ROBUST KERNEL SPARSE SUBSPACE CLUSTERING

Ivica Kopriva

Division of Computing and Data Science, Rudjer Boskovich Institute, Zagreb, Croatia

ABSTRACT

Kernel methods are widely used in pattern recognition, including subspace clustering (SC). They transform nonlinear problems in the input data space into linear ones in a high-dimensional feature space. This transformation, achieved through the kernel trick, makes computationally tractable nonlinear algorithms possible. However, kernelizing linear algorithms through the kernel trick is infeasible in case of gross sparse corruptions that are modeled by the ℓ_1-norm of the error term. To address this, we propose, for the first time, a robust kernel sparse SC (RKSSC) algorithm for data with gross sparse corruptions. We validated the proposed approach on two well-known datasets, using the linear robust SSC algorithm and nonlinear (kernel-based) SSC algorithm as baseline models. According to the Wilcoxon test, the RKSSC's algorithm clustering performance is statistically significantly better than baselines.

Index Terms— Robust kernel sparse subspace clustering, Nonlinear projection trick, Kernel Trick

1. INTRODUCTION

Clustering high-dimensional data into disjoint homogeneous groups is one of the fundamental problems in data analysis [2]. However, the high dimensionality of the ambient domain deteriorates clustering performance, a phenomenon related to the well-known curse of dimensionality. Consequently, identifying low-dimensional structures within a high-dimensional data is a critical challenge in engineering and mathematics [1]. In many applications, data can be well represented by a union of multiple linear subspaces, leading to linear subspace clustering (SC) [3]-[7]. However, real world data do not always conform to linear subspaces. One way to address this problem is by formulating SC algorithms in the reproducible kernel Hilbert space (RKHS) $\mathcal{H}$, also known as the feature space induced by mapping $\phi: \mathcal{X} \to \mathcal{H}$, instead of the original input space $\mathcal{X}$. This approach is justified by Cover's theorem [8], which states that the number of separating hyperplanes is proportional to the dimension of the data-generating space. Therefore, a large dimension of the feature space is crucial for making the data linearly separable. Working in high (possible infinite) dimensional feature space is computationally intractable. The standard solution to this problem is employment of the kernel trick [9]-[11]. However, linear SC algorithms that do not use the Frobenious norm of the error term, such as the robust version of the sparse SC (SSC) algorithm [5], cannot be kernelized [12]. Kernelization that accounts for outliers has been accomplished in [10] and [11] for the $\ell_{2,1}$-norm of the error term. However, as pointed out in [12], it remains an open problem how to kernelize linear algorithms based on the ℓ_1-norm of the error term that models gross sparse corruptions. Herein we focus this issue in formulating the robust kernel sparse SC (RKSSC) algorithm.

Instead of applying linear algorithms in an induced high dimensional space, it is possible to apply them in the empirical feature space [13]. This space is induced by an approximate explicit feature map that includes only a limited number of terms from ϕ [14], or by an empirical kernel map (EKM), see definition 3 in [15]. Notably, the kernel principal component analysis (KPCA) transform [16] is a special type of EKM [17][13]. This EKM was referred to as the nonlinear projection trick (NPT) in [18]. When data in the feature space are centered, NPT allows for the calculation of coordinates of mapped data with respect to the orthonormal basis of a subspace of the empirical feature space. Then, applying the linear method to these coordinates is equivalent to applying the kernel method to the original data [18]. Here, we follow a rigorous proof of equivalence between the linear robust ℓ_1-norm based SSC algorithm and its NPT-based kernel version, and derive an optimization method for the RKSSC algorithm. *We show that optimal performance of the RKSSC algorithm is not achievable by simply applying the linear SSC algorithm to NPT-mapped data. Instead, the derived optimization method reveals that the regularization constant in the in the NPT-induced space becomes a function of the square root of the rank of the centered kernel matrix*. The MATLAB code for the proposed RKSSC algorithm is available at: https://github.com/ikopriva/RKSSC.

The rest of the paper is organized as follows. In Section 2, we revisit the background and related work. Section 3 presents derivation of the robust kernel SSC algorithm. Experimental results are presented in Section 4, and conclusions are drawn in Section 5.

2. BACKGROUND AND RELATED WORK

Although, the nonlinear projection trick concept is presented in [18], it is necessary to understand this concept to follow the optimization procedure for the RKSSC algorithm presented in Section 3. Thus, we reproduce the basic derivations behind the NPT.

Let $\mathbf{X} \triangleq [\mathbf{x}_1,...,\mathbf{x}_N] \in \mathbb{R}^{D\times N}$ represent the in-sample (training) dataset comprised of N samples in a D-dimensional input space. Let $\Phi(\mathbf{X}) \triangleq [\phi(\mathbf{x}_1),...,\phi(\mathbf{x}_N)] \in \mathbb{R}^{F\times N}$ stand for the in-sample zero-mean data in the F-dimensional feature space. If $\Psi(\mathbf{X})$ denotes the non-zero mean version of $\Phi(\mathbf{X})$, we obtain $\Phi(\mathbf{X})$ through:

$$\Phi(\mathbf{X}) = \Psi(\mathbf{X})(\mathbf{I}_N - \mathbf{E}_N) \tag{1}$$

where $\mathbf{I}_N$ is an $N\times N$ identity matrix, and $\mathbf{E}_N \triangleq \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^{\mathrm{T}}$ is an $N\times N$ matrix with all elements equal to $1/N$. Let $\mathbf{K} \triangleq \Phi(\mathbf{X})^{\mathrm{T}}\Phi(\mathbf{X}) \in \mathbb{R}^{N\times N}$ be the kernel (Gramm) matrix of the in-sample data. The rank of $\mathbf{K}$ is R. Because $\Phi(\mathbf{X})$ is centered, it follows that $R\le N$-1. Let $\mathcal{K} \triangleq \Psi(\mathbf{X})^{\mathrm{T}}\Psi(\mathbf{X})$ be the non-centered kernel matrix. We obtain the centered kernel matrix as:

$$\mathbf{K} = (\mathbf{I}_N - \mathbf{E}_N)\mathcal{K}(\mathbf{I}_N - \mathbf{E}_N)\ . \tag{2}$$

Let the kernel vector associated with an arbitrary $\mathbf{x} \in \mathbb{R}^D$ be defined as:

$$k(\mathbf{x}) \triangleq \Phi(\mathbf{X})^{\mathrm{T}}\phi(\mathbf{x}) \in \mathbb{R}^N\ . \tag{3}$$

where $k(\mathbf{x})$ is the EKM with regard to the in-sample data $\mathbf{X}$ (see definition 3 in [15]). Let $\kappa(\mathbf{x})$ be the non-centered kernel vector. We obtain the centered one as:

$$k(\mathbf{x}) = (\mathbf{I}_N - \mathbf{E}_N)\left[\kappa(\mathbf{x}) - \tfrac{1}{N}\mathcal{K}\mathbf{1}_N\right]. \tag{4}$$

Let $\mathcal{P}$ be the R-dimensional subspace of the feature space formed by $\Phi(\mathbf{X})$. Let $\phi_{\mathbf{w}}(\mathbf{x})$ be the projection of $\phi(\mathbf{x})$ onto 1-D vector space formed by $\mathbf{w} \in \mathbb{R}^F$. We restrict $\mathbf{w}$ to be in $\mathcal{P}$, i.e., $\mathbf{w}=\Phi(\mathbf{X})\boldsymbol{\alpha}$ for some $\boldsymbol{\alpha} \in \mathbb{R}^N$. Let $\mathbf{K}$ have the eigenvalue decomposition $\mathbf{K} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}$, with $\boldsymbol{\Lambda}_R = \mathrm{diag}(\lambda_1,...,\lambda_R)$ being the leading R eigenvalues and $\mathbf{U}_R$ containing the corresponding R eigenvectors. The orthonormal basis of $\mathcal{P}$ is obtained as:

$$\boldsymbol{\Pi} \triangleq \Phi(\mathbf{X})\mathbf{U}_R\boldsymbol{\Lambda}_R^{-1/2} = [\boldsymbol{\pi}_1,...,\boldsymbol{\pi}_R] \in \mathbb{R}^{F\times R}\ . \tag{5}$$

We obtain the coordinates of the mapped training data $\Phi(\mathbf{X})$ in the Cartesian coordinate system spanned by $\boldsymbol{\Pi}$ as:

$$\mathbf{Y} = \boldsymbol{\Lambda}_R^{-1/2}\mathbf{U}_R^{\mathrm{T}}\mathbf{K} = \boldsymbol{\Lambda}_R^{1/2}\mathbf{U}_R^{\mathrm{T}} \in \mathbb{R}^{R\times N}. \tag{6}$$

We obtain the coordinate of the projection of any $\mathbf{x} \in \mathbb{R}^D$ onto $\mathcal{P}$ in the same coordinate system as:

$$\mathbf{y} = \boldsymbol{\Pi}^{\mathrm{T}}\phi(\mathbf{x}) = \boldsymbol{\Lambda}_R^{-1/2}\mathbf{U}_R^{\mathrm{T}}k(\mathbf{x})\ . \tag{7}$$

Using Lemma 1 from [18], we obtain the coordinates of the projection vector $\mathbf{w}$ in $\mathcal{P}$ as:

$$\mathbf{w} = \Phi(\mathbf{X})\boldsymbol{\alpha} = \boldsymbol{\Pi}\boldsymbol{\beta} \tag{8}$$

where $\boldsymbol{\beta} = \mathbf{Y}\boldsymbol{\alpha} \in \mathbb{R}^R$. Given that we now have the coordinates of the in-sample data (6), and the out-of-sample (test) data (7) in the empirical feature space, applying the kernel method to the in-sample data $\mathbf{X}$ is equivalent to applying the linear method to their coordinates $\mathbf{Y}$. The same applies to the out-of-sample data $\mathbf{x}$ and its coordinates $\mathbf{y}$. In this regard, we point out that EKM (7) is the same as the KPCA transform given by equation (34) in [17]. In other words, KPCA can be implemented in a new, more interpretable way by applying PCA on coordinates (6)/(7) [18].

3. ROBUST KERNEL SPARSE SUBSPACE CLUSTERING

We now present an optimization method for deriving the RKSSC algorithm. The linear SSC algorithm derived in [5] is obtained as the solution of the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \lambda_e\|\mathbf{E}\|_1 + \frac{\lambda_z}{2}\|\mathbf{Z}\|_F^2 \\ &\text{s.t.}\,\mathbf{X} = \mathbf{X}\mathbf{C} + \mathbf{E} + \mathbf{Z},\ \mathrm{diag}(\mathbf{C}) = 0. \end{aligned} \tag{9}$$

In the above equation, $\mathbf{E} \in \mathbb{R}^{N\times N}$ models sparse gross corruption and $\mathbf{Z} \in \mathbb{R}^{N\times N}$ models Gaussian noise. The term $\mathbf{C} \in \mathbb{R}^{N\times N}$ stands for the representation matrix in the self-expressive model $\mathbf{X}=\mathbf{X}\mathbf{C}$, and diag($\mathbf{C}$)=0 constraint is employed to prevent the trivial solution where each data point is represented by itself. By setting $\lambda_e = 0$, we obtain the SSC objective function optimized for additive Gaussian noise. The linear SSC algorithm is derived in [5] and the corresponding nonlinear kernel SSC algorithm is derived in [9]. Here, we focus on a robust nonlinear kernel-based SSC algorithm. By setting $\lambda_z = 0$ to model gross sparse corruptions in (9), we obtain the robust SSC objective function:

$$\begin{aligned} &\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \lambda_e\|\mathbf{X} - \mathbf{X}\mathbf{C}\|_1 \\ &\text{s.t.}\,\mathbf{X} = \mathbf{X}\mathbf{C} + \mathbf{E},\ \mathrm{diag}(\mathbf{C}) = 0. \end{aligned} \tag{10}$$

By mapping data into empirical features space $\mathbf{X} \to \Phi(\mathbf{X})$, equation (10) becomes:

$$\begin{aligned} &\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \lambda_e\|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{C}\|_1 \\ &\text{s.t.}\,\Phi(\mathbf{X}) = \Phi(\mathbf{X})\mathbf{C} + \mathbf{E},\ \mathrm{diag}(\mathbf{C}) = 0. \end{aligned} \tag{11}$$

It is evident that the error term in (11) cannot be expressed solely in terms of the inner product $\Phi(\mathbf{X})^{\mathrm{T}}\Phi(\mathbf{X})$. Thus, the *kernel trick* cannot be employed to obtain the nonlinear kernel-based version of the linear SSC algorithm robust to gross sparse corruptions. According to [18], the RKSSC algorithm is obtained by applying the existing linear robust

SSC algorithm to the coordinates $\mathbf{Y}$ in equation (4) of the data in the empirical feature space. Consequently, the objective function (10) becomes:

$$\begin{aligned} &\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \lambda_e \|\mathbf{Y} - \mathbf{YC}\|_1 \\ &\text{s.t.}\, \mathbf{Y} = \mathbf{YC} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0. \end{aligned} \tag{12}$$

As the subsequent derivations will show, the issue with (12) is that the regularization constant λ_e cannot be properly tuned. This is because, in the mapped space, λ_e becomes a function of $\sqrt{R}$. That will be demonstrated shortly. Following [18], we re-formulate the optimization problem (11) in the projected 1-D vector space:

$$\begin{aligned} &\min_{\mathbf{C},\mathbf{w}} \|\mathbf{C}\|_1 + \lambda_e \left\|\mathbf{w}^{\mathrm{T}} \left(\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{C}\right)\right\|_1 \\ &\text{s.t.}\, \Phi(\mathbf{X}) = \Phi(\mathbf{X})\mathbf{C} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0 \ \text{ and } \ \|\mathbf{w}\|_2 = 1. \end{aligned} \tag{13}$$

Using (8) and the fact that $\mathbf{\Pi}$ is orthonormal, it follows $\|\mathbf{w}\|_2 = 1 \Rightarrow \|\boldsymbol{\beta}\|_2 = 1$. Now, bearing in mind that $\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{X}) = \boldsymbol{\beta}\mathbf{Y}$ [18], equation (13) becomes

$$\begin{aligned} &\min_{\mathbf{C},\boldsymbol{\beta}} \|\mathbf{C}\|_1 + \lambda_e \left\|\boldsymbol{\beta}^{\mathrm{T}} (\mathbf{Y} - \mathbf{YC})\right\|_1 \\ &\text{s.t.}\, \mathbf{Y} = \mathbf{YC} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0 \ \text{ and } \ \|\boldsymbol{\beta}\|_2 = 1. \end{aligned}$$

By using the matrix norm inequality $\|\mathbf{Ab}\|_1 \le \|\mathbf{A}\|_1 \|\mathbf{b}\|_1$ from equation (2.3.4) in [20], we obtain:

$$\begin{aligned} &\min_{\mathbf{C},\boldsymbol{\beta}} \|\mathbf{C}\|_1 + \lambda_e \|(\mathbf{Y} - \mathbf{YC})\|_1 \|\boldsymbol{\beta}\|_1 \\ &\text{s.t.}\, \mathbf{Y} = \mathbf{YC} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0 \ \text{ and } \ \|\boldsymbol{\beta}\|_2 = 1. \end{aligned}$$

Applying the inequality between vector norms $\|\boldsymbol{\beta}\|_2 \le \|\boldsymbol{\beta}\|_1 \le \sqrt{R}\,\|\boldsymbol{\beta}\|_2$ from equation (2.2.5) in [20], we obtain:

$$\begin{aligned} &\min_{\mathbf{C},\boldsymbol{\beta}} \|\mathbf{C}\|_1 + \sqrt{R}\,\lambda_e \|(\mathbf{Y} - \mathbf{YC})\|_1 \|\boldsymbol{\beta}\|_2 \\ &\text{s.t.}\, \mathbf{Y} = \mathbf{YC} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0 \ \text{ and } \ \|\boldsymbol{\beta}\|_2 = 1. \end{aligned} \tag{14}$$

Since the minimum of (14) is constrained with $\|\boldsymbol{\beta}\|_2 = 1$, from the optimization point of view, we obtain the equivalent formulation of (14) as:

$$\begin{aligned} &\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \underbrace{\sqrt{R}\,\lambda_e}_{\bar{\lambda}_e} \|(\mathbf{Y} - \mathbf{YC})\|_1 \\ &\text{s.t.}\, \mathbf{Y} = \mathbf{YC} + \mathbf{E},\ \text{diag}(\mathbf{C}) = 0. \end{aligned} \tag{15}$$

The important difference between optimization problems (15) and (12) lies in regularization constant. *The regularization constant in (15), $\bar{\lambda}_e$, is proportional to the square root of the rank of the kernel matrix* $\mathbf{K}$, i.e., $\bar{\lambda}_e \propto \sqrt{R}$. Since, along with the kernel parameters, R is an additional hyperparameter selected via cross-validation it will influence the value of $\bar{\lambda}_e$. In other words, the regularization constant $\bar{\lambda}_e$ can not be adjusted independently of R. That is why the direct application of the linear robust SSC algorithm in [5] to equation (12) will yield suboptimal results.

The RKSSC algorithm yields an estimate of the self-representation matrix $\mathbf{C}$, from which we calculate the data affinity matrix:

$$\mathbf{A} = \frac{|\mathbf{C}| + |\mathbf{C}|^{\mathrm{T}}}{2}. \tag{16}$$

Using (16), we compute the normalized graph Laplacian matrix [21]:

$$\mathbf{L} = \mathbf{I} - (\mathbf{D})^{-1/2}\,\mathbf{A}\,(\mathbf{D})^{-1/2} \tag{17}$$

where the elements of the diagonal degree matrix are given with the sum of rows of $\mathbf{A}$, i.e., $\mathbf{D}_{ii} = \sum_{j=1}^{N} \mathbf{A}_{ij}$. The spectral clustering algorithm [22] is then applied to $\mathbf{L}$ to assign the cluster labels $\mathbf{F} \in \mathbb{N}_0^{N \times C}$ to data points, where C denotes the number of clusters. We summarize the RKSSC algorithm in Algorithm 1.

Algorithm 1: Robust KSSC (RKSSC)

Input: Data $\mathbf{X} \in \mathbb{R}^{D \times N}$, number of clusters C, R, $\bar{\lambda}_e \propto \sqrt{R}$, parameters of the selected kernel function $\kappa(\cdot,\cdot)$.

Output: Assigned cluster indicator matrix $\mathbf{F} \in \mathbb{N}_0^{N \times C}$.

Step 1: Compute the uncentered kernel matrix $\mathcal{K} = \kappa(\mathbf{X}, \mathbf{X})$.

Step 2: Compute the centered kernel matrix through: $\mathbf{K} = (\mathbf{I}_N - \mathbf{E}_N)\,\mathcal{K}\,(\mathbf{I}_N - \mathbf{E}_N)$.

Step 3: Compute the eigenvalue decomposition of $\mathbf{K}$ such that $\mathbf{K} \leftarrow \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}$, with $\mathbf{\Lambda}_R = \text{diag}(\lambda_1, ..., \lambda_R)$ being the leading R eigenvalues and $\mathbf{U}_R$ containing the corresponding R eigenvectors.

Step 4: Compute the coordinates: $\mathbf{Y} = \Lambda_R^{1/2}\mathbf{U}_R^{\mathrm{T}}$.

Step 5: Apply the robust SSC algorithm [5] to $\mathbf{Y}$ to estimate the self-representation matrix $\mathbf{C}$ using the regularization constant $\bar{\lambda}_e \propto \sqrt{R}$.

Step 6: Calculate the data affinity matrix $\mathbf{A}$ using (16), the normalized Laplacian matrix $\mathbf{L}$ using (17), and apply spectral clustering on $\mathbf{L}$ to assign cluster labels $\mathbf{F} \in \mathbb{N}_0^{N \times C}$ to data points $\mathbf{X}$.

Algorithm 1 works on the in-sample (training) data with time complexity $O(N^3)$ and space complexity $O(N^2)$. To cluster out-of-sample (test) data every data point is mapped through equation (7). From clustering partitions obtained on training set we estimate the orthonormal bases from the first d left singular vectors of each partition. Afterward, we assign label to a test data according to a minimum point-to-a-subspace distance criterion [28].

Here d represents an *a-priori* known subspace dimension. Bases estimation sets requirement on minimal size of the training dataset as: $N \geq C \times d$. Because typically d and C are small, d=9 for face images [29], and d=12 for handwritten digits [30], hyperparameters tuning can be performed on reasonably small training set. The rest of data form an arbitrarily large test data the size of which does not affect the hyperparameters tuning process.

4. EXPERIMENTAL RESULTS

We compared the proposed RKSSC algorithm with the linear robust SSC (RSSC) algorithm [5] and non-robust nonlinear kernel SSC (KSSC) algorithm [9] as baselines. For this purpose, we used the Extended Yaleb (EYaleb) dataset [24] and MNIST [25] dataset. The EYaleb dataset contains 38 groups of face images, with 64 images per group. We clustered all 38 groups, i.e., C=38. The MNIST dataset contains 10 groups of handwritten digits with 1000 images per group. We clustered all 10 groups, i.e., C=10. Regarding the kernel function, we used the polynomial kernel: $\kappa_{poly}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left(\left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle + b\right)^d$, and the Gaussian kernel: $\kappa_{Gauss}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp^{-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2 / \left(2\sigma^2\right)}$. Along with the kernel rank R, (b, d) or σ^2 are hyperparameters introduced by kernelization. They are tuned on randomly generated subsets containing 46 and 200 data samples per group from the EYaleb and MNIST datasets, respectively. Their values are reported in publicly available MATLAB code. Testing subsets contained 19 and 200 data samples in respective order. We used accuracy (ACC), normalized mutual information (NMI), and F_1 score as performance measures. We validated them on 100 randomly generated in-sample and out-of-sample subsets. The in-sample subsets contained the same number of samples as those used for hyperparameters tuning. In terms of preprocessing, we normalized all data to unit-column ℓ_2 norm prior to further processing. We conducted statistical significance analysis using the Wilcoxon sum rank test. The obtained results are shown in Table 1 for the EYaleb dataset and Table 2 for the MNIST dataset. Performance on in-sample data is given in the first row, and out-of-sample data in the second row. RKSSC yielded statistically significantly better results than the linear RSSC algorithm [5]. For the EYaleB dataset, it also outperforms significantly nonlinear KSSC algorithm [9]. For the MNIST dataset, the RKSSC algorithm and KSSC algorithm yield comparative results. To further emphasize the quality of the results obtained by the RKSSC algorithm, we report results achieved by autoencoder (AE)-based deep networks [26] in Table 1, and for deep embedding for clustering (DEC) network [27] in Table 2.

Table 1. Clustering performance on the EYaleb dataset.

Algorithm	ACC [%]	NMI [%]	F_1[%]
RSSC	75.65±1.84	80.48±1.65	40.77±4.39
	81.36±2.17	85.79±1.67	58.73±3.81
RKSSC Gauss	80.63±1.78	84.78±0.09	68.01±2.63
	81.72±1.84	86.62±1.09	70.73±2.49
p values vs. RSSC	4.41×10^{-30}	1.05×10^{-33}	2.56×10^{-34}
	0.1801	1.33×10^{-4}	7.99×10^{-34}
RKSSC poly	**81.74±1.73**	**85.59±0.91**	**70.51±2.59**
	82.03±1.83	**86.97±1.12**	**71.95±2.59**
p values vs. RSSC	2.28×10^{-32}	3.56×10^{-34}	2.56×10^{-34}
	0.0435	5.54×10^{-8}	4.02×10^{-34}
KSSC Gauss	71.17±1.51	75.78±1.06	42.42±2.14
	77.63±1.84	82.76±1.20	59.00±2.97
AE [26]	84.73	86.75	-

Table 2. Clustering performance on the MNIST dataset.

Algorithm	ACC [%]	NMI [%]	F_1[%]
RSSC	60.25±4.89	62.10±2.95	51.31±3.86
	60.30±4.35	60.40±2.36	52.52±3.43
RKSSC Gauss	**64.57±2.72**	62.94±2.03	**64.57±2.72**
	65.18±2.83	**64.02±2.09**	**65.18±2.83**
p values vs. RSSC	2.05×10^{-11}	3.09×10^{-2}	9.29×10^{-18}
	1.13×10^{-13}	6.56×10^{-20}	4.50×10^{-17}
RKSSC poly	62.49±2.48	62.11±1.72	54.44±2.19
	59.27±2.84	58.64±1.89	47.83±2.70
p values vs. RSSC	9.97×10^{-5}	0.9971	2.46×10^{-10}
	0.1136	1.68×10^{-7}	5.57×10^{-18}
KSSC Gauss	63.68±3.53	**63.53±2.28**	55.76±2.27
	63.83±3.55	63.45±2.33	55.89±2.91
DEC [27]	61.20	57.53	-

5. CONCLUSIONS

In this paper, we introduced the robust kernel sparse subspace clustering (RKSSC) algorithm for nonlinear SC of data contaminated by gross sparse corruptions. In such cases, the error term in the related linear optimization problem is modeled by the ℓ_1-norm, and kernelization based on a *kernel trick* cannot be directly applied. By centering data in the empirical feature space (EFS) induced by the kernel-based mapping, the use of a nonlinear projection trick enables us to calculate the coordinates of mapped data with respect to the orthonormal basis of a subspace of the EFS. The RKSSC is obtained by applying the existing linear RSSC to the coordinates of data mapped in the EFS. However, derived optimization procedure reveals that the regularization constant of the RKSSC algorithm is proportional to the square root of the rank of the centered kernel matrix and, therefore, has to be tuned properly.

6. ACKNOWLEDGMENTS

This work is supported by the Croatian Science Foundation grant IP-2022-10-6403.

7. REFERENCES

[1] J. Wight and Y. Ma, *High-Dimensional Data Analysis with Low-Dimensional Models - Principles, Computation and Applications*, Cambridge University Press, Cambridge, UK, 2022.

[2] A. K. Jain, "Data clustering: 50 years beyond *k*-means," *Pattern Rec. Lett.*, vol. 31, no. 8, pp. 651-666, 2010.

[3] R. Vidal, "Subspace clustering", *IEEE Signal Process. Mag*., vol. 28, no. 2, pp. 52-68, 2011.

[4] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, 2013.

[5] E. Elhamifar and R. Vidal, "Sparse Subspace Clustering: Algorithm, Theory, and Applications," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 35, no. 1, pp. 2765-2781, 2013.

[6] M. Brbić and I. Kopriva, "ℓ_0 Motivated Low-Rank Sparse Subspace Clustering," *IEEE Trans. Cyber.*, vol. 50, no. 4, pp. 1711-1725, 2020.

[7] C. Lu, J. Feng, Z. Lin, T. Mei, and S. Yan, "Subspace Clustering by Block Diagonal Representation," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 41, no. 2, pp. 487-501, 2018.

[8] T. M. Cover, "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition," *IEEE Trans. Electronic Computers,* vol. EC-14, no. 3, pp. 326-334, 1965.

[9] V.M. Patel and R.Vidal, "Kernel sparse subspace clustering," in: *IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 2849–2853.

[10] S. Xiao, M. Tan, D. Xu, Z.Y. Dong, "Robust kernel low-rank representation", *IEEE Trans. Neural Netw. Learn. Sys*t., vol. 27, no. 11, pp. 2268–2281, 2015.

[11] Y. Xie, J. Liu, Y. Qu, D. Tao, W. Zhang, L. Dai, and L. Ma, "Robust Kernelized Multiview Self-Representation for Subspace Clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 868-881, 2021.

[12] M. Abdolali and N. Gillis, "Beyond linear subspace clustering: A comparative study of nonlinear manifold clustering algorithms," *Computer Science Review*, vol. 42, article no. 100435, 2021.

[13] H. Xiong, N. S. Swamy, and M. O. Ahmad, "Optimizing the Kernel in the Empirical Feature Space," *IEEE Trans. Nerual Netw.*, vol. 16, no. 2, pp. 460-474, 2005.

[14] A. Cotter, J. Keshet and N. Srebro, "Explicit approximation of the Gaussian kernel," in CoRR abs/1109.4603 (2011). arXiv: 1109.4603

[15] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. R. Müller, G. Rätsch, and A. J. Smola, "Input Space Versus Feature Space in Kernel-Based Methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000-1017, 1999.

[16] B. Schölkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalues problem," *Neural Comput.*, vol. 10, pp. 1299-1319, 1998.

[17] A. Ruiz and P. E. López-de-Teruel, "Nonlinear Kernel-Based Statistical Pattern Analysis," *IEEE Trans. Neural Netw.*, vol. 12, no. 1, pp. 16-32, 2001.

[18] N. Kwak, "Nonlinear Projection Trick in Kernel Methods: An Alternative to the Kernel Trick," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 12, pp. 2113-2119, 2013.

[19] N. Kwak, "Principal component analysis based on l1 norm maximization," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 30, no. 9, pp. 1672-1680, 2008.

[20] G. H. Golub and C. F. Van Loan, Matrix Computtaion, 4th Edition, John Hopkins University Press, 2013.

[21] A. Khan, and P. Maji, "Approximate Graph Laplacians for Multimodal Data Clustering," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 43, no. 3, pp. 798-813, 2021.

[22] U. von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

[23] P. A. Traganitis and G. B. Giannakis, "Sketched Subspace Clustering," *IEEE Trans. Sig. Proc.*, vol. 66, 1663-1675, 2018.

[24] 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, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, 2001.

[25] J. J. Hull, "A database for handwritten text recognition research," I*EEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[26] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J. T. Zhou, and S. Yang, "Structured autoencoders for subspace clustering," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5076-5086, 2018.

[27] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in ICML. PMLR, 2016, pp. 478–487.

[28] P. A. Traganitis and G. B. Giannakis, "Sketched Subspace Clustering," *IEEE Trans. Sig. Proc.*, vol. 66, 1663-1675, 2018.

[29] K.-C. Lee, J. Ho, and D. Kriegman, "Acquiring Linear Subspaces for Face Recognition under Variable Lighting," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684-698, 2005.

[30] T. Hastie and P. Y. Simard, "Metrics and models for handwritten character recognition," in *Proc. Stat. Sci.*, pp. 54–65, 1998.