Efficient Sparse Subspace Clustering by Nearest Neighbour Filtering

Yi Guo[1]

Western Sydney University, Parramatta, NSW 2150, Australia

Stephen Tierney, Junbin Gao

The University of Sydney Business School

Abstract

Subspace identification has been used extensively because its ability to detail the internal subspace structure of data, which can be used in a variety of applications such as dimension reduction, anomaly detection and so on. However, many advanced algorithms are limited on their applicability in large data sets due to large computation and memory requirements with respect to the number of input data points. To overcome this problem, we propose a simple method that screens out a large number of data points by using k nearest neighbours and subspace recovery is performed on reduced set. The proposed method is surprisingly simple with significant reduction to both memory and computations requirements, and yet possesses desirable probability lower bound for its success in the context of big data. Besides theoretical analysis, our experiments also show that our method exceeds theoretical expectations and outperforms existing similar algorithms.

Keywords: Clustering, Subspace Identification, KNN, Optimisation

1. Introduction

Identifying a union of subspaces, also called subspace clustering, is proven useful in a large number of applications. Examples include temporal video segmentation [32, 37], segmentation of hyperspectral mineral data [12, 32], feature extraction [20, 18] and many more.

We first of all formalise subspace clustering problem in matrix algebra as follows: given a data matrix of N observed column-wise samples $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{D\times N}$, where D is the dimension of the data in the ambient space. Data within $\mathbf{X}$ is assumed to be drawn from a union of L subspaces $\{\mathcal{S}_j\}_{j=1}^L$ of dimensions $\{d_j\}_{j=1}^L$. The objective of subspace clustering is to learn the corresponding subspace labels $\mathbf{l} = [l_1, l_2, \dots, l_N] \in \mathbb{N}^N$ for all the data points where each $l_i \in \{1, \dots, L\}$. Both the number of subspaces L and the dimension of each subspace d_j are unknown. To further complicate the problem it is rarely the case that $\mathbf{X}$ is clean. The data is often subject to noise or corruption either at the time of capture (e.g. a digital imaging device) or during transmission (e.g. wireless communication). It is quite clear that

[1]The author to whom all the correspondence should be addressed.

subspace clustering is a difficult task since one must produce accurate results quickly while contending with numerous unknown parameters and large volume of potentially noisy data.

The usefulness of subspace clustering has spurred the development of subspace clustering algorithms, from early algebraic methods such as Generalised Principal Component Analysis (GPCA) [37, 21] to most recent multi-view subspace clustering [39, 43], improving accuracy and robustness [10, 16] in various ways.

Spectral methods have come to dominate subspace clustering literature as they offer some advantages over other types of methods. They mainly consist of two stages: learning a similarity matrix for the data then assigning cluster labels through segmentation of the similarity matrix. A forerunner of spectral methods called "Sparse Subspace Clustering" (SSC) was introduced in [6]. SSC exploits the self-expressive property of data [7] to find the subspaces:

$$\mathbf{x}_i = \mathbf{X}\mathbf{z}_i,\ z_{ii} = 0,\ \forall i = 1 \sim N \tag{1}$$

where $\mathbf{z}_i$ is a vector of reconstruction coefficients for $\mathbf{x}_i$ and z_{ii} is the ith element in $\mathbf{z}_i$. To reconstruct $\mathbf{x}_i \in \mathcal{S}_j$, one only needs d_i other points from the same subspace after removal of noise. This means that each data point can be sparsely represented by other points coming from the same subspace as the given one. Sparse Subspace Clustering as its name suggests exploits this fact. Combining the additive noise model, the objective function for SSC is

$$\min_{\mathbf{Z}} \lambda\|\mathbf{Z}\|_1 + \frac{1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}\|_F^2 \quad \text{s.t. } \mathrm{diag}(\mathbf{Z}) = \mathbf{0}, \tag{2}$$

where $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_N]$ is the coefficients matrix and $\|\mathbf{Z}\|_1 = \sum_i \sum_j |Z_{ij}|$ is the ℓ_1 norm. Alternatively one can instead pronounce the error in the objective as

$$\min_{\mathbf{Z}} \lambda\|\mathbf{Z}\|_1 + \frac{1}{2}\|\mathbf{E}\|_F^2 \quad \text{s.t. } \mathbf{X} = \mathbf{X}\mathbf{Z} + \mathbf{E}, \mathrm{diag}(\mathbf{Z}) = \mathbf{0} \tag{3}$$

where $\mathbf{E}$ is the error matrix and $\lambda \geq 0$ is used to control the trade-off between the sparsity of $\mathbf{Z}$ and the error.

To obtain the final subspace labels the reconstruction coefficients in $\mathbf{Z}$ are given a secondary interpretation as the affinity or similarity between the data points. Spectral clustering is applied to $\mathbf{Z}$. Typically N-CUT [28] is used as it produces the most accurate segmentation even for poorly constructed affinity matrices and is relatively fast.

While SSC has promising theoretical guarantees [7] and has shown good performance for small evaluation datasets it is not widely applied in large size data sets. This is due to (1) $\mathrm{O}(N^2)$ memory requirements and (2) $\mathrm{O}(N^2)$ FLOP (floating point operations) requirements. The first is easily understood as $\mathbf{Z} \in \mathbb{R}^{N\times N}$. One could contend that $\mathbf{Z}$ could be stored in a sparse format, however since the support of $\mathbf{Z}$ is unknown and varies between iterations of the SSC algorithm this approach would introduce significant overhead. Similarly the high FLOP count is due to the dimensions of $\mathbf{Z}$, since each element must be calculated per iteration.

Although SSC has some desirable properties such as correct subspace identification guarantee [30] , and flexibility in modelling which inspired many methods like [12, 32, 11], huge memory and computational costs prevent it and its later variants from being applied to even modestly sized datasets. In light of this important issue, there has been considerable interests in developing tractable subspace clustering algorithms. We elaborate the latest developments in the next section and propose our solution.

2. Related Work on Efficient Solutions

To alleviate the computational complexity of an algorithm, one proliferate direction is to approximate existing methods at the cost of minimum accuracy loss. In subspace clustering domain, approximation methods can be divided into two classes: inductive and heuristic. Inductive methods perform some subspace clustering algorithm or learn the similarity matrix on a small subset of the data. The full structure of the similarity matrix or labels is then obtained by inductive transfer from the subset. Heuristic methods directly assign cluster labels by greedy selection of nearest neighbours based on a defined metric.

Scalable SSC (SSSC) [26] was probably the first attempt to resolve computational issues. As an inductive method, it first selects some candidate samples from the data and performs SSC on these samples. Then the remaining samples are assigned to clusters based on their fit into the clusters formed by the training candidates. This approach has considerable issues. First, the candidate samples are selected by uniform random sampling. This does not guarantee that every cluster will be accounted for in the candidate set. Second, there must be enough candidate samples for the candidate clusters to generalise to the remaining samples. Correctly choosing the number of samples is a difficult task.

Arguably the most prominent heuristic method is Orthogonal Matching Pursuit (OMP), which has been long used as a greedy sparse approximation method [34]. For each data point, a residual vector is set as the data point. Then the nearest neighbour to the residual is found and then the residual is updated by a projection of the data point onto the span formed by the currently picked up neighbours. This is repeated until the number of neighbours is reached or the norm of the residual is small enough. OMP is also known by other names such as Greedy Feature Selection (GFS) [5] and is a constant well that researches draw from [41].

Although OMP is advertised as a fast approximate method however in practice we do not find this to be the case. First, the nearest neighbour search is performed for every iteration. Second, the computational and memory requirements of a naive implementation tend to increase dramatically as D increases due to the need to create a $D \times D$ matrix in each iteration for every data point. This, in some cases, makes it just as intractable as SSC. Third, the naive implementation requires successive computation of the SVD of the span matrix at each iteration. The second and third points have fortunately been mitigated by improvements such as Rank-1 updating scheme of Moore-Penrose Inverse [27], factorisation approaches such as QR and Cholesky decomposition [31] or more esoteric methods

[34, 38, 22]. However the speed gain from OMP algorithm is shadowed by its performance. As shown in Section 5 we find that GFS (OMP) performs poorly in terms of clustering accuracy.

OMP has inspired other methods such as Greedy Subspace Clustering (GSC) [25] and ORGEN [40]. GSC differs from OMP in neighbour selection. Each neighbour is selected by finding the data point which has the largest norm of projection onto the span formed by the current neighbours. Although at first glance GSC appears to be simple and thus likely to scale well w.r.t. N, the projection step is quite computationally intensive and just like OMP the nearest neighbour search is performed in each iteration.

ORGEN extends the SSC model to the Elastic-Net model. That is, it uses the ℓ_2 norm in tandem with the ℓ_1 norm. From an initialisation point of some neighbours of each $\mathbf{x}_i$ it solves the Elastic-Net objective then determines an "oracle point", which is the residual from fitting the coefficients from the Elastic-Net procedure to the model. This oracle point is then used to find potential new neighbours. The procedure terminates when no new neighbours are added at the end of an iteration. ORGEN suffers from a number of problems. First it is highly initialisation dependant. The authors suggest performing ℓ_2 sub problem and choosing the largest valued elements as the initialisation pool for each $\mathbf{x}_i$. This can be slow as when D is large and the ℓ_2 problem lacks rigorous guarantees of successful subspace identification. Second the repeated computation of elastic net is problematic when the active set grows large. This is a very real concern as termination only occurs when the active set stops growing. The active set could grow to the full size of the data set. Third the claim of improved running time is not evident. The authors show running times for single $\mathbf{x}_i$ instead of the whole data $\mathbf{X}$ and do not compare to different approaches such as [26] or [14].

Heuristic methods are often incredibly simple. For example Robust Subspace Clustering via Thresholding (TSC) [14] essentially performs nearest neighbour based spectral clustering. For each point the nearest neighbours are found and the affinity matrix is constructed using exponential inner product between each of the neighbours. However, the subspace identification accuracy is problematic.

3. Efficient Sparse Subspace Clustering

There is still much room to improve, because there are problems of existing methods, compromising either efficiency or accuracy. Our contribution to efficient subspace clustering is inspired by the sparsity of SSC. It has been shown repeatedly that each data point can be reconstructed by only d_i (the dimensionality of the underlying subspace) other data points from its corresponding subspace [7, 30, 44]. This is the basis of SSC's operation. By finding the sparsest representation one will be left with the minimum support to represent a data point $\mathbf{x}_i$, corresponding to data points in the same subspace. Therefore it is clear that blindly considering all other points as candidates for reconstruction is very wasteful since only a relative few points will be left as support. Furthermore the process is very intensive in terms of

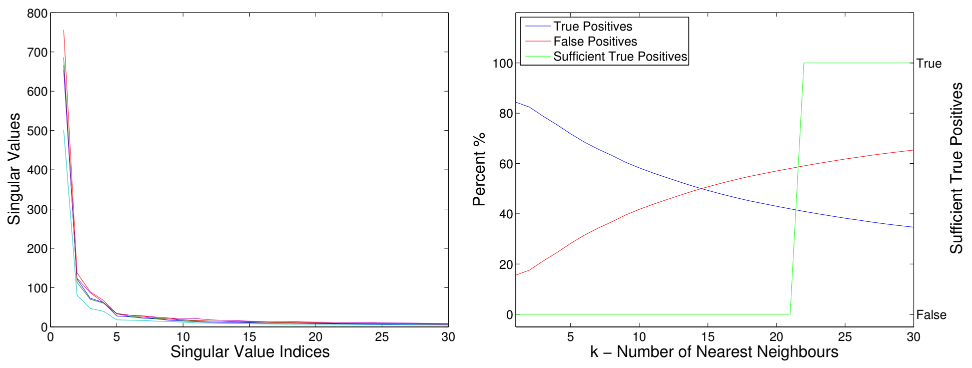


Figure 1: Left: Singular values of several faces (different colours) from the Extended Yale B dataset. Right: Average percentage of true positive and false positive nearest neighbour selection from the entire Extended Yale B dataset and correspondingly, in green, a plot of when the sufficient true positives are reached on average.

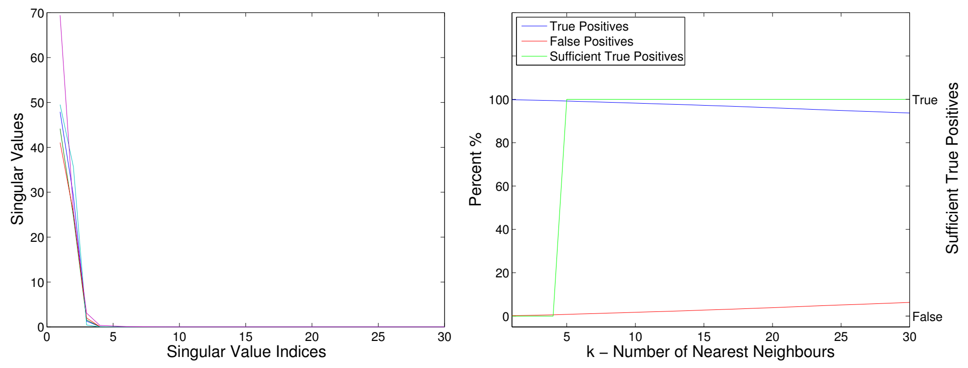


Figure 2: Left: Singular values of several motions from the Hopkins 155 Motion Dataset. Right: Average percentage of true positive and false positive nearest neighbour selection from the checkerboard and traffic sequences in the Hopkins 155 Motion Dataset dataset and correspondingly, in green, a plot of when the sufficient true positives are reached on average.

memory requirements. The most efficient algorithm for solving SSC requires $\mathrm{O}\big(N^2\big)$ FLOPs per iteration and the storage of $\mathrm{O}\big(N^2\big)$ over the algorithm's entire operation w.r.t. $\mathbf{Z}$.

Therefore if we can safely prune a vast majority of data points as candidates when reconstruct one data point, then we can massively reduce computational and memory load. In other words this means that for any column of $\mathbf{Z}$, say $\mathbf{z}_i$, we would only solve for a small subset of the entries of $\mathbf{z}_i$ while the other entries are left zero. To this end we propose kSSC, in which we limit each data point to be represented by at most k other data points selected simply by kNN (k-Nearest Neighbours) with normalised inner product. Thus the relaxed objective function for kSSC is

$$\min_{\mathbf{z}_i} \lambda \|\mathbf{z}_i\|_1 + \frac{1}{2}\sum_{i}^{N} \|\mathbf{x}_i - \sum_{j \in \Omega_i} \mathbf{x}_j z_{ij}\|_F^2 \tag{4}$$

where Ω_i is the set of data points to use for reconstruction of data point i. Under this objective we can reduce both the memory and FLOP requirements to $\mathrm{O}\big(kN\big)$ w.r.t. $\mathbf{z}_i$, which when $k \ll N$ provides massive savings.

Evidently the success of kSSC relies heavily upon both the size of k and the scheme that is used to select Ω_i. First one should always choose $k \geq d_i$ since each data point needs at a minimum d_i other points for reconstruction. Figure 1 shows singular values for multiple subspaces (a single subspace corresponds to a single subject or face) from the Extended Yale B dataset. The point at which the singular values begin to trail off reveals the underlying subspace dimension d_i, which in this case is 9 [7]. Therefore in that case we should set $k \geq 9$. Similarly in Figure 2 we perform the same analysis on the motion segmentation dataset and find that the subspace dimension is 4. Since in almost all cases $d_i \ll N$ and thus $k \ll N$ the computational and memory requirements of kSSC will be much lower than SSC. However even in cases where there is no prior information about d_i's, one can set a large, conservative value for k with little impact on overall performance when $k \ll N$, for example let $k = D$.

Moreover, one must choose Ω_i such that it contains the "right" data points from $\mathbf{x}_i$'s subspace. Uniformly random sampling k points is a poor choice since the selected ones may not belong to the same subspace. Recent works such as [5, 25, 44] have demonstrated that even in noisey cases or cases of subspace intersection that the points closest to each $\mathbf{x}_i$ in the ambient space usually correspond to the most strongly connected data points in $\mathbf{Z}$, i.e. data points from the same subspace. We come to the same conclusions and most importantly, we provide the conditions and probability bounds of the effectiveness of selecting points from the subspace using k nearest neighbours. We present our two central theorems here. First in the case of noise free data:

Theorem 1. *Given the conditions in Corollary 9, if* $\mathcal{A}_{\ell,1} \leq \min_\ell \left\{ \frac{\sqrt{d_\ell}\delta}{2\sqrt{(d_1 \wedge d_\ell)(t \log N_\ell + t^2)}} \right\} \forall l = 2, \dots, L,$ *where L is the total number of subspaces, then the points selected for any sample $\mathbf{y}$ from subspace $\mathcal{S}_1$ by using kNN with $k = k_0$ contains no samples from other subspaces but $\mathcal{S}_1$ with probability at least* $1 - (L-1)e^{-t} - \exp\left(-k_0(n - \ln n + 1) - (n-1)\right)$.

Second in the case of noisey data with the assumption that the noise is Gaussian with zero mean and variance σ:

Theorem 2. *Given the conditions in Corollary 9 and the noise model* (A.10)*, for a small positive ϵ, if* $\mathcal{A}_{\ell,1} \leq \min_\ell \left\{ \frac{\sqrt{d_\ell}(\delta - 6\epsilon)}{2\sqrt{(d_1 \wedge d_\ell)(t \log N_\ell + t^2)}} \right\}$*, then the samples selected for any sample $\mathbf{y}_1$ from subspace $\mathcal{S}_1$ by using kNN with $k = k_0$ contains no samples from other subspaces but $\mathcal{S}_1$ with probability at least* $1 - (L-1)e^{-t} - \exp\left(-k_0(n - \ln n + 1) - (n-1)\right) - 2L\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - L\frac{D\sigma^4}{\epsilon^2}$.

We leave the lengthy proofs, detailed conditions and symbols definitions in the appendix in order not to obstruct the flow. Theorem 1 says that for subspace clustering purpose we can simply set Ω_i for $\mathbf{x}_i$ as its k nearest neighbours from the original ambient space when N is large. The neighbours are very likely coming from the same subspace as the given one under some assumptions listed in Theorem 1. However when $\mathbf{X}$ is subject to noise, the assumptions are stronger with also lower probability for kNN to select neighbours from the same subspace. For this reason we recommend setting k well above d_i to provide sufficient head room. Furthermore we suggest increasing k as the magnitude of expected noise

Algorithm 1 kSSC

```
Require: $\mathbf{X}^{D\times N}$ - observed data, $k$ - number of neighbours, $L$ - number of subspaces
1: for $i \to N$ in parallel do
2:     Set $\Omega_i$ by kNN
3:     Obtain coefficients $\mathbf{z}_i$ by solving (4)
4: end for
5: Form the similarity graph
       $\mathbf{W} = |\mathbf{Z}| + |\mathbf{Z}|^T$        (5)
6: Apply N-Cut to $\mathbf{W}$ to partition the data into $L$ subspaces
7: return Subspaces $\{S_i\}_{i=1}^{L}$
```

increases, since as noise increases, so does the likelihood of false positive neighbour selection as shown in Theorem 2. Moreover, these results suggest that a further subspace identification procedure is *absolutely necessary* and hence gives rise to kSSC. In Figure 1 and Figure 2, we demonstrate this effect on the Yale and Rigid Motion datasets respectively. We note that the required value for k to select sufficient true positives via kNN exceeds d_i. This is due to the presence of noise and corruptions in the data and the sometimes small distance between subspaces, particularly for the Extended Yale B dataset. Although still extremely small relative to N.

In summary we propose to eliminate the calculation of redundant elements of $\mathbf{Z}$ by computing only k rather than N coefficients for each $\mathbf{x}_i$. An overview of the entire method can be found in Algorithm 1. Subspace identification accuracy can be exactly maintained from SSC provided that the following conditions are met: (a) k is equal to or greater than $\max(d_i)$, and (b) the elements of Ω_i are nearest neighbours of $\mathbf{x}_i$. These conditions are sufficient but not necessary. In some cases clustering accuracy could be maintained when k is less than $\max(d_i)$ or different filtering method is used. However when these conditions are met kSSC ensures that SSC's guarantee of correct subspace identification and robustness to noise is preserved since kNN is guaranteed to correctly identify neighbours (see Appendix A). Furthermore kSSC is easily solved in parallel as Ω_i and each column of $\mathbf{Z}$ is independent.

3.1. Optimisation

We use FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) [3, 15] to solve (4). FISTA is an accelerated gradient descent scheme for solving objective functions containing a smooth part and non-smooth part as is the case with (4). One of the key abilities of FISTA is that it guarantees a convergence rate of $\mathrm{O}\left(\frac{1}{\mathtt{t}^2}\right)$ where $\mathtt{t}$ is the iteration counter. This is achieved by dynamically setting the rate of descent parameter (Lipschitz constant) and using the two previous iteration points to accelerate the gradient descent. Furthermore FISTA provides the aforementioned ability with minimal computational and memory overhead. Each iteration of FISTA only requires solving a closed form proximity problem which in the case of ℓ_1 minimisation can be solved at an element wise level. This allows us to resolve the selective fitting term of (4) since we can enforce it by ignoring the elements of $\mathbf{Z}$ that are outside of Ω.

We begin by re-writing, with some abuse of notation, the original objective (4) for a single column of $\mathbf{Z}$

$$\min_{\mathbf{z}_i} L = \lambda \|\mathbf{z}_i\|_1 + \frac{1}{2}\|\mathbf{x}_i - \mathbf{X}_i\mathbf{z}_i\|_2^2 \tag{6}$$

where $\mathbf{z}_i = \mathbf{z}_{\Omega_i i}$ the vector of elements indexed in Ω_i and $\mathbf{X}_i = \mathbf{X}_{(:,\Omega_i)}$ the matrix formed from the columns of $\mathbf{X}$ indexed by Ω_i. Note that we have removed the constraint $\text{diag}(\mathbf{Z}) = \mathbf{0}$ since we enforce it by ensuring that no diagonal entries are present in each Ω_i.

At each iteration in the FISTA scheme one must solve the ℓ_1 proximal linearised form of L. Denote the linearisation of L at point $\mathbf{z}_i^{\mathrm{t}}$

$$\min_{\mathbf{z}_i} \widetilde{L}_\rho(\mathbf{z_i}, \mathbf{z_i^t}) = \lambda\|\mathbf{z}_i\|_1 + \frac{\rho}{2}\|\mathbf{z}_i - (\mathbf{z}_i^{\mathrm{t}} - \frac{1}{\rho}\partial F(\mathbf{z}_i^{\mathrm{t}}))\|_2^2, \tag{7}$$

where $F = \frac{1}{2}\|\mathbf{x}_i - \mathbf{X}_i\mathbf{z}_i\|_2^2$ and correspondingly $\partial F = -\mathbf{X}_i^T(\mathbf{x}_i - \mathbf{X}_i\mathbf{z}_i^{\mathrm{t}})$. The solution to (7) is given by the closed-form ℓ_1 shrinkage function $\mathcal{S}_\tau$ as follows

$$\mathcal{S}_{\frac{\lambda}{\rho}}(\mathbf{z}_i^{\mathrm{t}}) = \text{sign}(\mathbf{z}_i^{\mathrm{t}} - \frac{1}{\rho}\partial F(\mathbf{z}_i^{\mathrm{t}}))\max(|\mathbf{z}_i^{\mathrm{t}} - \frac{1}{\rho}\partial F(\mathbf{z}_i^{\mathrm{t}})| - \frac{\lambda}{\rho}, 0). \tag{8}$$

We refer readers to [2, 19] for further details. The full algorithm is outlined in Algorithm 2.

Algorithm 2 Solving (4) via FISTA

```
Require: $r_i = \infty$, $\mathbf{z}_i = \mathbf{0}$, $\mathbf{j}_i = \mathbf{0}$, $\alpha_i = 1$, $\lambda$, $\rho_i$, $\gamma$, $\epsilon$
  while $r_i^{\mathrm{t}} - r_i^{\mathrm{t}-1} \geq \epsilon$ do
    while $L(\mathcal{S}_{\frac{\lambda}{\rho}}(\mathbf{j}_i^{\mathrm{t}})) \geq \widetilde{L}_\rho(\mathcal{S}_{\frac{\lambda}{\rho}}(\mathbf{j}_i^{\mathrm{t}}), \mathbf{j}_i^{\mathrm{t}})$ do
      $\rho_i = \gamma\rho_i$
    end while
    $\mathbf{z}_i^{\mathrm{t}+1} = \mathcal{S}_{\frac{\lambda}{\rho}}(\mathbf{j}_i^{\mathrm{t}}))$
    $\alpha_i^{\mathrm{t}+1} = \frac{1+\sqrt{1+4\alpha_i^{t}{}^2)}}{2}$
    $\mathbf{j}_i^{\mathrm{t}+1} = \mathbf{z}_i^{\mathrm{t}+1} + \left(\frac{\alpha_i^{\mathrm{t}}-1}{\alpha_i^{\mathrm{t}+1}}\right)(\mathbf{z}_i^{\mathrm{t}+1} - \mathbf{z}_i^{\mathrm{t}})$
    $r_i^{\mathrm{t}+1} = L(\mathcal{S}_{\frac{\lambda}{\rho}}(\mathbf{j}_i^{\mathrm{t}}))$
  end while
```

3.2. Segmentation

After solving (4) for each $\mathbf{z}_i$ the next step is to form $\mathbf{Z}$ and use the information encoded in $\mathbf{Z}$ to assign each data point to a subspace. A robust approach is to use spectral clustering. The matrix $\mathbf{Z}$ can be interpreted as the affinity or distance matrix of an undirected graph. Element Z_{ij} corresponds to the edge weight or affinity between vertices (data points) i and j. Then we use the spectral clustering technique, Normalised Cuts (N-Cut) [28] in particular as in SSC, to obtain final segmentation. Since we expect $\mathbf{Z}$ to be sparse in most cases N-Cut should have reasonable computation time, particularly in comparison to a full $\mathbf{Z}$ matrix. However in cases where N-Cut is too slow one can use approximate techniques such as the Nyström method [8].

Spectral segmentation techniques such as N-Cut require the number of subspaces p as a parameter. In the case where the number of subspaces is unknown one can use either the Eigen-gap [42, 36, 30] or the closely related SVD-gap heuristic of [17]. The Eigen-gap heuristic uses the eigenvalues of $\mathbf{Z}$, see Eq. (5), to find the number of subspaces. It does this by finding the largest gap between the ordered eigenvalues, the number of eigenvalues before this point is treated as the number of clusters. Let $\{\delta_i\}_{i=1}^N$ be the descending sorted eigenvalues of $\mathbf{Z}$ such that $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_N$. Then L can be estimated by

$$L = \underset{i=1,\ldots,N-1}{\operatorname{argmax}} \ (\delta_i - \delta_{i+1})$$

The SVD-gap heuristic is the same procedure with eigenvalues of $\mathbf{Z}$ replaced with singular values. Further improvements upon the Eigen-gap heuristic have been made, see [42] for details.

3.3. Complexity Analysis

The complexity of kSSC only varies from SSC w.r.t. $\mathbf{Z}$ as can be seen from Algorithm 1 and a comparison of Algorithm 2. It differs in two ways. First we must find the k nearest neighbours of each $\mathbf{x}_i$. Fortunately fast approximate methods exist for computing kNN and are freely available in packages such as FLANN [23]. The computation time for kNN is on the order of $\mathrm{O}\big(N \log N\big)$ and $\mathrm{O}\big(\log N\big)$ for preprocessing and searching respectively [13, 1, 24, 23].

Second is the updating of $\mathbf{z}_i$ at each iteration. Since we are only updating k entries of each column of $\mathbf{Z}$ instead of the full N entries the FLOP count is drastically reduced. Similarly the amount of memory required for updating $\mathbf{Z}$ is drastically reduced. They are both reduced from $\mathrm{O}\big(N^2\big)$ to $\mathrm{O}\big(N\big)$. We call the solver for (2) the relaxed variant and the solver for (3) the exact variant. Note that the relaxed variant has markedly lower FLOP counts than the exact variants. This assumes one execution of the ℓ_1 shrinkage operator per iteration. However in the case of FISTA, a single iteration may require many executions of the shrinkage operator due to the search scheme for optimal rate of descent parameter. In practice we find that solving the relaxed variant by FISTA is usually faster since choosing ρ is not a difficult task and can be estimated by running the solver on a small sub section of the data. Furthermore the FISTA based solver will converge much faster than other solvers. We provide a brief sample of running time differences in Figure 3 to illustrate the difference between implementation variants. We also demonstrate the effect of varying the number of available cores for the parallel implementations in Figure 4. We find that in the case of SSC as the number of cores increase the computation time also increases, which indicates that the performance of SSC is not as straight forward. In fact the performance is markedly worse than expected due to the overhead of sharing and multiple accessing of the full data matrix $\mathbf{X}$, which further

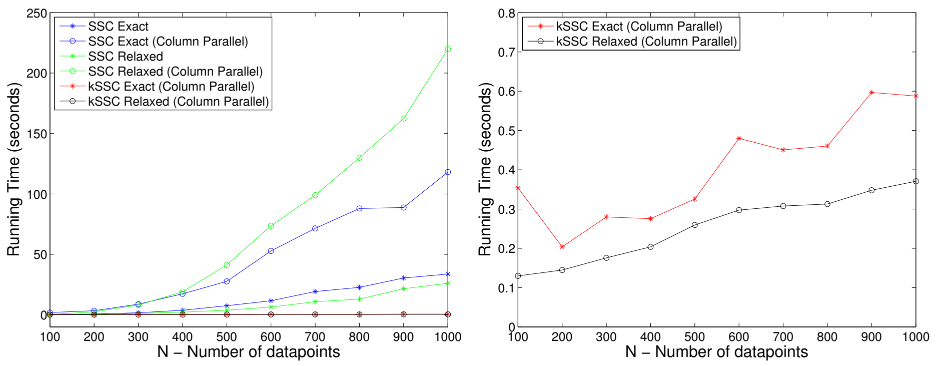


Figure 3: Left: A comparison of running times with increasing N between kSSC, SSC and their various implementations. Right: A zoomed comparison of running times with increasing N for kSSC relaxed and exact variants (taken from the Left plot). Note that the scales are different since kSSC takes a fraction of the running time of SSC.

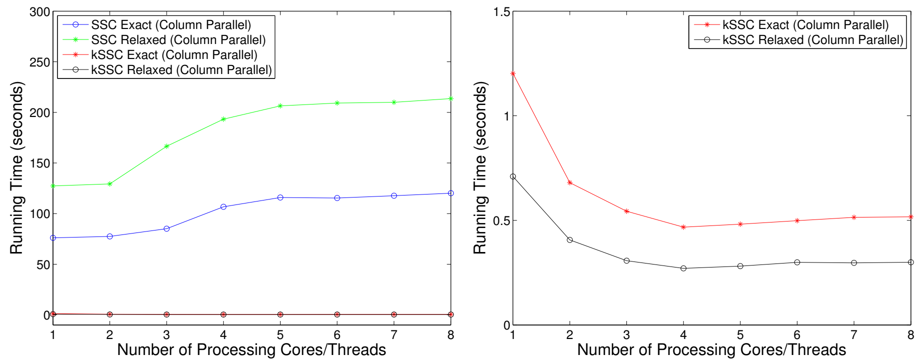


Figure 4: Left: A comparison of running times with increasing threads/cores between kSSC, SSC and their various implementations. Right: A zoomed comparison of running times with increasing threads/core for kSSC relaxed and exact variants (taken from the Left plot). Note that the scales are different.

reinforces the point that SSC does not scale well with large datasets with naive parallelisation. On the other hand, kSSC benefits greatly from increasing the core count and eventually plateaus due to it's own overhead.

4. Synthetic Evaluation

In this section we use synthetic data to experimentally evaluate our hypothesis proposed in Section 3 and the therotetical analysis in Appendix A that kSSC can match the clustering accuracy of SSC.

In an effort to maximise transparency and repeatability, all MATLAB code and data used for these experiments and those in Section 5 can be found online at `https://github.com/sjtrny/kSSC`. To help evaluate consistency parameters except for k were fixed for each experiment, which we further explain in the following subsections and are recorded in the code repository.

4.1. Metrics

Segmentation accuracy was measured using the subspace clustering error (SCE) metric [7], which is defined as

$$\text{SCE} = \frac{\text{num. of misclassified points}}{\text{total num. of points}} \times 100, \tag{9}$$

where lower subspace clustering error means greater clustering accuracy. In cases where we inject extra noise we report the level of noise using Peak Signal-to-Noise Ratio (PSNR) defined as

$$\text{PSNR} = 10 \log_{10} \left(\frac{s^2}{\frac{1}{mn} \sum_i^m \sum_j^n (X_{ij} - A_{ij})^2} \right) \tag{10}$$

where $\mathbf{X} = \mathbf{A} + \mathbf{N}$, $\mathbf{A}$ is the original data, $\mathbf{N}$ is noise and s is the maximum possible value of an element of $\mathbf{A}$. Decreasing values of PSNR indicate increasing amounts of noise.

4.2. Effect of subspace dimension, cluster size and ambient dimension

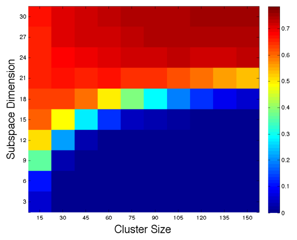
As noted in other works such as [14, 29] the ratio of the subspace dimension d_i to the number of points in each cluster N_i can play a dramatic role in the clustering accuracy of SSC. However these works also ignore the role of ambient dimension D. In this section we demonstrate the relationship between all three variables.

We generate 5 subspaces and vary their dimension d_i from 3 to 30 and N_i from 15 to 150. Each subspace is created using random orthonormal vectors as the basis with uniform random coefficients. For each pair of d_i and N_i we take the mean of the SCE over 50 trials. We repeat this again for 3 instances with D set to $30, 50$ and 100. For this experiment we set $k = \frac{N_i}{2}$ or $k = 1.5D$, whichever is smaller. The results shown in Figure 5 that kSSC can match the performance of SSC even when $k \ll d_i$.

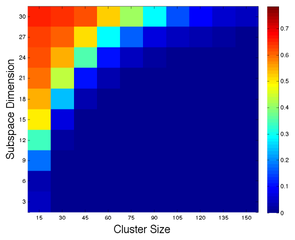
4.3. Effect of mean, variance and noise in subspace distribution

Note that the coefficients chosen are uniform random in synthetic data experiments in the prior subsection. This strategy is also adopted by several other works in this field. However data encountered in the real world may be normally distributed in their respective subspace. Furthermore the data points are often corrupted with noise, which we assume will be $\mathcal{N}(0, 1)$.

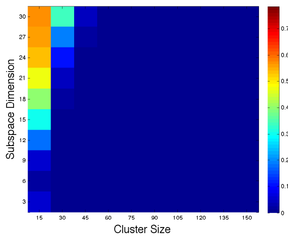
For this consideration, in this experiment we vary the mean μ and variance σ^2 of Gaussian distributed data points using random orthonormal vectors as the basis for each subspace. We create 5 subspaces with $d_i = 5$, $N_i = 50$ and $D = 50$. For each pair of μ and σ^2 we take the mean SCE over 50 trials. We repeat this again for 3 instances, each time increasing the noise factor, which we report using PSNR. For this experiment we set $k = 10$. The results shown in Figure 6 that kSSC can match the performance of SSC. We note that the effect of mean and variance on point distribution in the subspaces is significantly more pronounced as PSNR decreases.



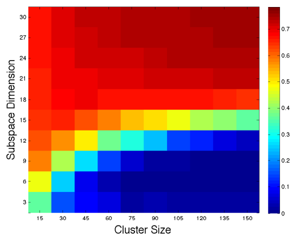
(a) SSC, $D = 30$



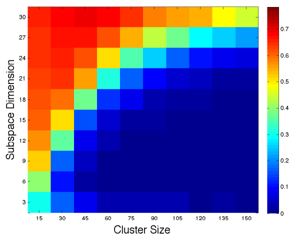
(b) SSC, $D = 50$



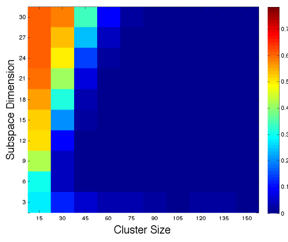
(c) SSC, $D = 100$



(d) kSSC, $D = 30$

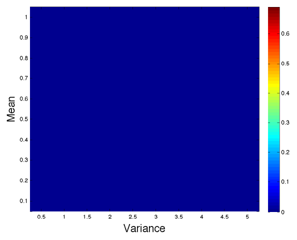


(e) kSSC, $D = 50$

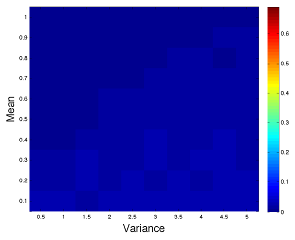


(f) kSSC, $D = 100$

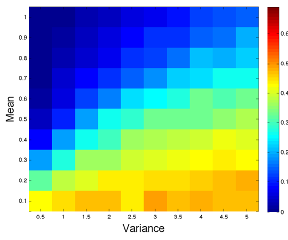
Figure 5: Effect of subspace dimension, cluster size and ambient dimension



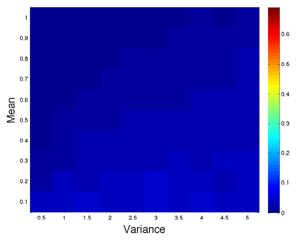
(a) SSC, PSNR 100



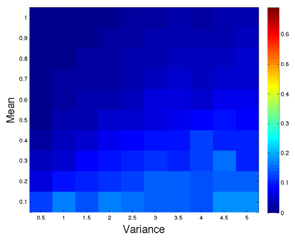
(b) SSC, PSNR 60



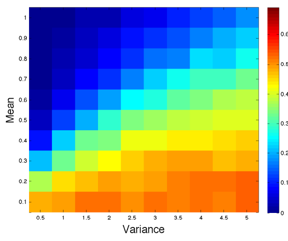
(c) SSC, PSNR 46



(d) kSSC, PSNR 100



(e) kSSC, PSNR 60



(f) kSSC, PSNR 46

Figure 6: Effect of mean, variance and noise in subspace distribution

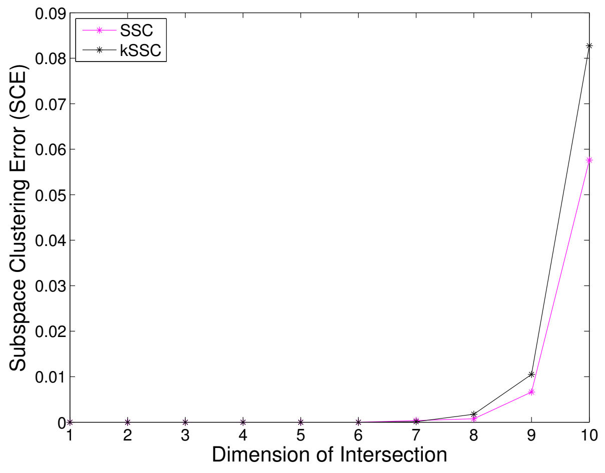


Figure 7: Effect of subspace intersection

4.4. Effect of subspace intersection

The intersection of subspaces (shared basis vectors) plays an important role in the clustering accuracy. As previously reported by others, the clustering accuracy decreases as the dimension of intersection increases. To demonstrate this effect and that kSSC can match SSC, we perform the same experiment as found in Section 8.1.1 of [14] and Section 5.1.2 of [29]. We generate two subspaces with $D = 200$, $d_i = 10$ and $N_i = 20d_i$ and vary the number of shared basis vectors b from 0 to d_i. We generate $\mathbf{U} \in \mathbb{R}^{D \times 2d_i - b}$ random orthonormal basis vectors and set the basis vectors for $\mathcal{S}_1$ to the first d_i columns of $\mathbf{U}$ and correspondingly the basis for $\mathcal{S}_2$ to the last d_i columns. We then take the average SCE over 20 trials for each b. Results are reported in Figure 7, where we can clearly see that kSSC closely matches the performance of SSC.

5. Experimental Evaluation

In this section, we evaluate the clustering performance of kSSC on semi-synthetic and real world datasets. We vary the amount of additional noise in some of these experiments to compare the robustness of kSSC against the pre-existing competitor algorithms Greedy Feature Selection (GFS), Greedy Subspace Clustering (GSC), Scalable Sparse Subspace Clustering (SSSC) and Robust Subspace Clustering via Thresholding (TSC). Additionally we use SSC to gauge baseline performance.

The running times of the experiments carried out in Sections 5.1 and 5.3 can be found in Figure 8. Since these experiments are small in size the running time reduction of kSSC is not that significant. However these tests indicate that kSSC matches the clustering accuracy of SSC very closely, an attribute that is not found in other methods. We perform a test in Section 5.2 to evaluate the running time of kSSC for a large scale data set.

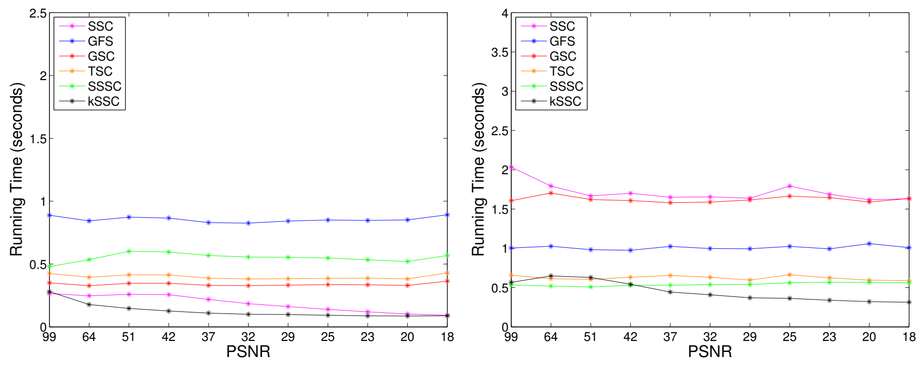
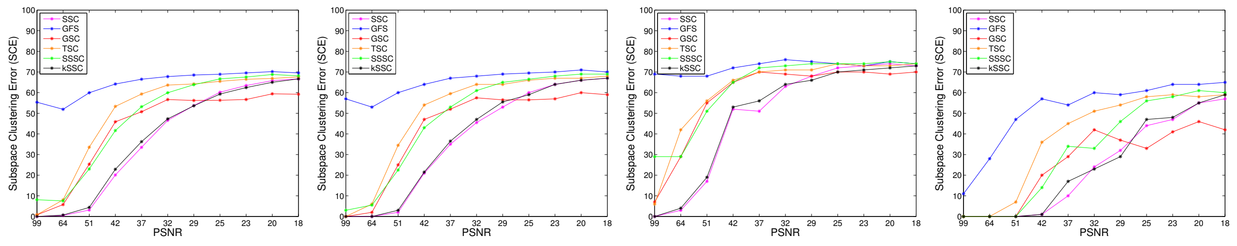


Figure 8: Left: Median running time of each tested algorithm for the Thermal Infrared Data Segmentation experiment found in Section 5.1. Right: Median running time of of each tested algorithm for the Motion Segmentation experiment found in Section 5.3. Overall in these experiments the benefit of kSSC is slight in comparison to other methods since N is low. We refer to readers Section 5.2 for a comparison in running time where N is large.



(a) Mean Error (b) Median Error (c) Max Error (d) Min Error

Figure 9: Semi-synthetic Hyperspectral TIR

5.1. *Thermal Infrared Data Segmentation*

We assemble synthetic data from a library of thermal infrared (TIR) hyper spectral mineral data. The library consists of 120 pure materials spectra samples with $D = 321$. We generate 5 subspaces with $d_i = 5$. For each subspace we randomly select 5 spectra in the TIR library and generate 50 points using uniform random nonnegative coefficients. We then corrupt the data with various levels of standard Gaussian noise and evaluate clustering performance of our kSSC and other contenders. The experiment is repeated for 50 trials for each level of noise to obtain an average SCE. Results can be found in Figure 9. kSSC closely tracks the performance of SSC and outperforms all other methods.

5.2. *Large Scale Thermal Infrared Segmentation*

The main goal of kSSC is to maintain SSC's clustering accuracy but in a fraction of the time. To confirm this ability, we create a large scale semi-synthetic dataset from the TIR data used in the previous subsection. We generate data in a similar fashion to the previous section. However for each subspace we generate N_i points using uniform random coefficients where we vary N_i from 100 to 4000. Therefore the size of the largest tested data set is 20,000. For this experiment, SSC, GFS and GSC stop early since

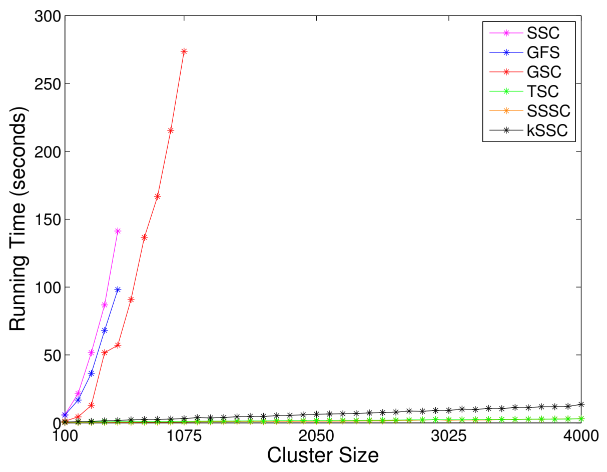
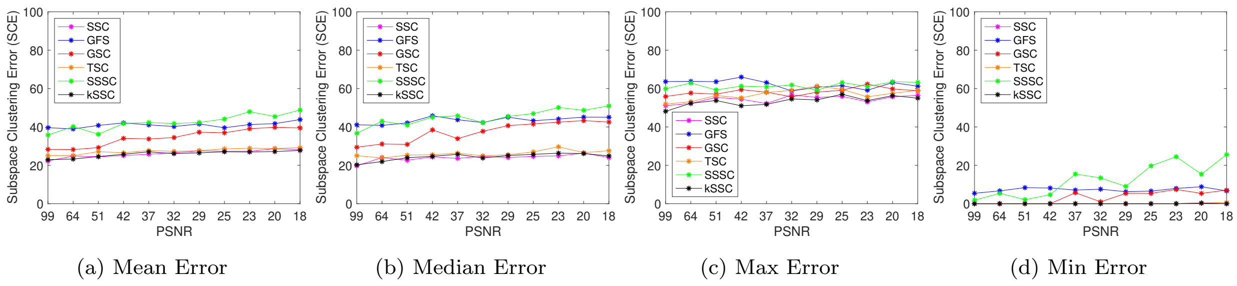


Figure 10: Running time of Large Scale Experiment. The size of the largest tested data set is 20,000.



(a) Mean Error (b) Median Error (c) Max Error (d) Min Error

Figure 11: Rigid Motion Segmentation

they do not scale well in this application (see Section 2). From Figure 10 we find that kSSC has similar run time characteristics to TSC and SSSC.

5.3. Hopkins 155 Motion Segmentation

The aim of this experiment is to assign feature points extracted from a video to their corresponding motion or object in the scene. As previously mentioned, it has been shown in [7] that these features trajectories actually correspond to low-dimensional subspaces. The data from this experiment is drawn from the rigid motion sequences of the Hopkins 155 dataset [33]. These sequences have around 200-500 feature trajectories and range in number of frames from 20-60. Results can be found in Figure 11. Again kSSC closely tracks the performance of SSC and consistently performs as the PSNR decreases.

5.4. Extended Yale B Face Clustering

The aim of this experiment is to cluster unique human subjects from a set of face images. We draw our data from the Exteded Yale Face Database B [9]. The dataset consists of approximately 64 photos of 38 subjects under varying illumination. We select three subjects randomly then resample their images to 96×84 and form data vectors $\mathbf{x}_i \in \mathbb{R}^{2016}$ by concatenating them together. This test was repeated 50

	Mean	Median	Min	Max	Std	Mean Run Time (s)
SSC	28.1%	31.5%	**0.0%**	65.6%	24.5%	21.38
GSC	30.6%	29.9%	0.5%	**55.2%**	15.6%	30.38
TSC	58.1%	61.2%	36.5%	66.1%	**7.3%**	7.59
SSSC	59.5%	58.6%	31.8%	100.0%	17.2%	**0.66**
kSSC	**22.3%**	**17.2%**	**0.0%**	65.1%	19.3%	30.38

Table 1: Face Clustering results from the Extended Yale B Dataset.

times with new random subjects each time. This is a challenging dataset since the original data is already corrupted by shadows from the varied illumination. Results can be found in Table 1. Surprisingly we find that kSSC even outperforms SSC for this task, which may be due to the aggressive kNN pre-screening process removing all but the most similar data points. In this dataset, there are many face images of different subjects that contain large regions of highly similar data due to the extreme occlusions from shadows. We believe the nearest neighbour filtering selection helps to prevent the possibility of extreme false positives connections in $\mathbf{Z}$. Note that in this case, kSSC is slower than SSC. This is caused by the overhead of kSSC pre-screening process using kNN. There is no computation saving when k and N are close.

6. Conclusion

In this paper we proposed a new algorithm, kSSC, to accurately and tractably approximate SSC for large scale datasets. By accurately screening out the vast majority of eligible data points as neighbours the memory and computational requirements are reduced from $\mathrm{O}\big(N^2\big)$ to $\mathrm{O}\big(N\big)$. Our theoretical analysis justifies the KNN screening process, which is able to find true neighbours from the same subspace with high probability. Moreover our empirical results on synthetic and real data demonstrate that kSSC outperforms the existing SSC approximation methods in terms of accuracy and matches or beats the computational and memory requirements.

Acknowledgment

The third authour is supported by the Australian Research Council (ARC) through grant DP140102270.

References

References

[1] Sunil Arya, David M Mount, Nathan S Netanyahu, Ruth Silverman, and Angela Y Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6):891–923, 1998.

[2] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Convex optimization with sparsity-inducing norms. *Optimization for Machine Learning*, pages 19–53, 2011.

[3] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal of Imaging Sciences*, 2(1):183–202, 2009.

[4] Nathanaël Berestycki. Concentration of measure . 2009.

[5] Eva L Dyer, Aswin C Sankaranarayanan, and Richard G Baraniuk. Greedy feature selection for subspace clustering. *The Journal of Machine Learning Research*, 14(1):2487–2517, 2013.

[6] Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797, 2009.

[7] Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[8] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the Nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

[9] Athinodoros S Georghiades, Peter N Belhumeur, and David 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.

[10] Y. Guo, S. Tierney, and J. Gao. Robust functional manifold clustering. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11, 2020.

[11] Yi Guo, Junbin Gao, and Feng Li. Spatial subspace clustering for drill hole spectral data. *Journal of Applied Remote Sensing*, 8(1):1–19, 2014.

[12] Yi Guo, Junbin Gao, Feng Li, Stephen Tierney, and Ming Yin. Low rank sequential subspace clustering. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015.

[13] Kaiming He and Jian Sun. Computing nearest-neighbor fields via propagation-assisted kd-trees. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 111–118. IEEE, 2012.

[14] Reinhard Heckel and Helmut Bölcskei. Robust subspace clustering via thresholding. *arXiv preprint arXiv:1307.4891*, 2013.

[15] Shuiwang Ji and Jieping Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the 26th annual international conference on machine learning*, pages 457–464. ACM, 2009.

[16] G. Liu, Z. Zhang, Q. Liu, and H. Xiong. Robust subspace clustering with compressed data. *IEEE Transactions on Image Processing*, 28(10):5161–5170, 2019.

[17] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, Jan 2013.

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

[19] Jun Liu and Jieping Ye. Efficient l1/lq norm regularization. *arXiv preprint arXiv:1009.4766*, 2010.

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

[21] Yi Ma, Allen Y Yang, Harm Derksen, and Robert Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM review*, 50(3):413–458, 2008.

[22] Boris Mailhé, Rémi Gribonval, Pierre Vandergheynst, and Frédéric Bimbot. Fast orthogonal sparse approximation algorithms over local dictionaries. *Signal Processing*, 91(12):2822–2835, 2011.

[23] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[24] Marius Muja and David G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36, 2014.

[25] Dohyung Park, Constantine Caramanis, and Sujay Sanghavi. Greedy subspace clustering. In *Advances in Neural Information Processing Systems*, pages 2753–2761, 2014.

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

[27] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[28] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

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

[30] Mahdi Soltanolkotabi, Ehsan Elhamifar, Emmanuel J Candes, et al. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

[31] Bob L Sturm and Mads Græsbøll Christensen. Comparison of orthogonal matching pursuit implementations. In *EUSIPCO*, pages 220–224, 2012.

[32] Stephen Tierney, Junbin Gao, and Yi Guo. Subspace clustering for sequential data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1019–1026, 2014.

[33] Roberto Tron and René Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[34] Joel A Tropp and Anna C Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on information theory*, 53(12):4655–4666, 2007.

[35] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[36] René Vidal. A tutorial on subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2011.

[37] Rene Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis (GPCA). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, 2005.

[38] Mehrdad Yaghoobi, Di Wu, and Mike E Davies. Fast non-negative orthogonal matching pursuit. *IEEE Signal Processing Letters*, 22(9):1229–1233, 2015.

[39] M. Yin, J. Gao, S. Xie, and Y. Guo. Multiview subspace clustering via tensorial t-product representation. *IEEE Transactions on Neural Networks and Learning Systems*, 30(3):851–864, 2019.

[40] Chong You, Chun-Guang Li, Daniel P Robinson, and Rene Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2016.

[41] Chong You, D Robinson, and René Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2016.

[42] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2004.

[43] Guang-Yu Zhang, Yu-Ren Zhou, Xiao-Yu He, Chang-Dong Wang, and Dong Huang. One-step kernel multi-view subspace clustering. *Knowledge-Based Systems*, 189:105126, 2020.

[44] Xin Zhang, Fuchun Sun, Guangcan Liu, and Yi Ma. Fast low-rank subspace segmentation. *IEEE Transactions on Knowledge and Data Engineering*, 26(5):1293–1297, May 2014.

Appendix A. Analysis of Subspace Discovery by kNN

We need a concentration of measure lower bound on sphere to start with.

Theorem 3. *Let $\mu(A_\delta)$ be the uniform probability measure on sphere $\mathbb{S}^{d-1}$ embedded in $\mathbb{R}^d$ and A_δ is a cap centered around a point $\mathbf{x}$ on $\mathbb{S}^{d-1}$, defined as $A_\delta = \{\mathbf{x}^\top \mathbf{y} \geq \delta | \mathbf{y} \in \mathbb{S}^{d-1}$ and $\delta \in [0,1]\}$. Then $\mu(A_\delta) \geq \frac{\pi(1-\delta)^{d-1}}{4(d-1)}$.*

Proof. We start with defining $A_r = \{d(\mathbf{x},\mathbf{y}) \leq r | \mathbf{y} \in \mathbb{S}^{d-1}$ and $\delta \in [0,1]\}$, where $d(\mathbf{x},\mathbf{y})$ is the geodesic metric on sphere. Write S_{d-1} the volume of $\mathbb{S}^{d-1}$. Then $\mu(A_r) = \frac{1}{S_{d-1}} \int_0^r \sin^{d-2}(x)dx$ as in [4]. Assume r is small, i.e. $r \in [0, \frac{\pi}{2}]$. Observe that $\sin(x) \geq \frac{2}{\pi}x$ when $x \in [0, \frac{\pi}{2}]$. Therefore,

$$\mu(A_r) \geq \frac{1}{S_{d-1}} \int_0^r (\frac{2}{\pi}x)^{d-2} dx = \frac{1}{S_{d-1}} \frac{(\frac{2}{\pi})^{d-2} r^{d-1}}{d-1}.$$

We then bound S_{d-1} from above. $S_{d-1} = \int_0^\pi \sin^{d-2}(x)dx$. Taking one $\sin(x)$ out and integrating by parts leads to $S_{d-1} = \frac{d-3}{d-2} S_{d-3}$ and apparently $S_{d-1} \geq S_{d-3}$. Use the inequality recursively and note that $S_3 = \frac{\pi}{2}$ and $S_2 = 2$, we have $S_{d-1} \leq 2$, which leads to

$$\mu(A_r) \geq \frac{(\frac{2}{\pi})^{d-2} r^{d-1}}{2d-2}.$$

Convert the metric to inner product as the following using $r = \text{acos}(\delta)$. $\mu(A_\delta) \geq \frac{(\frac{2}{\pi})^{d-2} \text{acos}^{d-1}(\delta)}{2d-2}$. Note that for $r \in [0, \frac{\pi}{2}]$ and $\delta \in [0,1]$, $\text{acos}(\delta) \geq \frac{\pi}{2}(1-\delta)$, therefore,

$$\mu(A_\delta) \geq \frac{\pi(1-\delta)^{d-1}}{4(d-1)}.$$

□

The above lower bound of measure for sphere cap can be used as the lower bound of the probability of points falling into A_δ when the points are uniformly distributed on $\mathbb{S}^{d-1}$, i.e.

$$P(\mathbf{y} \in A_\delta) \geq \frac{\pi(1-\delta)^{d-1}}{4(d-1)} = p_0. \tag{A.1}$$

Although it does not seem to be a large probability, when the number of points grows large, the number of samples falling into the same patch A_δ becomes large, as shown in the following lemma.

Lemma 4. *Let $\mathbf{y}$ be the random variable uniform distributed on sphere $\mathbb{S}^{d-1}$ embedded in $\mathbb{R}^d$ and A_δ is a cap centered around a point $\mathbf{x}$ on $\mathbb{S}^{d-1}$, when $N \geq \frac{4k_0(d-1)}{\pi(1-\delta)^{d-1}}$, the probability of k_0 points falling into A_δ is lower bounded, i.e.*

$$P(K > k_0) \geq 1 - \sum_{k=0}^{k_0} \binom{N}{k} p_0^k (1-p_0)^{N-k}, \tag{A.2}$$

where N is the total number of points.

Proof. First, it is easy to check that $f(x) = x^k(1-x)^{N-k}$ is monotonically decreasing when $\frac{k}{N} < x < 1$. Using probability bound in (A.1), we obtain that $P(K = k) \leq \binom{N}{k} p_0^k (1-p_0)^{N-k}$. Connect (A.1) with monotonicity requirement, we have $p_0 > \frac{k}{N}$ and the condition for the number of points as stated in the theorem, i.e. $N \geq \frac{4k_0(d-1)}{\pi(1-\delta)^{d-1}}$. Using $P(K > k_0) = 1 - \sum_{k=0}^{k_0} P(K = k)$, we complete the proof. $\square$

The probability seems complicated. Enlightened by the asymtotic approximation of Poisson to Binomial, we seek a proper λ for Poisson that is larger than Binomial with N and p_0. To this end, we first have the following lemma.

Lemma 5. *Given Binom(N,p) the binomial distribution with N trials and $p \geq 0$ success probability. When* $p \geq \left[2\frac{\sum_{i=0}^{k-1}\ln(N-i) - k\ln(N-k_0)}{N-k}\right]^{\frac{1}{2}} \equiv f(k)$,

$$\binom{N}{k} p^k (1-p)^{N-k} \leq e^{-(N-k_0)p} \frac{((N-k_0)p)^k}{k!} \tag{A.3}$$

for a given k such that $k \leq k_0$.

Proof. The R.H.S of Eq. (A.3) is $P(K = k)$ under Poisson distribution $Pois(\lambda)$ where $\lambda = (N - k_0)p$. We first observe that both sides of the inequality are non-negative so we can consider natural logarithm transform, i.e. $\ln(\cdot)$. Note that $\ln(1-x) = \sum_{n=1}^{\infty} -\frac{x^n}{n}$ and therefore, when $x > 0$, we have $\ln(1-x) \leq -x$ and $\ln(1-x) \leq -x - \frac{1}{2}x^2$.

When $k = 0$, the ln of L.H.S of (A.3) is $N\ln(1-p) \leq -Np \leq -(N-k_0)p$ which is the ln of the R.H.S of (A.3). So (A.3) holds when $p \geq 0$. When $k > 0$, we have

$$\begin{aligned}
\ln\left[\binom{N}{k} p^k (1-p)^{N-k}\right] &= \sum_{i=0}^{k-1} \ln(N-i) + (N-k)\ln(1-p) - \ln(k!) + k\ln(p) \\
&\leq \sum_{i=0}^{k-1} \ln(N-i) - (N-k)(p + \frac{1}{2}p^2) - \ln(k!) + k\ln(p) \\
&\leq \sum_{i=0}^{k-1} \ln(N-i) - \frac{1}{2}(N-k)p^2 - (N-k_0)p - \ln(k!) + k\ln(p) \\
&\leq k\ln(N-k_0) - (N-k_0)p - \ln(k!) + k\ln(p)
\end{aligned} \tag{A.4}$$

where in the second line we use $\ln(1-x) \leq -x - \frac{1}{2}x^2$ when $x > 0$ by series expansion of $\ln(1-x)$ at $x = 0$, and in the fourth line we apply the condition in the lemma to p^2 only. Note that the end of the above inequality is exactly the ln of the R.H.S of Eq. (A.3) . $\square$

This lemma says that $P(Binom(N,p) = k) \leq P(Pois((N-k_0)p) = k)$ when the conditions are satisfied in the Lemma especially the lower bound of p. However, the lower bound of p varies along k. Fortunately it is easy to find a unified lower bound as stated in the following corallary.

Corollary 6. *When* $p \geq f(k_0) = \left[2\frac{\sum_{i=0}^{k_0-1}\ln(N-i) - k_0\ln(N-k_0)}{N-k_0}\right]^{\frac{1}{2}}$, $P(Binom(N,p) = k) \leq P(Pois((N - k_0)p) = k)$ *for any k such that* $0 \leq k \leq k_0$.

Proof. Following from Lemma 5, $p \geq \max_{k\in\{0,\dots k_0\}} f(k)$ will satisfy all k that are no larger than k_0. Also

$$f(1) = \sqrt{\frac{2\ln(\frac{N}{N-k_0})}{N-1}} \leq \sqrt{\frac{2\ln(\frac{N}{N-k_0})}{N-2}} \leq \sqrt{\frac{2(\ln(\frac{N}{N-k_0}) + \ln(\frac{N-1}{N-k_0}))}{N-2}} = f(2).$$

By induction, we can easily show that $f(k)$ is non-decreasing along k. As such,

$$\max_{k\in\{0,\dots k_0\}} f(k) = f(k_0)$$

□

Corollary 6 shows that when the success probability p in binomial distribution $Binom(N, p)$ is not less than $f(k_0)$, binomial distribution can be upper bounded by Poisson distribution with $\lambda = (N - k_0)p$. Therefore, the lower bound in (A.2) in Lemma 4 can be replaced by the probabilities from Poisson distribution. However we need to balance the values of N and p_0. The the following lemma is for this purpose.

Lemma 7. *Let* $\mathbf{y}$ *be the random variable uniform distributed on sphere* $\mathbb{S}^{d-1}$ *embedded in* $\mathbb{R}^d$ *and* A_δ *is a cap centered around a point* $\mathbf{x}$ *on* $\mathbb{S}^{d-1}$*, when* $p_0 \geq \frac{k_0+1}{N-k_0}$ *and* $N \gg k_0$*, the probability of* k_1 *and* $k_1 \leq k_0$ *points falling into* A_δ *is lower bounded, i.e.*

$$P(K > k_1) \geq 1 - P(Pois((N - k_0)p_0) \leq k_1) \tag{A.5}$$

Proof. First notice that $\sum_{i=0}^{k-1} \ln(N-i) - k\ln(N-k_0) = \sum_{i=0}^{k} \ln(\frac{N-i}{N-k_0})$, and $\ln(1+x) \leq x$ when $x > 0$ using series expansion. Therefore,

$$\begin{aligned} f(k_1) \leq f(k_0) &= \left[2\frac{\sum_{i=0}^{k_0-1} \ln(N-i) - k_0\ln(N-k_0)}{N-k_0}\right]^{\frac{1}{2}} \leq \sqrt{\frac{2\sum_{i=0}^{k} \frac{k_0-i}{N-k_0}}{N-k_0}} \\ &= \sqrt{\frac{k_0(k_0+1)}{(N-k_0)^2}} \leq \frac{k_0+1}{N-k_0}. \end{aligned} \tag{A.6}$$

So when $p_0 \geq \frac{k_0+1}{N-k_0}$, the conditions in Lemma 6 are satisfied. Apparently, $\frac{k_0+1}{N-k_0} \geq \frac{k_0}{N}$, which enable us to combine Lemma 4 to give the following

$$P(K > k_1) \geq 1 - \sum_{k=0}^{k_1} P(Pois((N - k_0)p_0) = k),$$

which is the required result in this lemma. □

Lemma 7 shows that when the number of data points N is large, the probability of at least k_1 points fall into a small patch is lower bounded if $p_0 \geq \frac{k_0+1}{N-k_0}$. There is a lower bound for N as well written as N_0. When $N \geq N_0$, this probability lower bound can be large. We have the following theorem for this.

Theorem 8. *Assume the same settings and conditions in Lemma 7. Let* $N_0 = \frac{k_0+1}{p_0} + k_0 = \frac{4(d-1)(k_0+1)}{\pi(1-\delta)^{d-1}} + k_0$*, for any* $n \geq \frac{N_0}{N_0-k_0} > 1$*, and* $k_1 \leq k_0$*, if* $N = n(N_0 - k_0)$*, then*

$$P(K > k_1) \geq 1 - \exp\left(-n(k_0+1) + k_1(\ln\frac{n(k_0+1)}{k_1} + 1)\right) \tag{A.7}$$

Proof. It is easy to see that when $N \geq N_0$, the conditions in Lemma 7 are satisfied. Then (A.5) holds and $\lambda = (N - k_0)p_0 = n(k_0 + 1)$ in Poisson distribution according to the choice of N_0 and n. Applying Chernoff bound to the Poisson distribution completes the proof. $\square$

Let $k_1 = k_0$, we have the following corollary immediately.

Corollary 9. *Follow the same settings and conditions in Theorem 8. Let* $N_0 = \frac{k_0+1}{p_0} + k_0 = \frac{4(d-1)(k_0+1)}{\pi(1-\delta)^{d-1}} + k_0$*, for any* $n \geq \frac{N_0}{N_0 - k_0} > 1$*, if* $N = n(N_0 - k_0)$*, then*

$$P(K > k_0) \geq 1 - \exp\left(-k_0(n - \ln n + 1) - (n - 1)\right)$$

Proof. This is just result of replacing k_1 by k_0 in Theorem 8 and the application of $\ln(1 + x) \leq x$. $\square$

Remark. *k_0 is the upper bound for the number of nearest neighbours to be considered, which is used to construct the Poisson approximator. Apparently, k_0 is connected to N_0, the baseline number of data points required to have large probability. k_1 is the actual number of nearest neighbours in KNN searching. In real applications, one should have $k_1 = k_0$. See Corollary 9. Theorem 8 gives the values of all these variables and associated probabilities. The probability is ultimately determined by δ, d and n. Larger probability requires large N, and it grows exponentially with dimension d.*

Next we bound the inner product between samples from different subspaces. Here we use the arguments in Lemma 7.5 in [29].

Lemma 10. *Let* $\mathbf{A} \in \mathbb{R}^{d_1 \times N_1}$ *be a matrix with columns uniformly distributed in* $\mathbb{S}^{d_1 - 1}$*,* $\mathbf{y} \in \mathbb{R}^{d_2}$ *be a vector uniformly sampled from* $\mathbb{S}^{d_2-1}$ *and a deterministic matrix* $\mathbf{\Sigma} \in \mathbb{R}^{d_1 \times d_2}$*. For* $t > 0 \in \mathbb{R}$*, the inner product between any column in* $\mathbf{A}$ *and* $\mathbf{\Sigma y}$ *is bounded as follows*

$$\mathbf{a}_i^\top \mathbf{\Sigma y} \leq \frac{2\sqrt{t \log N_1 + t^2}\|\mathbf{\Sigma}\|_F}{\sqrt{d_1}}$$

with probability at least $1 - e^{-t}$.

Proof. Using Borell's inequality on the mapping $\mathbf{y} \mapsto \|\mathbf{\Sigma y}\|$ with Lipschitz constant of σ_1, the largest singular value of $\mathbf{\Sigma}$ leads to

$$P(\|\mathbf{\Sigma y}\| > \varepsilon + \sqrt{E\|\mathbf{\Sigma y}\|^2}) < e^{-\frac{1}{2}\varepsilon^2/\sigma_1^2}.$$

As $E\|\mathbf{\Sigma y}\|^2 = \|\mathbf{\Sigma}\|_F^2/d_2$, we choose $\varepsilon = (b-1)\|\mathbf{\Sigma}\|_F/\sqrt{d_2}$ so that

$$P(\|\mathbf{\Sigma y}\| > \frac{b\|\mathbf{\Sigma}\|_F}{\sqrt{d_2}}) < e^{-\frac{1}{2}(b-1)^2/d_2}, \tag{A.8}$$

where we used the fact that $\|\mathbf{\Sigma}\|_F/\sigma_1 > 1$.

The next step is to bound the inner product of a column in $\mathbf{A}$, i.e. $\mathbf{a}_i$, $i = 1 \ldots N_1$, with any vector $\mathbf{x} \in \mathbb{R}^{d_1}$ by upper bound of spherical caps

$$P(\mathbf{a}_i^\top \mathbf{x} > \varepsilon\|\mathbf{x}\|) < e^{-\frac{1}{2}d_1\varepsilon^2}, \ \forall i$$

which leads to the following using the union bound

$$P(\bigcup_i \mathbf{a}_i^\top \mathbf{x} > \varepsilon\|\mathbf{x}\|) < N_1 e^{-\frac{1}{2}d_1\varepsilon^2}. \tag{A.9}$$

Let $\varepsilon = \sqrt{\frac{2\log N_1 + 2t}{d_1}}$, $b = \sqrt{2d_2 t}$. Substituting (A.8) to (A.9) gives

$$P(\bigcup_i \mathbf{a}_i^\top \mathbf{\Sigma}\mathbf{y} > \frac{2\sqrt{t\log N_1 + t^2}\|\mathbf{\Sigma}\|_F}{\sqrt{d_1}}) \le e^{-t}.$$

Therefore

$$P(\bigcap_i \mathbf{a}_i^\top \mathbf{\Sigma}\mathbf{y} \le \frac{2\sqrt{t\log N_1 + t^2}\|\mathbf{\Sigma}\|_F}{\sqrt{d_1}}) \ge 1 - e^{-t},$$

which concludes the proof. □

The above gives the upper bound of the inner product, which connects to samples in subspaces with the following corollary.

Corollary 11. *Let $\mathbf{X} \in \mathbb{R}^{d\times N_1}$ be a matrix with columns formed by sampling uniformly from subspace $\mathcal{S}_1$ with dimensionality d_1 and $\mathbf{y} \in \mathbb{R}^d$ uniformly drawn from subspace $\mathcal{S}_2$ with dimensionality d_2. The inner product between any column in $\mathbf{X}$ and $\mathbf{y}$ is bounded as the following*

$$\mathbf{x}_i^\top \mathbf{y} \le \frac{2\mathcal{A}_{1,2}\sqrt{\min\{d_1, d_2\}(t\log N_1 + t^2)}}{\sqrt{d_1}}$$

with probability at least $1 - e^{-t}$ for t given previously.

Proof. This is the simple application of 10 with $\mathbf{\Sigma} = \mathbf{U}_1^\top \mathbf{U}_2$ where $\mathbf{U}_j$ $(j = 1, 2)$ is the orthonormal basis for subspace $\mathcal{S}_j$ and $\mathcal{A}_{1,2}$ is the affinity between subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$ described in Definition 1.2 in [30], which we recall here:

$$\mathcal{A}_{i,j} = \sqrt{\frac{\cos^2\theta^{(1)} + \cdots + \cos^2\theta^{(d_i\wedge d_j)}}{d_i \wedge d_j}}$$

where $\{\cos^2\theta^{(1)}, \ldots, \cos^2\theta^{(d_i\wedge d_j)}\}$ are the principal angles between subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$, and $d_i \wedge d_j$ stands for $\min\{d_i, d_j\}$. So $\|\Sigma\|_F = \|\mathbf{U}_1^\top \mathbf{U}_2\|_F = \mathcal{A}_{1,2}\sqrt{d_1 \wedge d_2}$. □

Without loss of generality, we consider a sample $\mathbf{x}_1$ from subspace $\mathcal{S}_1$. The following theorem ensures that kNN will find k nearest neighbors of $\mathbf{x}_1$ from $\mathcal{S}_1$ only.

Now we are ready to prove Theorem 1 the following.

Proof. This is a straightforward application of Lemma 9 and Collary 11 and the following. If

$$A_{\ell,1} \le \min_\ell \left\{ \frac{\sqrt{d_\ell}\delta}{2\sqrt{(d_1 \wedge d_\ell)(t\log N_\ell + t^2)}} \right\}$$

then

$$A_{\ell,1} \le \frac{\sqrt{d_\ell}\delta}{2\sqrt{(d_1 \wedge d_\ell)(t\log N_\ell + t^2)}}$$

and

$$\mathbf{x}^\top \mathbf{y} \le \frac{2\mathcal{A}_{\ell,1}\sqrt{\min\{d_1, d_\ell\}(t\log N_\ell + t^2)}}{\sqrt{d_\ell}} \le \delta,\ \forall \mathbf{x} \in \mathcal{S}_\ell.$$

Using union bound, we obtain the required probability. □

The above discussion deals with clean data only. In the following we show that the results are similar for noisey data as long as the noise level is not too great. We begin with the following series of lemmas with the noise assumed to be Gaussian.

Lemma 12. *Let random variables X and Y in $\mathbb{R}^d$ both be from Gaussian distribution $\mathcal{N}(0, \sigma^2\mathbf{I})$. For any given positive ϵ, we have*

$$P(|X^\top Y| > \epsilon) \le \frac{d\sigma^4}{\epsilon^2}.$$

Proof. First we assume X and Y are standard Gaussian, we have

$$P(|X^\top Y| > \epsilon) = P((X^\top Y)^2 > \epsilon^2) \le \frac{E(X^\top Y)^2}{\epsilon^2},$$

where the inequality is by Chernoff bound. Since both X and Y are both standard Gaussian, they are isotropic, so we have

$$E(X^\top Y)^2 = d.$$

The above can be obtained by

$$E(X^\top Y)^2 = E_X\{E_{X|Y}(X^\top y)^2\} = E_Y(|y|^2) = d,$$

where the first equality comes from law of iterative expectation, the second from isotropic property and the last from the fact that sum of standard Gaussian is Chi-square with d degrees of freedom.

After proper rescaling, we obtain the result in the lemma. □

Lemma 13. *Let $X \in \mathbb{R}^d$ be Gaussian random variable from $\mathcal{N}(0, \sigma^2\mathbf{I})$ and $\mathbf{y} \in \mathbb{R}^d$ be a fixed vector. The following holds with any positive ϵ*

$$p(|X^\top \mathbf{y}| > \epsilon) \le \exp(1 - \frac{c\epsilon^2}{\sigma^2\|\mathbf{y}\|_2^2}),$$

where c is a constant related to sub-Gaussian norm [35] of a standard Gaussian.

This is a straightforward application of sub-Gaussian tail to $X^\top \mathbf{y}$ with rescaling. By applying linear transformation to multivariate Gaussian distribution, we can also obtain

$$p(|X^\top \mathbf{y}| > \epsilon) = 2\Phi(\frac{\epsilon}{\sigma}),$$

where $\Phi()$ is probability function of standard Gaussian. Now we consider the inner product between two unitary vectors in subspaces with noise. We use the following model

$$\mathbf{Y} = \mathbf{X} + \mathbf{E} \tag{A.10}$$

where Y is the observation, X is the clear signal in some subspace and E is the noise assumed to be from $\mathcal{N}(0, \sigma^2\mathbf{I})$. We assume that the observations have been rescaled properly such that X is from a unit sphere in $\mathbb{S}^{d-1}$ and the variance of the noise is bounded.

First we note that under these conditions, the noise can increase or decrease inner product between observed signals by only a small amount, which is shown in the following lemma.

Lemma 14. *Let $\mathbf{y}_i$ $(i = 1, 2)$ be observations from the model in* (A.10)*, such that $\mathbf{y}_i = \mathbf{x}_i + \mathbf{e}_i$, $\|\mathbf{x}_i\|_2 = 1$ and $\mathbf{e}_i \sim \mathcal{N}(0, \sigma^2\mathbf{I})$. If $P(\mathbf{x}_1^\top \mathbf{x}_2 > v) \geq p$, we have*

$$P(\mathbf{y}_1^\top \mathbf{y}_2 > v - 3\epsilon) \geq p - 2\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - \frac{d\sigma^4}{\epsilon^2}.$$

If $P(\mathbf{x}_1^\top \mathbf{x}_2 < v) \geq p$, we have

$$P(\mathbf{y}_1^\top \mathbf{y}_2 < v + 3\epsilon) \geq p - 2\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - \frac{d\sigma^4}{\epsilon^2}.$$

Proof. We prove the $P(\mathbf{x}_1^\top \mathbf{x}_2 > v) \geq p$ case. The other cases can be proved similarly. Writing $\mathbf{y}_1^\top \mathbf{y}_2$ in terms and using triangular inequality gives

$$\mathbf{y}_1^\top \mathbf{y}_2 \geq \mathbf{x}_1^\top \mathbf{x}_2 - |\mathbf{x}_1^\top \mathbf{e}_2| - |\mathbf{e}_1^\top \mathbf{x}_2| - |\mathbf{e}_1^\top \mathbf{e}_2|.$$

Then

$$\begin{aligned} P(\mathbf{y}_1^\top \mathbf{y}_2 > v - 3\epsilon) &\geq P(\mathbf{x}_1^\top \mathbf{x}_2 > v \bigcap |\mathbf{x}_1^\top \mathbf{e}_2| > \epsilon \bigcap |\mathbf{e}_1^\top \mathbf{x}_2| > \epsilon \bigcap |\mathbf{e}_1^\top \mathbf{e}_2| > \epsilon) \\ &\geq p - 2\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - \frac{d\sigma^4}{\epsilon^2}. \end{aligned}$$

By using Lemma 12 and 13, we obtain the desired result. □

Lemma 14 states that the noise will dispel the vectors when they are very close and attract them when they are far away in terms of the inner product induced distance. The effect of noise for a given sample in subspace $\mathcal{S}_1$ is then to make the samples from other subspaces closer to it and more difficult to separate reflected by the reduced probability as shown in Theorem 2. We proceed with its proof as follows.

Proof. According to the model (A.10), $\mathbf{y}_1 = \mathbf{x}_1 + \mathbf{e}_1$ and $\mathbf{x}_1$ is on a unit sphere. From Lemma 9, we know that there are at least k_0 samples from $\mathcal{S}_1$ in the patch A_δ centred at $\mathbf{x}_1$ with probability at least $1 - \exp\left(-k_0(n - \ln n + 1) - (n - 1)\right)$. Combining this with Lemma 14 leads to the following

$$P(\min_{j \in \mathcal{N}_1} \{\mathbf{y}_1^\top \mathbf{y}_j\} \geq \delta - 3\epsilon) \geq 1 - \exp\left(-k_0(n - \ln n + 1) - (n - 1)\right) - 2\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - \frac{D\sigma^4}{\epsilon^2}$$

where $\mathcal{N}_1 \subset \mathcal{S}_1$ is the set of k_0 samples around $\mathbf{y}_1$ in A_δ patch.

Using Corollary 11 and Lemma 14 results in that with probability at least $1 - e^{-t} - 2\exp(1 - \frac{c\epsilon^2}{\sigma^2}) - \frac{D\sigma^4}{\epsilon^2}$

$$\mathbf{y}_i^\top \mathbf{y}_1 \leq \frac{2\mathcal{A}_{\ell,1}\sqrt{(d_1 \wedge d_\ell)(t \log N_\ell + t^2)}}{\sqrt{d_\ell}} + 3\epsilon$$

for any $\mathbf{y}_i$ from subspace $\mathcal{S}_\ell$.

Similar to Theorem 1, combing the above two statements, if

$$A_{\ell,1} \leq \min_{\ell} \left\{ \frac{\sqrt{d_\ell}(\delta - 6\epsilon)}{2\sqrt{(d_1 \wedge d_\ell)(t \log N_\ell + t^2)}} \right\}$$

then

$$\mathbf{y}_i^\top \mathbf{y}_1 \leq \frac{2\mathcal{A}_{\ell,1}\sqrt{(d_1 \wedge d_\ell)(t \log N_\ell + t^2)}}{\sqrt{d_\ell}} + 3\epsilon \leq \delta - 3\epsilon$$

with the probability stated in this theorem. □