.10)) as the intersection between subspaces increments. The average is over 20 independent trials. We compared the results of the proposed method with SSC [20] and S^{3}C [47]. As illustrated, the proposed method is more accurate compared with state-of-the-art algorithms.

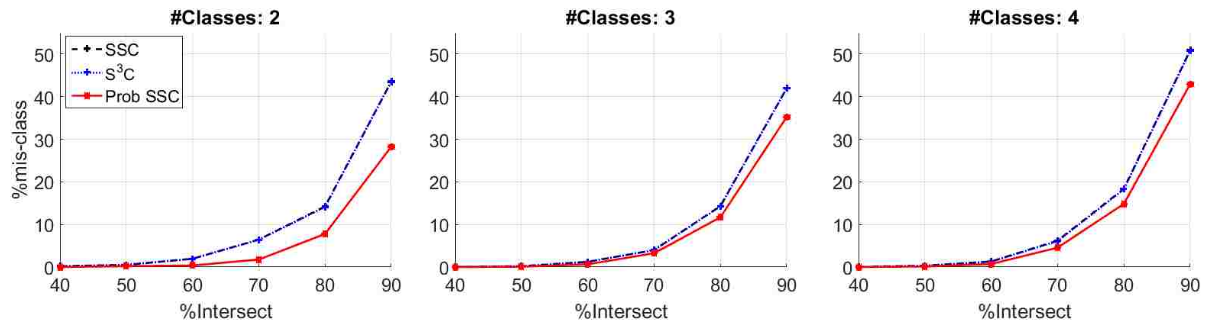


Figure 5.5: Average $\%$ misclassification errors (5.10) for Prob-SSC, SSC and S^{3}C methods over 20 independent trials.

The joint optimization algorithm, proposed in this chapter, alternates between searching for the best self-representation matrix $\{Z, E\}$ and best association matrix A in which we expect to boost the accuracy as iteration continues. Graphs in Figure 5.6 illustrate the average changes in misclassification of subspace clustering over the iterations. These graphs compare the accuracy of the proposed algorithm with the approaches in S^{3}C [46] and in SSC.

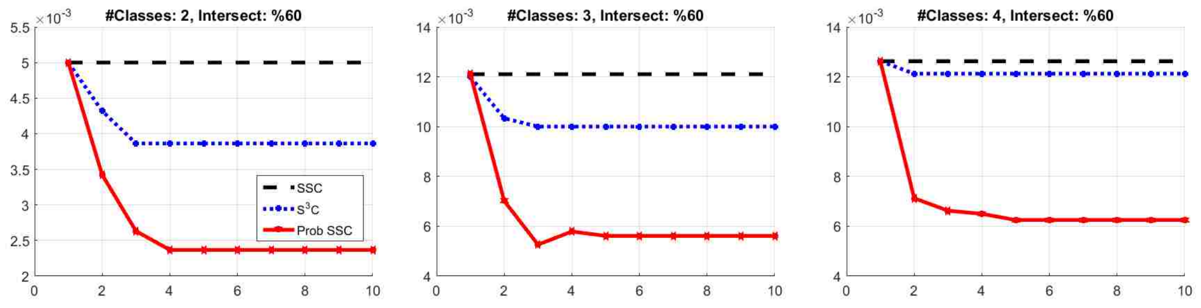


Figure 5.6: Average %misclassification errors over 10 iterations. Results are shown for Prob-SSC, SSC and S^{3}C.

To show that the percentage of uncertain points ($\kappa(\Phi)$) decreases and leads to converge the proposed method, we removed constraints of $\Phi^{(t)} = \Phi^{(t-1)}$ or $\kappa\left(\Phi^{(t)}\right) \geq \kappa\left(\Phi^{(t-1)}\right)$ in algorithm 5 and repeated the process for 10 iterations. Figure 5.7 shows the average percentage of uncertain points decreases generally. The experiment is on 50% intersection between subspaces over 20 independent trials.

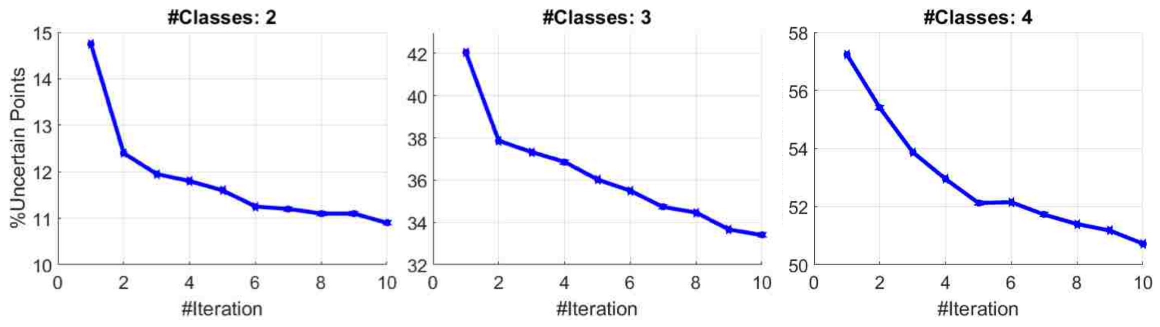


Figure 5.7: Average decrease of %*uncertain* points.

Points on intersections: In a dataset with intersecting subspaces, each subspace may contain many points that are on the intersection of subspaces. These points are naturally more challenging to cluster, particularly when they are equally close to intersecting subspaces, and in the presence of noise. In this section, we analyze the accuracy of the proposed subspace clustering method on clustering *certain* and *uncertain* points as described earlier. Table 5.3 shows the percentage of points that are found approximately on the intersection of subspaces. The table shows the value for different number of subspaces and percentage of intersections (note that points are generated randomly). As expected, by increasing the number of subspaces C and percentage of overlap, a larger number of points fall on the intersection of subspaces.

Table 5.3: Average $\%$ of points falling on the intersection of subspaces in the synthetic data.

	Intersection	40%	50%	60%	70%	80%	90%
#Classes	2	0.13	0.47	1.73	4.93	11.1	25.8
	3	0.26	0.53	1.91	5.94	13.0	31.9
	4	1.00	1.48	2.15	6.68	14.6	34.5

To show how points farther from subspace intersections are clustered in the proposed method, We next examine the misclassification error of the proposed subspace clustering in clustering points that are nor falling the intersections. Graphs in Figure 5.8 shows the misclassification errors for different number of classes as the percentage of subspace intersection increases. We compare the misclassification error for points that are not falling on the intersection of subspaces and the entire data set. As illustrated, the misclassification error Err_{κ} for points outside of the intersection of subspaces is significantly lower compared to the misclassification error of the entire population Err. We also added the percentage of the points $\frac{N_{\hat{\kappa}}}{N}\%$ in these graphs.

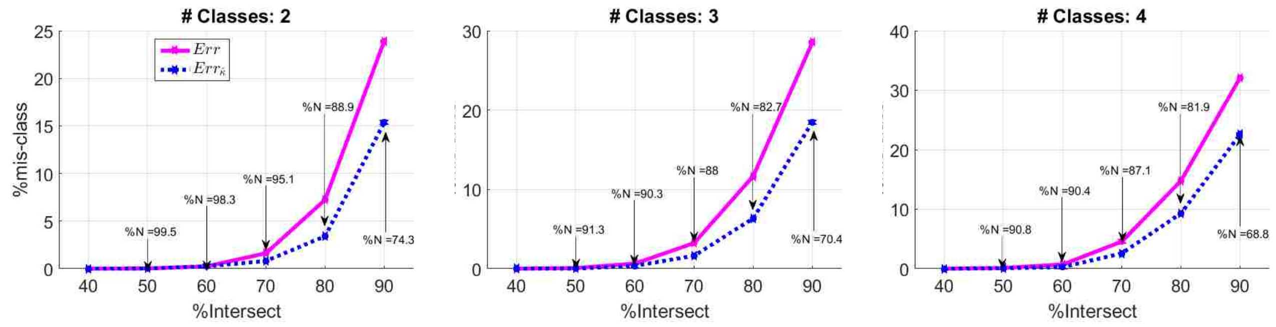


Figure 5.8: Average %misclassification errors for points that are not falling on the intersection of subspaces.

Table 5.4 shows the average percentage of points remaining in the *uncertain* group using the proposed algorithm. The table shows the value for different number of subspaces and percentage of intersections. As expected, by increasing the number of subspaces C and percentage of overlap, a larger number of points remains *uncertain*.

Table 5.4: Average $\%$ of points classified as *uncertain* in the proposed Prob-SSC algorithm.

	Intersection	40%	50%	60%	70%	80%	90%
#Classes	2	0.40	0.65	0.95	5.85	8.23	9.10
	3	0.61	0.72	3.10	9.97	12.8	16.7
	4	0.73	0.87	3.45	12.1	16.9	19.7

5.4.3 Face Clustering With Illumination and Pose Variation

Face classification is one of the many applications of subspace clustering. Face image classification techniques try to cluster images of the same subject under varying illumination or imaging conditions in one group. Studies in [44, 2, 32] show that a set of images of an object under varying illumination lies in a low-dimensional linear subspace of the image space of up to nine dimensions. Thus, face images in this condition can be clustered using subspace clustering techniques. The Extended Yale B Database [26] is a facial dataset widely used in subspace clustering literature [21, 64, 96] and contains $2,414$ frontal face images of 38 human subjects taken under approximately 64 different illumination conditions. This dataset is considered as a challenging one for clustering techniques due to its extreme lighting variations (Figure 5.9)

Figure 5.9: Sample images in YaleB Dataset

In this experiment, we used the proposed algorithm to study the improvement in accuracy of face clusters compared to previous methods. We used subsets of $C = \{2, 3, 5, 8, 10\}$ different subjects (subspaces) from the dataset. Each subject includes 64 images ($N_j = 64$). Each downsampled image has a dimension of 48×42 that is vectorized to a 2016-dimensional vector. These 38 subjects are divided in 4 groups of 1-10, 11-20, 21-30 and 31-38 [46]. Similar to the reported results in [46], we kept the original size of the image vectors (2016) and reported the result on the whole dataset. We examined the accuracy of the proposed subspace clustering algorithm on this dataset. An example of updates in association matrix A and similarity matrix $\bar{Z}$ on YaleB database is illustrated in Figure 5.10. The data inclues $C = 5$ subspaces (subjects). As shown in this figure, the percentage of *uncertain* points are decreased from $\%\kappa_1 = 30\%$ in the first iteration to $\%\kappa_2 = 5\%$ and $\%\kappa_3 = 1\%$ in the second and third iterations, respectively. Also, the misclassification errors of subspace clustering are decreased from 7.19% to 0.1%.

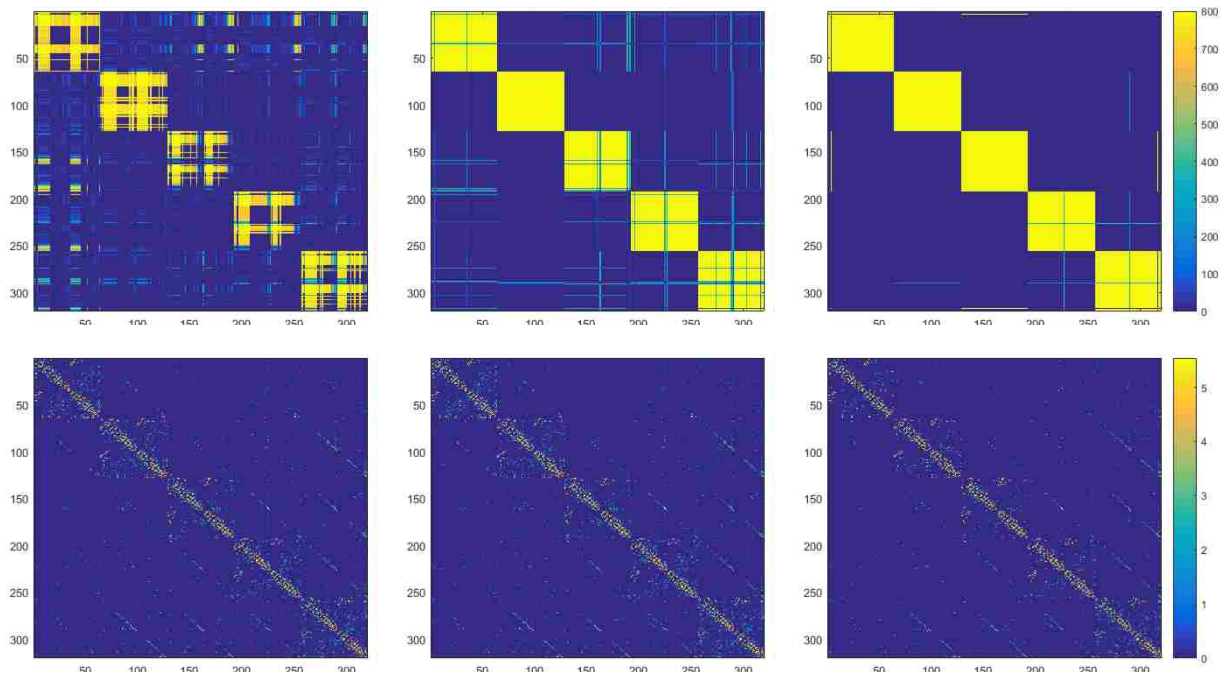


Figure 5.10: Updates in A (top) and $\bar{Z}$ (bottom) over three consecutive iterations. Data is from the Yale B Face Dataset with 5 subjects. The misclassification errors are 7.19%, 1.25% and 1.0% in iterations $t = \{1, 2, 3\}$ respectively.

Table 5.5 shows the clustering error percentages for the proposed algorithm. We compared the error of our proposed method with S^{3}C [47], SSC[21], LRR [48], LatLRR [49] and LRSC [86, 22]. We cite the reported results in [86, 49] and [47] in this table. As it is shown in this table, our method outperforms all other state-of-the-art algorithms.

Table 5.5: Average %misclassification errors on Extended Yale B Dataset [26].

# Subjects	2		3		5		8		10	
%Err	Average	Median	Average	Median	Average	Median	Average	Median	Average	Median
LRR	6.74	7.03	9.30	9.90	13.94	14.38	25.61	24.80	29.53	30.00
LRSC	3.15	2.34	4.71	4.17	13.06	8.44	26.83	28.71	35.89	34.84
LatLRR	2.54	0.78	4.21	2.60	6.90	5.63	14.34	10.06	22.92	23.59
SSC	1.87	0.00	3.35	0.78	4.32	2.81	5.99	4.49	7.29	5.47
$S^3C^\dagger$ [46]	1.43	**0.00**	3.09	**0.52**	4.08	2.19	4.84	4.10	6.09	5.16
$S^3C^\ddagger$ [46]	1.40	**0.00**	3.08	**0.52**	3.83	1.88	4.45	3.52	5.42	4.53
S^3C	1.27	**0.00**	2.71	**0.52**	3.41	1.25	4.15	2.93	5.16	4.22
Soft $S^3C^\dagger$	0.52	**0.00**	0.89	**0.52**	1.51	1.25	2.31	2.25	2.81	2.50
Soft $S^3C^\ddagger$	0.51	**0.00**	0.94	**0.52**	1.65	1.25	2.54	2.34	2.92	2.97
Soft S^3C	0.76	**0.00**	0.82	**0.52**	1.32	1.25	2.14	1.95	2.40	2.50
Prob SSC	**0.48**	**0.00**	**0.77**	**0.52**	**1.23**	**0.93**	**2.08**	**1.26**	**2.14**	**2.19**

5.4.4 Motion Segmentation

Motion segmentation of trajectory data has been widely studied in computer vision applications such as reconstructing dynamic scenes [101]. In dynamic scenes with multiple moving rigid objects, the trajectories can be represented by high-dimensional vectors. Yet, they can span low-dimensional linear manifolds [101]. It is shown that, in an affine subspace, [80] trajectories of a single motion in an ambient dimension $\mathbb{R}^{2F}$, where F is the number of frames, lies in a low-dimensional linear subspace of up to four dimensions. Thus, subspace clustering algorithms can be used to cluster the trajectories of different motions in separate subspaces. In this part, we used Hopkins 155 motion segmentation dataset [82] to examine the proposed subspace clustering method. Hopkins 155 motion database contains 156 sequences, each of which has 39550 data points drawn from two or three independent motions. Figure 5.11 shows examples from this dataset.

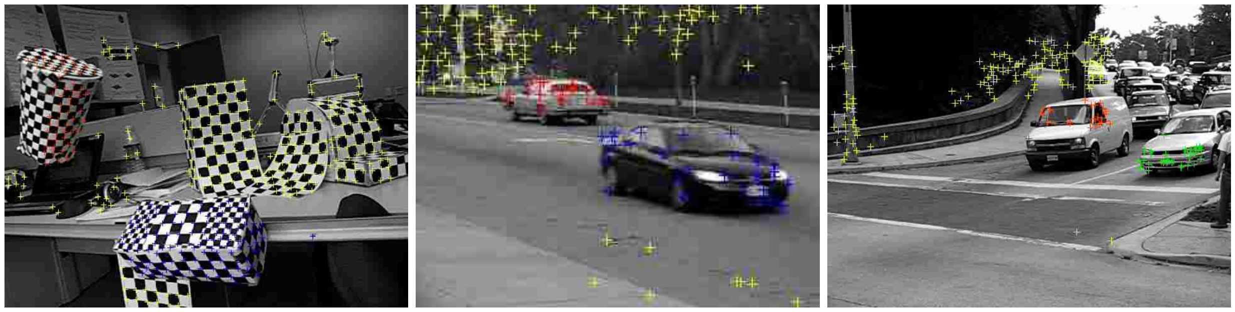


Figure 5.11: Images from the Hopkins 155 dataset.

Figure 5.12, shows updates in association matrix A and similarity matrix $\bar{Z}$ on the Hopkins 155 database with $C = 2$ subspaces (subjects). As illustrated, coefficients associated with wrong subspaces are removed from the sparse similarity matrix $\bar{Z}$ during iterations. The percentage of *uncertain* points is decreased from $\%\kappa_1 = 10\%$ in the first iteration to $\%\kappa_2 = 2\%$ and $\%\kappa_4 = 0.0\%$ in the second and fourth iterations, respectively. Also, the misclassification errors of subspace clustering are decreased from 2.42% to 0.43%.

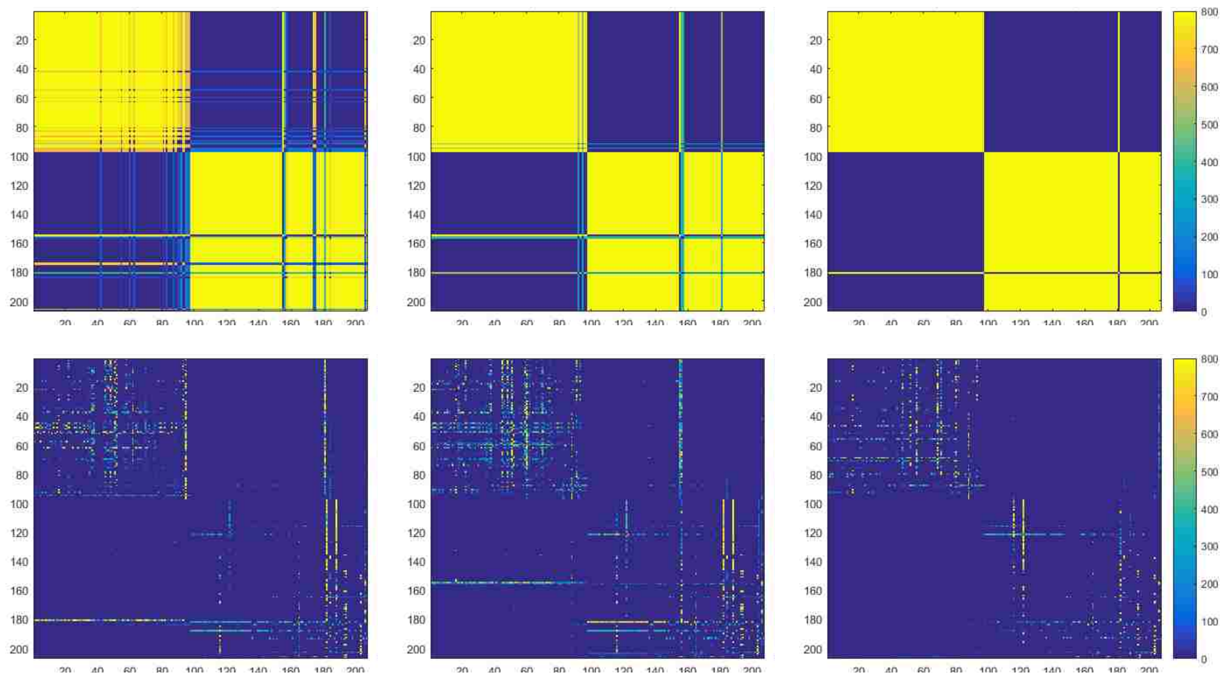


Figure 5.12: Updates in A (top) and $\bar{Z}$ (bottom). Data is from the Hopkins155 with 2 subjects. The percentage of *uncertain* points ar 10%, 2% and 0.0% and misclassification errors are 2.42%, 1.93% and 0.48% in iterations $\{1, 2, 4\}$ respectively.

We compared the misclassification errors of our proposed method on this dataset with LRR[48], LRSC [86], SSC [20], and S^{3}C [46]. Table 5.6 shows the clustering error percentages for these methods. We cite the reported results in [46] in this table. As shown, our proposed method outperforms all other state-of-the-art algorithms.

Table 5.6: Average and median of %misclassification error on Hopkins dataset.

	2 Motions		3 Motions		Total	
%ERR	Average	Median	Average	Median	Average	Median
LRR	3.76	**0.00**	9.92	1.42	5.15	**0.00**
LRSC	2.57	0.00	6.62	1.76	3.47	0.09
LatLRR	—-	—-	—-	—-	2.95	5.86
SSC	1.95	**0.00**	4.94	0.89	2.63	**0.00**
$S^3C^\dagger$	1.68	**0.00**	5.26	0.81	2.49	**0.00**
$S^3C^\ddagger$	1.73	**0.00**	5.29	0.81	2.53	**0.00**
S^3C	1.73	**0.00**	5.50	0.81	2.58	**0.00**
Soft-$S^3C^\dagger$	1.60	**0.00**	4.27	0.73	2.20	**0.00**
Soft-$S^3C^\ddagger$	1.64	**0.00**	4.11	0.73	2.20	**0.00**
Soft-S^3C	1.65	**0.00**	4.27	0.61	2.24	**0.00**
Prob SSC	**1.57**	**0.00**	**3.60**	**0.58**	**2.13**	**0.00**

5.5 Discussion and Conclusion

In this chapter, we introduced a new subspace clustering method that outperforms state-of-the-art methods reported recently in the literature. This boost in accuracy is caused by replacing the usual hard clustering matrix with an *association matrix* A that allows us to track the assignment of points in the same clusters, and hence delay hard assignments until later iterations, when more confidence is gained. This is possible because, at each iteration, the method splits the points into two groups of *certain* and *uncertain*, allowing the latter group's association to be delayed until later iterations when the association probabilities become higher. A direct advantage of this delayed association is that the method performs better when subspaces are highly overlapping (i.e. high intersection of bases). The results on both synthetic and real data confirm these advantages. Using incremental spectral clustering can reduce the time complexity because we can apply it to *uncertain* points and their connections to the rest of the data. However, when we have a high rate of *uncertain* points, we need to re-initialize the clusters using classical spectral clustering methods to prevent growth of inaccuracy.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this dissertation, we address the two fundamental problems of high volume and high dimensionality of big data. In a wide range of applications in computer vision, many tasks have been introduced in understanding the underlying patterns and trends in data. The need for more sophisticated data analysis methods has increased significantly due to the vast growth of data in terms of volume and dimension. We present a broad study of big data analysis in multi-structural environments in the field of computer vision, using novel approaches to sparse sampling and dimensionality reduction.

In the first part of the dissertation, Chapter 3, we address the problem of high-volume data by sparse sampling. Sparse sampling is the process of determining the smallest sample size needed to guarantee sufficient points for a faithful representation of the entire dataset, when the data includes multiple structures. To address this problem, we propose a method referred to as the Sparse Withdrawal of Inliers in a First Trial (SWIFT). The proposed method determines the smallest required number of points to be sampled in a one-time grab and provides a sufficient number of points to represent the entire population in the presence of a large number of outliers. We show that the problem can be modeled using either a hypergeometric or a multinomial probability mass function (pmf), and derive accurate mathematical bounds to calculate a tight approximation of the sample size, leading to a sparse sampling strategy. The key features of the proposed method are: (i) the sparsity of the sampled subset for analyzing data, where the level of sparsity is independent of the population size; (ii) no prior knowledge of the distribution of data, or the number of underlying group structures in the data; and (iii) robustness in the presence of an overwhelming number of outliers. We include a comprehensive evaluation of the method in terms of its accuracy and its behavior under different parameters and its effectiveness in reducing the computational cost in various applications of computer vision, such as data clustering, model fitting, and multibody structure from motion.

In the second part of this dissertation, we address the problem of high dimensionality in multi-structural environments. Dimensionality reduction is another tool in analyzing big data in a multi-structural environment, when the data resides in a high-dimensional ambient space. Subspace clustering is the process of reducing the dimension of data by identifying and grouping points into subspaces of lower dimension. Unlike classical clustering methods where points in a metric space are grouped based on a distance measurement, the challenge in subspace clustering is that points can be arbitrarily close and yet belong to separate subspaces. In Chapter 4, we studied the effect of using SWIFT sampling to address large scale high dimensional data. The proposed method in this chapter assumes that the structure of subspaces in the entire population can be learned from a small portion of data. We start with a small portion of data points selected as in-sample data to learn the structure of all subspaces in the population. Later, out-of-sample data points are assigned to the best subspace that represents them using a defined residual error. We show that SWIFT sampling can estimate a very accurate sample size when it is sampled uniformly at random from high dimensional vectors. These sampled points are used to learn the bases of subspaces before assigning out-of-sample data to the learned subspaces. We evaluated the accuracy and behavior of the method and its effectiveness in reducing the computational complexity of the process under different scenarios. We also show the result on face clustering as an example of a real application in computer vision.

In Chapter 5, we propose a Probabilistic-Based Sparse Subspace Clustering (ProbSSC) algorithm, in which we try to jointly search for a lower dimensional representation of points and segment them into separate subspaces. Earlier subspace clustering methods divided the problem into two separate stages: (i) finding a pairwise similarity between points in terms of bases representations and (ii) clustering data into subspaces. In this chapter, we propose a new algorithm to capture the relationship between the representations of the points and their segmentation by integrating these two steps using an alternating optimization approach. This method delays the assignment of points with low confidence to clusters until their subspace association certainty improves. This is employed by introducing a joint optimization scheme that incorporates soft and hard clustering techniques in assigning points to clusters. We demonstrate that delayed association is better suited for clustering subspaces that have ambiguities, i.e. when subspaces intersect or when data is contaminated with outliers/noise. We demonstrate experimentally that such delayed probabilistic association leads to more accurate self-representation and final clusters. To support the idea of the proposed method and to measure its effectiveness, we performed extensive experiments with a variety of challenging data and compared our method with several competitive baselines.

6.2 Future Work

One problem that needs further investigation for sparse sampling in big data analysis is the extension of SWIFT to a hierarchical sampling technique where structures and their parameter can be estimated in a multi-level approach. Note that this idea of multi-level sampling is not the same as sequential sampling, since at each level all the subspaces are sought to be determined. We experimented with this idea in one of the example applications, and the results indicated that this divide-and-conquer approach can further reduce the time complexity of solving these problems by sampling. Thus, studying the hierarchical solution in detecting models in a multi-structural environment is a potential area for future study. Moreover, the estimated sample size in the proposed sampling method (SWIFT) assumes that all structures in the population are the same size. To obtain a more accurate estimation of sample size in big data with multiple structures and different proportions, one can study the extension of SWIFT in handling models with different sizes.

In the second section of this dissertation, we study the effect of dimensionality reduction in a multi-structural environment to represent points in lower dimension and segment them into separate subspaces. One limitation of classical subspace clustering algorithms is that shallow data is often used to search for the best self-representation of data where it is unable to capture latent structures of complex data. Moreover, these solutions are heavily reliant on the linearity of input data. This assumption is violated in more complex data where nonlinear functions are more suitable to capture the relationship between data points. In the literature, neural networks are identified as a solution to model the nonlinearity of data and extract low dimensional features. Recent studies in this area try to adapt deep learning in extracting latent space of data in an unsupervised manner. Auto-encoder, for instance, is used to understand partially labeled or unlabeled data. As a future work, one can study how conventional auto-encoders (CAEs) can be used to capture latent-space structures in the data and learn bases for high dimensional subspaces. Additionally, as stated earlier, with sufficient number of points, one can learn the representation of the entire population in each subspace. In other words, the search for the sparse self-representation of points can be achieved using a sample subset. Therefore, a future study can be performed on the effect of using sparse sampling (SWIFT) to learn bases for each subspace and to estimate the low dimensional representation of points.

APPENDIX A: DERIVATIONS AND PROOFS

Proof of Theorem 1:

Denote $d = \frac{r}{C}$, so that r can be represented as $r = Cd$. Then, the probability of selecting less than ε points from the first model instance d_1 can be calculated as:

$$\Delta = \sum_{k=0}^{\varepsilon-1} P(d_1 = k) = \sum_{k=0}^{\varepsilon-1} \frac{\binom{\theta}{k}\binom{(C-1)\theta}{Cd-k}}{\binom{C\theta}{Cd}} = P(d_1 = 0)\left[1 + \sum_{k=1}^{\varepsilon-1} \frac{P(d_1 = k)}{P(d_1 = 0)}\right], \tag{A.1}$$

where $Cd - k = \sum_{i=2}^{C} d_i$. In order to find an upper bound for the right-hand side of the above equation, we first find an upper-bound for $P(d_1 = 0)$:

$$P(d_1 = 0) = \frac{\binom{(C-1)\theta}{Cd}}{\binom{C\theta}{Cd}} = \prod_{j=0}^{\theta-1} \frac{C\theta - Cd - j}{C\theta - j}. \tag{A.2}$$

If $(C-1)\theta \geq Cd$, then the right-hand side of equation (A.2) can be further simplified using the inequality $\frac{x-a}{y-a} \leq \frac{x}{y}$ if $a < x < y$:

$$P(d_1 = 0) \leq \prod_{j=0}^{\theta-1} \frac{C\theta - Cd}{C\theta} = \left(1 - \frac{r}{C\theta}\right)^{\theta} \leq e^{-r/C}, \tag{A.3}$$

yielding (3.8). Now, let us find an upper bound for the ratio $P(d_1 = k)/P(d_1 = 0)$, the exact value of which can be evaluated as follows:

$$\begin{aligned}
\frac{P(d_1 = k)}{P(d_1 = 0)} &= \frac{\binom{\theta}{k}\binom{(C-1)\theta}{Cd-k}}{\binom{(C-1)\theta}{Cd}} \\
&= \frac{(Cd)!\,[(C-1)\theta - Cd]!}{k!\,(Cd-k)!\,[(C-1)\theta - (Cd-k)]!} \prod_{j=1}^{k} (\theta - j + 1) \\
&= \binom{Cd}{k} \prod_{j=0}^{k-1} \frac{\theta - j}{(C-1)\theta - Cd + k - j} \\
&\leq \binom{Cd}{k} \left[\frac{\theta}{(C-1)\theta - Cd + k}\right]^{k}.
\end{aligned} \tag{A.4}$$

Combination of the last inequality, (A.1) and (A.3) complete the proof.

Proof of Theorem 2:

Let, as before, $d = r/C$. Note that the value of Δ' can be expressed as:

$$\begin{aligned}\Delta' &= \sum_{k=0}^{\varepsilon-1}\sum_{l=0}^{\varepsilon-1} \frac{\binom{\theta}{k}\binom{\theta}{l}\binom{(C-2)\theta}{Cd-k-l}}{\binom{C\theta}{Cd}} \\ &= P_0\left[1 + 2\sum_{k=1}^{\varepsilon-1}\frac{P(d_1=k, d_2=0)}{P_0} + \sum_{k=1}^{\varepsilon-1}\sum_{l=1}^{\varepsilon-1}\frac{P(d_1=k,d_2=l)}{P_0}\right] \\ &\equiv P_0[1+2\Delta_1+\Delta_2)],\end{aligned} \tag{A.5}$$

where $Cd - k - l = \sum_{i=3}^{C} d_i$, $\Delta_1 = P(d_1 = k, d_2 = 0)/P_0$, $\Delta_2 = P(d_1 = k, d_2 = l)/P_0$ and P_0 is defined in (3.11). The first component, P_0, due to $(C-2)\theta \geq Cd$, can be bounded above as follows:

$$\begin{aligned}P_0 &= \frac{\binom{(C-2)\theta}{Cd}}{\binom{C\theta}{Cd}} = \prod_{j=0}^{2\theta-1}\frac{C\theta - Cd - j}{C\theta - j} \\ &\leq \prod_{j=0}^{2\theta-1}\frac{C\theta - Cd}{C\theta} = \left(1-\frac{d}{\theta}\right)^{2\theta} \leq e^{-2d},\end{aligned} \tag{A.6}$$

which yields the right-hand-side of (3.11). The second component in (A.5) can be re-written as:

$$\begin{aligned}\Delta_1 &= \frac{P(d_1=k,d_2=0)}{P(d_1=0,d_2=0)} = \frac{\binom{\theta}{k}\binom{(C-2)\theta}{Cd-k}}{\binom{(C-2)\theta}{Cd}} \\ &= \frac{(Cd)!\,[(C-2)\theta - Cd]!}{k!\,(Cd-k)!\,[(C-2)\theta-(Cd-k)]!}\prod_{j=1}^{k}(\theta-j+1) \\ &= \binom{Cd}{k}\prod_{j=0}^{k-1}\frac{(\theta-j)}{(C-2)\theta - Cd + k - j}.\end{aligned} \tag{A.7}$$

This yields the following upper bound for Δ_1:

$$\Delta_1 \leq \binom{Cd}{k} \left[\frac{\theta}{(C-2)\theta - Cd + k}\right]^{\cdot} \tag{A.8}$$

Finally, Δ_2 in (A.5) can be expressed as:

$$\begin{aligned}\Delta_2 &= \frac{P(d_1 = k, d_2 = l)}{P(d_1 = 0, d_2 = 0)} = \frac{\binom{\theta}{k}\binom{\theta}{l}\binom{(C-2)\theta}{Cd-k-l}}{\binom{(C-2)\theta}{Cd}} \\ &= \frac{(Cd)!}{k!\; l!\; (Cd-k-l)!} \frac{[(C-2)\theta - Cd]!}{[(C-2)\theta - (Cd-k-l)]!} \\ &\times \prod_{j=1}^{k}\prod_{i=1}^{l}(\theta - j + 1)(\theta - i + 1) \\ &= \binom{Cd}{k,\; l} \frac{\prod_{j=0}^{k-1}(\theta - j)\prod_{i=0}^{l-1}(\theta - i)}{\prod_{u=0}^{k+l-1}[(C-2)\theta - Cd + u]} = \binom{Cd}{k,\; l} A_{kl}\end{aligned} \tag{A.9}$$

Now, denote $v = k + l - 1 - u$, so the A_{kl} can be written as follows:

$$\begin{aligned}A_{kl} &= \frac{\prod_{j=0}^{k-1}(\theta - j)\prod_{i=0}^{l-1}(\theta - i)}{\prod_{v=0}^{k+l-1}[(C-2)\theta - Cd + k + l - 1 - v]} \\ &= \frac{\prod_{j=0}^{k-1}(\theta - j)}{\prod_{v=0}^{k-1}[(C-2)\theta - Cd + k + l - 1 - v]} \times \frac{\prod_{i=0}^{l-1}(\theta - i)}{\prod_{v=k}^{k+l-1}[(C-2)\theta - Cd + k + l - 1 - v]}\end{aligned} \tag{A.10}$$

Using $j = v$ and $i = v - k$, this equation can be rewritten as:

$$\begin{aligned}A_{kl} &= \prod_{j=0}^{k-1}\left[\frac{(\theta - j)}{(C-2)\theta - Cd + k + l - 1 - j}\right] \times \prod_{i=0}^{l-1}\left[\frac{(\theta - i)}{(C-2)\theta - Cd + l - 1 - i}\right] \\ &\leq \left[\frac{\theta}{(C-2)\theta - Cd + k + l - 1}\right]^{k} \times \left[\frac{\theta}{(C-2)\theta - Cd + l - 1}\right]^{l}\end{aligned} \tag{A.11}$$

Since $l \geq 1$, this inequality yields:

$$A_{kl} \leq \left[\frac{\theta}{(C-2)\theta - Cd + k}\right]^k \left[\frac{\theta}{(C-2)\theta - Cd + l - 1}\right]^l \tag{A.12}$$

Using the inequalities in (A.6), (A.8), and (A.9) the probability of selecting less than ε points from the first and the second model instances d_1, d_2 can be bounded above as:

$$\Delta' \leq P_0 \times \left[1 + 2\sum_{k=1}^{\varepsilon-1} \binom{Cd}{k} \left[\frac{\theta}{(C-2)\theta - Cd + k}\right]^k \right. \tag{A.13}$$
$$\left. + \sum_{k=1}^{\varepsilon-1}\sum_{l=1}^{\varepsilon-1} \binom{Cd}{k,\ l} \left[\frac{\theta)}{(C-2)\theta - Cd + k}\right]^k \times \left[\frac{\theta}{(C-2)\theta - Cd + l - 1}\right]^l\right]$$

Using $k, l \geq 1$, and the following inequities:

$$\begin{gathered}\left[\frac{\theta}{(C-2)\theta - Cd + k}\right] \leq \left[\frac{\theta}{(C-2)\theta - Cd}\right], \\ \left[\frac{\theta}{(C-2)\theta - Cd + l - 1}\right] \leq \left[\frac{\theta}{(C-2)\theta - Cd}\right], \\ \binom{Cd}{k, 0} = \binom{Cd}{k}, \quad \binom{Cd}{0, 0} = 1,\end{gathered} \tag{A.14}$$

we can simplify (A.13) as follows:

$$\Delta' \leq P_0 \times \sum_{k=0}^{\varepsilon-1}\sum_{l=0}^{\varepsilon-1} \binom{Cd}{k,\ l} \left[\frac{\theta)}{(C-2)\theta - Cd}\right]^{k+l} \tag{A.15}$$

which completes the proof.

Proof of Theorem 3: Choosing

$$t = \frac{4}{3}\ln\left(\frac{C}{\delta}\right) + \sqrt{\frac{2r(C_1)}{C^2}\,\ln\left(\frac{C}{\delta}\right)} \tag{A.16}$$

obtain that

$$P\left(Bin\left(r, \frac{1}{C}\right) < \frac{r}{C} - t\right) \leq \frac{\delta}{C} \tag{A.17}$$

Now, in order $P(d_1 \leq \varepsilon - 1) = P(Bin(r, 1/C) \leq \varepsilon - 1) \leq \delta/C$ hold, we need to ensure that r is large enough, so that $r/C - t \geq \varepsilon - 1$. For this purpose we need to solve the inequality

$$\frac{r}{C} - \frac{4}{3}\ln\left(\frac{C}{\delta}\right) + \sqrt{\frac{2r(C_1)}{C^2}\,\ln\left(\frac{C}{\delta}\right)} \geq \varepsilon - 1 \tag{A.18}$$

for r. The inequality (A.18) is valid provided

$$\sqrt{r} \geq \sqrt{\frac{(C-1)}{2}\,\ln\left(\frac{C}{\delta}\right)} + \sqrt{\frac{(C-1)}{2}\,\ln\left(\frac{C}{\delta}\right) + C\left[\varepsilon - 1 + \frac{4}{3}\ln\left(\frac{C}{\delta}\right)\right]} \tag{A.19}$$

Since $(a+b)^2 \leq 2a^2 + 2b^2$ and $C_1 < C$, (A.19) holds provided r satisfies (3.16).

Proof of Proposition 1: It is possible to find an approximation of the sampling size r by adding some additional conditions to equation (3.14). Indeed, if the total sample size r is relatively large, then the binomial pmf in equation (3.14) can be approximated by the normal probability using the Central Limit Theorem [7],Based on the Central Limit Theorem, if the total sample size r is relatively large, (say, r is an order of magnitude bigger than C), then the binomial distribution for the sample size d_i in the i^{th} model instance can be approximated by the normal distribution: $Bin(r, \frac{1}{C}) \approx \mathcal{N}\left(\frac{r}{C}, \frac{r(C-1)}{C^2}\right)$. The value of r can be approximated by rewriting equation (3.14) as follows:

$$\begin{aligned} P(d_i \le \varepsilon - 1) &= P\left(\frac{d_i - \frac{r}{C}}{\sqrt{r\frac{1}{C}\left(1-\frac{1}{C}\right)}} \le \frac{\varepsilon - 1 - \frac{r}{C}}{\sqrt{r\frac{1}{C}\left(1-\frac{1}{C}\right)}}\right) \\ &\approx \Phi\left(\frac{(\varepsilon-1)C - r}{\sqrt{r(C-1)}}\right) = \bar{\Phi}\left(\frac{r-(\varepsilon-1)C}{\sqrt{r(C-1)}}\right) \end{aligned} \tag{A.20}$$

where

$$\bar{\Phi}(\kappa) = \int_\kappa^\infty \frac{1}{\sqrt{2\pi}} e^{\frac{t^2}{2}} dt$$

Our goal now is to find r that guarantees an upper-bound $P\left[\cup_{i=1}^{C}(d_i \le \varepsilon - 1)\right] \le CP(d_i \le \varepsilon - 1) \le \delta$. Since $\bar{\Phi}(\kappa) \le (\sqrt{2\pi}\kappa)^{-1}\exp(-\frac{\kappa^2}{2})$ for any $\kappa \ge 1$, inequality (3.5) holds whenever r satisfies the inequality:

$$\frac{r-(\varepsilon-1)C}{\sqrt{r(C-1)}} \ge \max\left(1, \sqrt{2\ln\left(\frac{C}{\sqrt{2\pi}\delta}\right)}\right) \tag{A.21}$$

Denote $X = \sqrt{r}$ and

$$A \equiv \max\left(1, \sqrt{2\ln\left(\frac{C}{\sqrt{2\pi}\delta}\right)}\right), \tag{A.22}$$

rewrite the inequality (A.21) as

$$X^2 - C(\varepsilon - 1) \ge AX\sqrt{C-1}, \tag{A.23}$$

and obtain its solution:

$$\sqrt{r} = X \ge \frac{A\sqrt{C-1}}{2} + \sqrt{\frac{A^2(C-1)}{4} + C(\varepsilon-1)} \tag{A.24}$$

provided r is relatively large [7]. Therefore, inequality (3.5) is guaranteed by

$$r \geq \max\left(10C, A^2(C-1) + 2C(\varepsilon - 1)\right) \tag{A.25}$$

where A is defined in (3.19).

LIST OF REFERENCES

[1] G. Andrew, R. Arora, J. Bilmes, and K. Livescu. Deep canonical correlation analysis. In *International Conference on Machine Learning*, pages 1247–1255, 2013.

[2] P. N. Belhumeur and D. J. Kriegman. What is the set of images of an object under all possible illumination conditions? *International Journal of Computer Vision*, 28(3):245–260, 1998.

[3] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 489–495. IEEE, 1999.

[4] T. E. Boult and L. G. Brown. Factorization-based segmentation of motions. In *Visual Motion, 1991., Proceedings of the IEEE Workshop on*, pages 179–186. IEEE, 1991.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[6] P. S. Bradley and O. L. Mangasarian. K-plane clustering. *Journal of Global Optimization*, 16(1):23–32, 2000.

[7] R. C. Bradley. Central limit theorems under weak dependence. *Journal of Multivariate Analysis*, 11(1):1–16, 1981.

[8] R. W. Butler and R. K. Sutton. Saddlepoint approximation for multivariate cumulative distribution functions and probability computations in sampling theory and outlier testing. *Journal of the American Statistical Association*, 93(442):596–604, 1998.

[9] G. Chen and G. Lerman. Spectral curvature clustering (scc). *International Journal of Computer Vision*, 81(3):317–330, 2009.

[10] H. Chen, P. Meer, and D. E. Tyler. Robust regression for data with multiple structures. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–1069. IEEE, 2001.

[11] X. Chen and D. Cai. Large scale spectral clustering with landmark-based representation. In *AAAI*, 2011.

[12] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

[13] A. Childs and N. Balakrishnan. Some approximations to the multivariate hypergeometric distribution with applications to hypothesis testing. *Computational statistics & data analysis*, 35(2):137–154, 2000.

[14] T.-J. Chin, H. Wang, and D. Suter. Robust fitting of multiple structures: The statistical learning approach. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 413–420. IEEE, 2009.

[15] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1197–1203. IEEE, 1999.

[16] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[17] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 1071–1076. IEEE, 1995.

[18] G. De Soete and J. D. Carroll. K-means clustering in a low-dimensional euclidean space. In *New approaches in classification and data analysis*, pages 212–219. Springer, 1994.

[19] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *International journal of computer vision*, 96(1):1–27, 2012.

[20] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE, 2009.

[21] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013.

[22] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1801–1807. IEEE, 2011.

[23] J. Feng, Z. Lin, H. Xu, and S. Yan. Robust subspace segmentation with block-diagonal prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3818–3825, 2014.

[24] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[25] A. W. Fitzgibbon and A. Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *Computer Vision-ECCV 2000*, pages 891–906. Springer, 2000.

[26] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(6):643–660, 2001.

[27] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[28] X. Guo. Robust subspace segmentation by simultaneously learning data representations and their affinity matrix. In *IJCAI*, pages 3547–3553, 2015.

[29] X. Guo, X. Liu, E. Zhu, and J. Yin. Deep clustering with convolutional autoencoders. In *International Conference on Neural Information Processing*, pages 373–382. Springer, 2017.

[30] T. Hastie and P. Simard. Metrics and models for handwritten character recognition. In *Conference on Statistical Science Honouring the Bicentennial of Stefano Franscinis Birth*, pages 203–219. Springer, 1997.

[31] Q. He, F. Kong, and R. Yan. Subspace-based gearbox condition monitoring by kernel principal component analysis. *Mechanical Systems and Signal Processing*, 21(4):1755–1772, 2007.

[32] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering appearances of objects under varying illumination conditions. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–11. IEEE, 2003.

[33] R. Hoseinnezhad and A. Bab-Hadiashar. Multi-bernoulli sample consensus for simultaneous robust fitting of multiple structures in machine vision. *Signal, Image and Video Processing*, pages 1–10, 2014.

[34] H. Isack and Y. Boykov. Energy-based geometric multi-model fitting. *International journal of computer vision*, 97(2):123–147, 2012.

[35] M. Jaberi, M. Pensky, and H. Foroosh. Swift: Sparse withdrawal of inliers in a first trial. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4849–4857, 2015.

[36] A. Jalali and R. Willett. Subspace clustering via tangent cones. In *Advances in Neural Information Processing Systems*, pages 6747–6756, 2017.

[37] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Discrete multivariate distributions*, volume 165. Wiley New York, 1997.

[38] I. T. Jolliffe. Principal component analysis and factor analysis. *Principal component analysis*, pages 150–166, 2002.

[39] K. Kanatani. Motion segmentation by subspace separation and model selection. *Computer Vision*, pages 586–591, 2001.

[40] F. Keller, E. Muller, and K. Bohm. Hics: high contrast subspaces for density-based outlier ranking. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1037–1048. IEEE, 2012.

[41] O. Kilinc and I. Uysal. Learning latent representations in neural networks for clustering through pseudo supervision and graph-based activity regularization. 2018.

[42] J. E. Kolassa. Multivariate saddlepoint tail probability approximations. *Annals of statistics*, pages 274–286, 2003.

[43] N. Lazic, I. Givoni, B. Frey, and P. Aarabi. Floss: Facility location for subspace segmentation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 825–832. IEEE, 2009.

[44] K.-C. Lee, J. Ho, and D. Kriegman. Nine points of light: Acquiring subspaces for face recognition under variable lighting. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–519. IEEE, 2001.

[45] B. Li, R. Liu, J. Cao, J. Zhang, Y.-K. Lai, and X. Liu. Online low-rank representation learning for joint multi-subspace recovery and clustering. *IEEE Transactions on Image Processing*, 27(1):335–348, 2018.

[46] C.-G. Li and R. Vidal. Structured sparse subspace clustering: a unified optimization framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 277–286, 2015.

[47] C.-G. Li, C. You, and R. Vidal. Structured sparse subspace clustering: A joint affinity learning and subspace clustering framework. *IEEE Transactions on Image Processing*, 26(6):2988–3001, 2017.

[48] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *Proc. of 27th Int. Conf. on machine learning (ICML)*, pages 663–670, 2010.

[49] G. Liu and S. Yan. Latent low-rank representation for subspace segmentation and feature extraction. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1615–1622. IEEE, 2011.

[50] G. Liu and S. Yan. Active subspace: Toward scalable low-rank learning. *Neural computation*, 24(12):3371–3394, 2012.

[51] H. Liu and S. Yan. Efficient structure detection via random consensus graph. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 574–581. IEEE, 2012.

[52] R. Liu, Z. Lin, F. De la Torre, and Z. Su. Fixed-rank representation for unsupervised visual learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 598–605. IEEE, 2012.

[53] 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 *European conference on computer vision*, pages 347–360. Springer, 2012.

[54] Y. Ma, H. Derksen, W. Hong, and J. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(9):1546–1562, 2007.

[55] L. Magri and A. Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3954–3961, 2014.

[56] L. Magri and A. Fusiello. Multiple model fitting as a set coverage problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3318–3326, 2016.

[57] M. Meila and J. Shi. A random walks view of spectral segmentation. 2001.

[58] F. Nie, D. Xu, I. W. Tsang, and C. Zhang. Spectral embedded clustering. In *IJCAI*, pages 1181–1186, 2009.

[59] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. S. Huang. Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition*, 43(1):113–127, 2010.

[60] V. M. Patel and R. Vidal. Kernel sparse subspace clustering. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 2849–2853. IEEE, 2014.

[61] X. Peng, J. Feng, J. Lu, W.-Y. Yau, and Z. Yi. Cascade subspace clustering. In *AAAI*, pages 2478–2484, 2017.

[62] X. Peng, H. Tang, L. Zhang, Z. Yi, and S. Xiao. A unified framework for representation-based subspace clustering of out-of-sample and large-scale data. *IEEE transactions on neural networks and learning systems*, 27(12):2499–2512, 2016.

[63] X. Peng, S. Xiao, J. Feng, W.-Y. Yau, and Z. Yi. Deep subspace clustering with sparsity prior. In *IJCAI*, pages 1925–1931, 2016.

[64] X. Peng, L. Zhang, and Z. Yi. Scalable sparse subspace clustering. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 430–437. IEEE, 2013.

[65] T. T. Pham, T.-J. Chin, J. Yu, and D. Suter. The random cluster model for robust geometric fitting. *IEEE transactions on pattern analysis and machine intelligence*, 36(8):1658–1671, 2014.

[66] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J.-M. Frahm. Usac: a universal framework for random sample consensus. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):2022–2038, 2013.

[67] M. Rahmani and G. K. Atia. Subspace clustering via optimal direction search. *IEEE Signal Processing Letters*, 2017.

[68] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

[69] J. A. Rice. *Mathematical Statistics and Data Analysis*. 3rd Edition, Duxbury Press, 2007.

[70] D. M. Rocke and D. L. Woodruff. Identification of outliers in multivariate data. *Journal of the American Statistical Association*, 91(435):1047–1061, 1996.

[71] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[72] K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers through model selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(6):983–995, 2006.

[73] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[74] H. Shinnou and M. Sasaki. Spectral clustering for a large data set by reducing the similarity matrix size. In *LREC*, 2008.

[75] M. Soltanolkotabi, E. J. Candes, et al. A geometric analysis of subspace clustering with outliers. *The Annals of Statistics*, 40(4):2195–2238, 2012.

[76] Y. Song, W. Cai, Q. Li, F. Zhang, D. Dagan Feng, and H. Huang. Fusing subcategory probabilities for texture classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4409–4417, 2015.

[77] A. Talwalkar, L. Mackey, Y. Mu, S.-F. Chang, and M. I. Jordan. Distributed low-rank subspace segmentation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3543–3550. IEEE, 2013.

[78] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999.

[79] R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *Computer Vision–ECCV 2008*, pages 537–547. Springer, 2008.

[80] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[81] P. H. Torr. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 356(1740):1321–1340, 1998.

[82] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[83] P. Tseng. Nearest q-flat to m points. *Journal of Optimization Theory and Applications*, 105(1):249–252, 2000.

[84] R. Unnikrishnan and M. Hebert. Robust extraction of multiple structures from non-uniformly sampled data. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1322–1329. IEEE, 2003.

[85] D. Verma and M. Meila. A comparison of spectral clustering algorithms. *University of Washington Tech Rep UWCSE030501*, 1:1–18, 2003.

[86] R. Vidal and P. Favaro. Low rank subspace clustering (lrsc). *Pattern Recognition Letters*, 43:47–61, 2014.

[87] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12):1945–1959, 2005.

[88] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *International Journal of Computer Vision*, 68(1):7–25, 2006.

[89] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[90] T. Vincent and R. Laganiére. Detecting planar homographies in an image pair. In *Image and Signal Processing and Analysis, 2001. ISPA 2001. Proceedings of the 2nd International Symposium on*, pages 182–187. IEEE, 2001.

[91] B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter. Joint detection and estimation of multiple objects from image observations. *Signal Processing, IEEE Transactions on*, 58(10):5129–5141, 2010.

[92] H. Wang, T.-J. Chin, and D. Suter. Simultaneously fitting and segmenting multiple-structure data with outliers. *IEEE transactions on pattern analysis and machine intelligence*, 34(6):1177–1192, 2012.

[93] L. Wang, C. Leckie, R. Kotagiri, and J. Bezdek. Approximate pairwise clustering for large data sets via sampling plus extension. *Pattern Recognition*, 44(2):222–235, 2011.

[94] Y.-X. Wang and H. Xu. Noisy sparse subspace clustering. *Journal of Machine Learning Research*, 17(12):1–41, 2016.

[95] H. S. Wong, T.-J. Chin, J. Yu, and D. Suter. A simultaneous sample-and-filter strategy for robust multi-structure model fitting. *Computer Vision and Image Understanding*, 117(12):1755–1769, 2013.

[96] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

[97] S. Xiao, M. Tan, D. Xu, and Z. Y. Dong. Robust kernel low-rank representation. *IEEE transactions on neural networks and learning systems*, 27(11):2268–2281, 2016.

[98] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[99] L. Xu, E. Oja, and P. Kultanen. A new curve detection method: randomized hough transform (rht). *Pattern recognition letters*, 11(5):331–338, 1990.

[100] D. Yan, L. Huang, and M. I. Jordan. Fast approximate spectral clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 907–916. ACM, 2009.

[101] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Computer Vision–ECCV 2006*, pages 94–106. Springer, 2006.

[102] Y. Yan, M. Liu, S. Chen, and F. Xiao. A novel robust model fitting approach towards multiple-structure data segmentation. *Neurocomputing*, 239:181–193, 2017.

[103] A. Y. Yang, S. R. Rao, and Y. Ma. Robust statistical estimation and segmentation of multiple subspaces. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*, pages 99–99. IEEE, 2006.

[104] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2):212–225, 2008.

[105] B. Yang, X. Fu, and N. D. Sidiropoulos. Learning from hidden traits: Joint factor analysis and latent clustering. *IEEE Transactions on Signal Processing*, 65(1):256–269, 2017.

[106] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5147–5156, 2016.

[107] M. Yin, Y. Guo, J. Gao, Z. He, and S. Xie. Kernel sparse subspace clustering on symmetric positive definite manifolds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5157–5164, 2016.

[108] T. Zhang, A. Szlam, and G. Lerman. Median k-flats for hybrid linear modeling with many outliers. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 234–241. IEEE, 2009.

[109] W. Zhang and J. Kseckä. Nonparametric estimation of multiple structures with outliers. In *Dynamical Vision*, pages 60–74. Springer, 2007.

[110] M. Zuliani, C. S. Kenney, and B. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–153. IEEE, 2005.