



# **VARIATIONS ON STRUCTURED SPARSITY FOR MACHINE LEARNING**

**YİĞİT OKTAR**

**Ph.D. Thesis**

**Graduate School Tzmir University of Economics Izmir 2020**

# **VARIATIONS ON STRUCTURED SPARSITY FOR MACHINE LEARNING**


**YİĞİT OKTAR**

**A Thesis Submitted to The Graduate School of İzmir University of  
Economics Ph.D. in Computer Engineering**

**İzmir  
2020**

Approval of the Graduate School

I certify that this thesis satisfies all the requirements as a thesis for a Ph.D. degree.

 Haya Oğuz He^tf of Division  
Director

/TJ

Asst. Prof./Dr. Haya Oğuz He^tf of Division

This is to certify that we have read this thesis and that in our opinion it is folly adequate, in scope and quality, as a thesis for a Ph.D. degree.



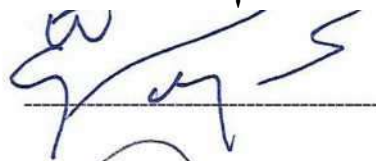
Ph.D. Exam Jury Members

Prof. Dr. Ender Mete Ekşioğlu

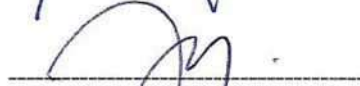


Prof. Dr. Türker İnce


Assoc. Prof. Dr. Behçet Uğur Töreyn



Asst. Prof. Dr. Kaya Oğuz



Asst. Prof. Dr. Mehmet Türkan



# **ABSTRACT**

## **VARIATIONS ON STRUCTURED SPARSITY FOR MACHINE LEARNING**

**Oktar, Yiğit**

**Ph.D. in Computer Engineering Advisor: Asst. Prof. Dr. Mehmet Türkan August,**

**2020**

**Dictionary learning is conventionally utilized as a feature learning method. Such framework is commonly used in reconstructive signal processing tasks. Learnt features can also be used as inputs to further classification and clustering schemes. Using block-sparsity, sparse framework can be cast as a clustering problem directly. In its conventional form, learning of linearly non-separable cases is not possible, due to inability of distinguishing two classes within the same subspace. With sum-to-one and non-negativity constraints on the sparse codes and still assuming block-sparsity, one can arrive at superproblems of k-means, called k-flats, k-simplexes, and k-polytopes. A polytope is defined to be an intact object composed of many same dimensional Simplexes. K-polytopes experimentally reaches the capacity of ensemble k-means and surpasses the capacity of kernel k-means. K-polytopes is further generalized through the concept of simplicial learning cast as a one-class learning method, in which intact**



ness is dropped and heterogeneous dimensionality is allowed. Due to combinatorial nature of the problem, an evolutionary approach is taken. Such adaptation solves linearly non-separable cases easily and appears to be a reliable method. Still an important shortcoming remains due to assuming orthogonality of dimensions. Convolution is a practical solution to the problem of orthogonality presented. Using convolutional case, a shift-invariant k-means version is formulated and unsupervised feature learning performance of convolutional dictionary learning is evaluated. With these new modifications and considerations, sparse and redundant representations framework appears to be a crucial tool for machine learning.

**Keywords:** Sparsity, Polytope, Simplicial, Convolution, Machine learning

## **ÖZET**

### **MAKİNE ÖĞRENİMİ İÇİN YAPISAL SEYREKLİK ÜZERİNE ÇEŞİTLEMELER**

**Oktar, Yiğit**

**Bilgisayar Mühendisliği Doktora Programı Danışman: Dr. Öğr. Üyesi Mehmet**

**Türkan Ağustos, 2020**

Seyrek ve bol gösterimler için sözlük öğrenimi genelde bir öznitelik öğrenimi yöntemidir. Bu yöntem yapıcı sinyal işleme uygulamalarında sıkça kullanılır. Öğrenilen öznitelikler, makine öğrenimi için sınıflandırma ve kümeleme yöntemlerine de

girdi olarak verilebilir. Kalıp seyreklik kullanarak, seyreklik sistemi bir kümeleme problemine çevrilebilir. Olağan durumda, aynı alt uzaydaki iki sınıfın ayırt edilememesinden dolayı, doğrusal olarak ayrılmayan durumların öğrenimi olası değildir. Bire toplam ve eksi olamama koşulları ile kalıp seyreklik birlikte kullanıldığında, k-flats, k-simplexes, k-polytopes olarak adlandırılacak çeşitli k-means üstproblemlerine ulaşılır. Polytope aynı boyut sayısına sahip simplekslerden oluşan bütün bir cisimi belirtir. K-polytopes deneysel olarak k-means toplulukları kadar iyi ve çekirdek k-means'ten daha iyi sonuçlar verir. Bütünsellik bırakıldığı ve boyutsal heterojenlik olduğu takdirde, k-polytopes bir tek sınıf öğrenim yöntemi olan simpleksel öğrenim ile genelleştirilebilir. Kombinasyonel doğası gereği, çözüm için evrimsel yöntem seçilmiştir. Bu çeşit bir uyarılma doğrusal ayrılmayan durumları kolayca öğrenebilmekte ve de güvenilir bir yöntem olarak görünmektedir. Boyutların birbirine dik olduğu varsayıldığı için hala eksiklikler vardır. Evrişim diklik sorununa pratik bir çözüm sağlar. Evrişimli durum kullanılarak, kaydırmaya değişimsiz k-means problemi sunulmuş ve evrişimli sözlük öğreniminin denetimsiz öznelik öğrenimi başarımları değerlendirilmiştir. Bu eklentiler ve değerlendirmeler sonucunda, seyrek ve bol gösterimler sistemi çok önemli bir makine öğrenimi yöntemi olarak karşımıza çıkmaktadır.

Anahtar kelimeler: Seyreklik, Polytope, Simpleksel, Evrişim, Makine öğrenimi  
**ACKNOWLEDGMENTS**

First of all, I would like to express my sincere gratitude to my supervisor, Dr. Mehmet Türkan for his continuous support of my Ph.D. study and related research. His guidance has opened doors in many interesting topics I could have not imagined by myself. His experience and also patience made this thesis possible.

Furhtermore, I would like to thank Prof. Türker İnce, Asst. Prof. Kaya Oğuz, Prof. Ender Mete Ekşioğlu, and Assoc. Prof. Behçet Uğur Töreyn for accepting to be members of my Ph.D. committee and for reviewing this work.

I would like to thank my former course advisor Prof. Cem Evrendilek for

bearing with me on countless issues. His eminent theoretical enthusiasm has also shaped this thesis. I also want to thank Assoc. Prof. Devrim Unay for his endless positive enthusiasm and his academic advices.

Many thanks to my former labmates Erdem Okur, Zaliike Keskin Erdoğan, Serkan Gürleyen, Çağrı Çavdaroglu, İbrahim Halil Özcan, Fırat Aslan, Serhat Uzun- bayır, Levent Tolga Eren, Berkehan Akçay, Hande Aka Uymaz, Teoman Unay, Mehmet Taze and Onur Çağırıcı for their academic and social support during my Ph.D. programme.

Last but not the least, I would like to thank my family for their neverending support throughout my life. I would like to give special thanks to my aunt Prof. Lütfiye Oktar and my father Prof. Nezih Oktar, for being great examples to me in the academic world.

TABLE OF CONTENTS

ABSTRACT ..... iii

OZET..... v

ACKNOWLEDGMENTS ..... vii

TABLE OF CONTENTS ..... viii

LIST OF TABLES ..... xi

LIST OF FIGURES ..... xii

CHAPTER 1. SUMMARY ..... 1

    1.1. *Introduction* ..... 1

        1.1.1. Sparse Representations ..... 2

        1.1.2. Structured Sparsity ..... 2

    1.2. *Contributions and Organization of Thesis* ..... 3

        1.2.1. Chapter 2: Clustering and sparsity..... 3

        1.2.2. Chapter 3: if-polytopes as a superproblem of /c-means.... 4

        1.2.3. Chapter 4: Evolutionary simplicial learning..... 6

        1.2.4. Chapter 5: The problem of orthogonality ..... 9

        1.2.5. Chapter 6: Conclusion and perspectives ..... 11

    1.3. *Conclusion* ..... 11



<b>CHAPTER 2. CLUSTERING AND SPARSITY .....</b>	<b>12</b>
<b>2.1. Introduction .....</b>	<b>12</b>
2.1.1. Challenges in Clustering .....	13
2.1.2. Remedies for Challenges .....	13
<b>2.2. Sparse Representations and Clustering.....</b>	<b>16</b>
2.2.1. Sparse representations: An overview .....	17
2.2.2. Sparse coding linked with clustering .....	18
2.2.2.1.....	
Sparsity as a feature transform .....	19
2.2.2.2.....	
Structured sparsity as a means of clustering.....	20
<b>2.3. Dictionary Learning and Clustering.....</b>	<b>22</b>
2.3.1. Dictionary learning: An overview .....	23
2.3.2. Dictionary learning <u>linked</u> with clustering.....	25
<b>2.4. Related Concepts .....</b>	<b>26</b>
2.4.1. Signal processing perspective .....	27
2.4.2. Machine learning perspective .....	28
<b>2.5. Shortcomings and a proposal.....</b>	<b>30</b>
<b>CHAPTER 3. K-POLYTOPES CONCEPT .....</b>	<b>32</b>
<b>3.1. Introduction .....</b>	<b>32</b>
3.1.1. Formulating block-sparsity.....	34
3.1.2. if-means within a sparse framework.....	34
<b>3.2. K-simplexes .....</b>	<b>35</b>
3.2.1. A solution to /c-simplexes.....	37
3.2.2. Complexity analysis.....	38
3.2.3. Drawbacks of A;-simplexes.....	38
<b>3.3. K-polytopes.....</b>	<b>39</b>
3.3.1. An algorithm for A;-polytopes .....	41
3.3.2. Complexity analysis.....	43
<b>3.4. Experimental Results.....</b>	<b>44</b>
<b>3.5. Discussion .....</b>	<b>46</b>

<b>CHAPTER 4. EVOLUTIONARY SIMPLICIAL LEARNING .....</b>	<b>48</b>
<b>4.1. Introduction .....</b>	<b>48</b>
<b>4.2. Simplicial Learning.....</b>	<b>51</b>
<b>4.2.1. Definitions.....</b>	<b>51</b>
<b>4.2.2. Related work .....</b>	<b>51</b>
<b>4.2.3. Mathematical formulation .....</b>	<b>52</b>
<b>4.3. Evolutionary Approach .....</b>	<b>53</b>
<b>4.3.1. The fitness function .....</b>	<b>54</b>
<b>4.3.2. Mutations and breeding .....</b>	<b>56</b>
<b>4.3.3. Implementation details.....</b>	<b>57</b>
<b>4.4. Experimental results .....</b>	<b>58</b>
<b>4.4.1. Outlier detection .....</b>	<b>59</b>
<b>4.4.2. Multi-class experiments .....</b>	<b>61</b>
<b>4.5. Computational Complexity .....</b>	<b>63</b>
<b>4.6. .... D</b>	
<b>Discussion.....</b>	<b>64</b>
<b>CHAPTER 5. THE PROBLEM OF ORTHOGONALITY .....</b>	<b>70</b>
<b>5.1. ....</b>	
<b>Introduction.....</b>	<b>70</b>
<b>5.2. ....</b>	
<b>Convolutional Sparse Representations.....</b>	<b>73</b>
<b>5.2.1. A solution to shift-invariant fc-means .....</b>	<b>74</b>
<b>5.2.2. Convolutional dictionary learning as a generalization</b>	<b>75</b>
<b>5.3. ....</b>	
<b>Experimental Results .....</b>	<b>76</b>
<b>5.3.1. Shift-invariant A;-means .....</b>	<b>76</b>
<b>5.3.2. Convolutional dictionary learning.....</b>	<b>78</b>
<b>5.4. ....</b>	
<b>Discussion.....</b>	<b>83</b>
<b>5.4.1. Variations on neural networks.....</b>	<b>83</b>
<b>5.4.2. Multilinear approach.....</b>	<b>87</b>

5.4.2.1. ....	
Tensor-based sparse representations .....	88
5.4.2.2. Complex, hypercomplex, and Geometric algebra based approaches.....	93
5.5. ....	
Conclusion.....	95
CHAPTER 6. CONCLUSION AND PERSPECTIVES .....	97
REFERENCES .....	100
APPENDICES	
APPENDIX A. RESIDUAL CODES .....	121
A.1. Introduction .....	121
A.2. Introducing Error Codes .....	121
A. 3. Experimental results .....	122
A.4. Conclusion .....	126
APPENDIX B. K-POLYTOPES ALGORITHM DETAILS .....	129
CURRICULUM VITAE.....	132

LIST OF TABLES

Table 1. Clustering frameworks and respective constraints ..... 35

Table 2. Clustering success rates over 100 randomly initialized runs. Top value in each cell designates the mean, middle value is the maximum accuracy attained and the bottom value is the average running time..... 41

Table 3. Distinctions between the terms for generic objects ..... 52

Table 4. Information regarding the datasets used in outlier detection experiments ..... 60

Table 5. Mean AUC ROC results, and running times ..... 66

Table 5. (continued) Mean AUC ROC results, and running times..... 67

Table 6. Classification accuracy and computation time for six synthetic datasets, and for the proposed binary MNIST8 problem ..... 68

Table 7. Classification error rates of various methods on handwritten digit datasets, USPS and MNIST. ESL appears as a superior generative method, nearly performing at the capacity of discriminative Gaussian SVM on both datasets..... 69

Table 8. Feature extraction methods in the benchmark..... 80

Table 9. Classification accuracy (%) of feature extraction methods with linear SVM applied on the whole MNIST and USPS datasets ..... 81

Table 10. Average approximation PSNR (dB) performance values of learnt dictionaries as in Figure 27 and Figure 28.  $k$  represents the sparsity used for testing ..... 125

Figure 1. Sparse and redundant representations implement a transformation  $Y = AX$  that increases the dimensionality of feature space from  $N$  to  $M$  and searches for suitable subspaces within the new  $M$ -dimensional feature space ..... 15

Figure 2. An example sparse coding case: (left) a portion of the dictionary is shown with colored columns (atoms), and (right) example sparse code-patterns colored with the same color as the atoms used during coding. There are two disjoint subspaces (green and blue code-patterns), and there

x

LIST OF FIGURES

also exists non-structural code-pattems (green-yellow and blue-yellow). ... 21

Figure 3. A block sparsity example illustrating the designation of disjoint sub-

spaces which are assigned as clusters: house (green) and face (blue) clus-

ters. The dictionary and the data represented in the sparse domain are

shown on the top-row ..... 23

Figure 4. The chosen dictionary may not always be appropriate for clustering

the data. An example sparse decomposition depicts that sparse codes are

not structured, but lie on intersecting subspaces ..... 26

Figure 5. On the left, a drawback of normalizing the atoms is presented. Two

classes cannot be discriminated. Superatoms on the right overcome this

problem ..... 31

Figure 6. A simple example of how additional constraints on sparse codes affect

the solution of sparse representations, (a) The conventional sparsity con-

straint together with (b) sum-to-one ( $t_1 + t_2 = 1$ ) and (c) sum-to-one and

non-negativity ( $t_1 + t_2 = 1$  and  $t_1, t_2 > 0$ ) constraints ..... 36

Figure 7. For the given two clusters: (left) a general solution to  $\ell_1/\ell_2$ -simplexes is

presented, i.e., there is no restriction on the sizes of simplexes, (right) a

more natural clustering occurs when simplex sizes are restricted. (Best

visualized in color.) ..... 38

Figure 8. Example convergence states for (top) Crescents and (bottom) MNIST.	
(a) Ground-truth clusters, and clustering results with (b) A-means, (c) A;—flats, (d) GMMs, (e) kernel /c-means, (f) ensemble A-means, (g) $k$ -polytopes. (Best visualized in color.)	45
Figure 9. A-polytopes aware of inter-cluster margins is capable of recovering exact number of clusters. Experimental results in (left) 2D and (right) 3D. (Best visualized in color.)	46
Figure 10. Conventional dictionary learning is incapable of distinguishing intensity/magnitude, or more technically two classes within the same subspace.	50
Figure 11. Examples of learned simplicials on synthetic datasets	61
Figure 12. (Left) There is orthogonal consideration. Every pairwise relation between dimensions is indistinguishable because of orthogonality. (Right) While considering spatial configuration of a 3-cell ID signal, the relation between cells $x$ and $z$ is obviously different from the other relations, i.e., $x$ and $z$ are not neighbors	71
Figure 13. Vectorized distance will dictate that 8 is closer to the main image. However, it is indeed more natural to say that two images of 9 are more similar to each other.	72
Figure 14. The local dictionary $A$ consists of convolutional atoms, whereas the global dictionary $D$ is filled with zeros outside the convolutional area	74
Figure 15. Clustering accuracy (%) of A-means (KM) based methods as a function of mean shift applied on MNIST. The proposed shift-invariant KM ( $KM_{si}$ ) is robust to shifts	77
Figure 16. Patch-based versus convolutional dictionaries learned on MNIST. For a clear visualization, atoms are of size $8 \times 8$	79
Figure 17. Classification accuracy (%) as a function of varying training sizes applied on (top) MNIST and (bottom) USPS	80

Figure 18. Classification accuracy (%) as a function of different patch/kernel sizes applied on (preprocessed) MIT-BIH using linear SVM classifiers	81
Figure 19. Classification accuracy (%) as a function of different patch/kernel sizes applied on the raw Electric Devices dataset using linear SVM classifiers.....	83
Figure 20. Classification accuracy (%) as a function of different patch/kernel sizes applied on MNIST using linear SVM classifiers .....	84
Figure 21. On the left graph there are edges within the hidden layer. On the right there is the classical fully connected neural network. Switching two nodes within the hidden layer makes no difference for classical neural network. ..	85
Figure 22. Concept of partial propagation. Partial propagation can be performed even when the hidden layer is composed of infinitely many nodes .....	86
Figure 23. A generalization of neural network layer cases. From left to right, discrete-discrete (classical), discrete-continuous, continuous-discrete, and continuous-continuous cases are depicted .....	87
Figure 24. A sparsely overlapping block-wise connected continuous layered network as an alternative to a 1D convolutional neural network in preserving the spatial information of a 1D signal.....	88
Figure 25. An encoding scheme to preserve spatio-temporal information for (top) 1D mono audio and (bottom) 2D grayscale image cases.....	94
Figure 26. Noise removal as an application of simplicial learning .....	98
Figure 27. Results of dictionary learning performed on the <i>Barbara</i> image. Dictionary size $64 \times 256$ , $k = 8$ for both learning and testing. Each column represents a different initial dictionary. First and middle columns are randomly initialized. Last column is of DCT initialization. Figures depict PSNR (dB) performance values versus iteration number. ....	122
Figure 28. Results for ten cases of uniformly random initial dictionaries. Lower and upper lines of each method correspond to minimum and maximum	

PNSR (dB) values attained among all ten cases. Middle lines represent average values .....	124
Figure 29. Performance gain factor for different sparsity levels, in the case of a relatively small and a large dictionary.....	126
Figure 30. The convergence performance for dictionary size 64 x 256 and $k$ as 8 for both learning and testing.....	127
Figure 31. A frequency subdomain of learnt dictionaries formed with same pro- cessing time. MOD on the left, EcMOD+ on the right.....	128
Figure 32. The block-diagram of the proposed algorithm to solve /c-polytopes problem .....	131
<b>CHAPTER 1: SUMMARY</b>	

### 1.1 Introduction

Machine learning is a currently very popular subdomain of artificial intelligence. Machine learning algorithms learn models based on observed data. An algorithm is a recipe for a machine to be executed step by step. As opposed to conventional algorithms, machine learning algorithms fit models to the data observed, thus provide an abstraction layer between the data and the machine, much like the human learning process. In short, machine learning algorithms aim to learn models based on observed data, later to be used for predictions without being hardcoded to perform the task (Bishop, 2006).

Two main types of such algorithms are fisted as *supervised* and *unsupervised*. In supervised setting, algorithm builds a model of data that contain both the inputs and the desired outputs, referred to as the training data. Then, according to the model it can then predict the output of a new input. Two versions of supervised algorithms include *classification* and *regression*. In classification, outputs or labels are restricted to be a set of values, whereas in regression outputs represent a range of values. In unsupervised learning, outputs are absent in the data, in other words there are no labels. Therefore, algorithms tend to analyze data only based on the input values. For example, in *cluster*



*analysis*, observations with similar input values are grouped together to form clusters, based on some similarity metric chosen.

Another type of machine learning algorithms is listed as *feature learning*, aiming at finding better representations of data (Bengio et al., 2013). Classic example of this domain includes Principal Component Analysis (PCA) (Pearson, 1901). These algorithms try to preserve the information in the data but transform it into a more useful form, as a preprocessing step before classification or clustering. Feature learning can itself be supervised or unsupervised, depending on whether it requires labels or not.

The framework that is adopted in this thesis, namely *sparse and redundant representations* equipped with *dictionary learning* is conventionally listed as a feature learning method. In fact, sparse and redundant representations are most commonly utilized in reconstructive signal processing tasks such as compression (Akbari et al., 2016), denoising (Nejati et al., 2016; Zhuang and Bioucas-Dias, 2018), and inpainting (Zhuang and Bioucas-Dias, 2018). However, they can also be used as preprocessing tools for further classification and clustering schemes for machine learning (Elhamifar and Vidal, 2013). In this thesis, examples of adapting sparse framework as direct applications of clustering and classification problems are presented through variations on structured sparsity (Huang et al., 2011), a notion to be introduced in Sect. 1.1.2. First of all, a more formal introduction to sparse representations is given.

### 1.1.1 Sparse Representations

Sparse representations are widely used in reconstruction tasks. They model the data through a few linear combinations attained from an overcomplete set of elements or basis, often referred to as *the dictionary*. This dictionary can either be fixed analytically or be adapted to the data at hand through learning. Conventional optimization of dictionary learning for sparse representations is given in Eqn. (1),

$$\underset{\{x_i\}_{i=1}^N}{\operatorname{argmin}} \|y_i - Ax_i\| \text{ subject to } \|x_i\|_0 \leq q, \forall i, \quad (1)$$

where  $A$  is the overcomplete dictionary and  $x_i$  denotes the sparse representation of  $y_i$ ,  $\forall i$ . While minimizing the reconstruction error of  $y^*$  over  $A$ , each sparse vector  $x^*$  may

have maximally  $q$  number of nonzero components due to  $\ell_0$ -norm constraint. In literature, approximate iterative solutions by *sparse coding* and *dictionary update* have been proposed to this nonconvex problem and its variants (Elad et al., 2010; Gribonval et al., 2015).

### 1.1.2 Structured Sparsity

In this original version of the problem, any of the atoms in the dictionary can be picked. In other words, there is no constraint on which parts of the sparse codes to be filled. Instead of picking atoms individually, if groups of them are to be picked collectively then this corresponds to structured sparsity and direct connections to clustering problem can be made when groups appear in blocks (Eldar and Mishali, 2009; Eldar and Bolcskei, 2009) and this is the topic of Chapter 2. Therefore, as of now structured sparsity can be seen as an additional constraint on sparse representations that make them appear closer to problems in the machine learning domain.

There are three main objectives of this thesis. First objective is to formulate a general sparsity based clustering framework that can effectively learn linearly non-separable cases as in Chapter 3. Such a framework is based on structured sparsity and additional constraints on sparse codes. In Chapter 4, this framework is further generalized to provide a direct solution to the classification problem by the introduction of one-class learning. The last objective is to show the importance of spatial information in machine learning of signals through convolutional sparse representations, and provide insight for future studies in this light as in Chapter 5.

## 1.2 Contributions and Organization of Thesis

### 1.2.1 Chapter 2: Clustering and sparsity

Chapter 2 introduces the problem of clustering in a formal manner. Famous challenges in clustering are listed. The notorious phenomenon, namely *the curse of dimensionality* is introduced as a challenge in high dimensional real-world machine

learning applications. *The problem of overfitting* is another well-known challenge, in which some kind of memorization is observable instead of learning. Noise and outliers issue is another notable challenge in clustering. Then, feature extraction, feature selection, and combinatorial feature selection methods are mentioned to alleviate these problems. All these methods are performed within original or reduced number of dimensions. Then, concept of sparse representations is introduced as a way of feature transformation with the benefit of expanded dimensions. Sparse representations are resilient to noise and outliers when certain conditions are met, and also by manipulating level of sparsity, problem of overfitting can be addressed.

A detailed mathematical overview of sparse representations is then given and sparsity concept is then tied with clustering. There are two ways to relate sparse representations with the clustering problem. First, sparse representations can be seen as a feature transformation method as a preprocessing step for clustering. More interestingly, structured sparsity in the form of block sparsity, corresponds to a subspace clustering scheme directly. However, the dual of sparsity concept namely the dictionary may not be proper to designate subspaces as clusters. Therefore, adaptive dictionaries are needed to shape the state-model entity into well-formed clusters. In this light, dictionary learning problem is introduced and formulated in a formal manner. Most notably, MOD (Engan et al., 1999) and K-SVD (Aharon et al., 2006) are two well known approximate iterative solutions to the problem. After establishing the necessary background information, examples from literature are given that use sparse representations framework as a direct tool for clustering to solve segmentation (Ramirez et al., 2010), and denoising (Dong et al., 2011) problems.

Later in the chapter, related concepts from signal processing and machine learning perspectives are given. Compressive sensing and multiple measurement vectors approach are mentioned under the hood of signal processing perspective. On the other hand, support vector machines, decision trees, and neural networks are reviewed from the machine learning perspective. The chapter concludes with a discussion of shortcomings of conventional sparse representations and proposals to overcome those issues. One issue is related to random initializations and addressed in Appendix A. The other issue is more fundamental and is about learning linearly non-

separable cases to be addressed both in Chapter 3 and 4.

### 1.2.2 Chapter 3: $K$ -polytopes as a superproblem of $k$ -means

Chapter 3 builds on top of Chapter 2 and presents the well-known  $\ell_1$ -means problem in the context of sparse representations. A motivation to generalize  $k$ -means arises when one considers its drawbacks even assuming an optimal solution. Namely,  $\ell_1$ -means cannot learn linearly nonseparable cases (i.e. clusters with nonconvex shapes) and the number of clusters is hardcoded. Kernelization and ensembling are two approaches that solve nonlinearity issue, but they do not utterly generalize  $\ell_1$ -means problem as a superproblem. Through special constraints applied on sparse representations, one can arrive at certain superproblems of  $\ell_1$ -means.  $K$ -flats (Canas et al., 2012), for example, is a superproblem of  $\ell_1$ -means, and one of the contributions of the chapter is to show that it can be implemented within a sparse framework. In this chapter,  $\ell_1$ -simplexes and  $\ell_1$ -polytopes problems are additionally introduced as superproblems of  $\ell_1$ -means in which piecewise-linear models can be learned.

In this light, first of all block-sparsity concept is mathematically defined, and the concept of *subdictionary* arises. Each assignment block, corresponds to a subsdictionary that can be thought of as a central prototype that claims an entity (or not) depending on the reconstruction error, much like a centroid claiming the closest point to itself. Therefore, sparse coding with block sparsity corresponds to assigning entities to respective central prototype, whereas dictionary update corresponds to updating the central prototypes as in an iterative solution. Proposed formulations originate from the fact that  $\ell_1$ -means problem can be represented within a sparse framework. In fact, in  $\ell_1$ -means there are  $k$  many blocks with sizes 1 and sparse codes are restricted to be 1 also. In that case, dictionary atoms directly designate centroids, and no further atom normalization should be carried out. Generalizing to bigger blocks when there are no constraints on the magnitudes of sparse codes corresponds to  $k$   $g$ -subspaces, when the block size is  $q$ . A sum-to-one constraint on sparse codes forces subspaces into  $(q - 1)$ -dimensional flats. A further non-negativity constraint forces  $(q - 1)$ -dimensional flats into  $(q - 1)$ -dimensional simplexes. In this regard, one can arrive at  $\ell_1$ -simplexes superproblem. A solution to  $\ell_1$ -simplexes problem is given, in which data points are first

assigned to closest simplex. In sparse coding perspective, this corresponds to finding the positive barycentric coordinates of the projection point. After acquiring all sparse codes, a dictionary update then updates the simplexes. A complexity of  $\mathcal{O}(m^3)$  is determined for this type of solution, where  $m$  is the number of data points.

However, there are two drawbacks of /c-simplexes formulation. First of all, there is no constraint on the size of simplexes and a single simplex is still of convex shape. Therefore, /c-simplexes is further generalized into Zc-polytopes formulation where simplex edge size is now limited and a prototype can be now piecewise-linear. A definition of the word *polytope* is given as an intact object composed of many simplexes. Intactness is important here, as it defines a single object. In /c-simplexes there is only a single projection object. However, in fc-polytopes there are many candidate simplexes to test for within a polytope. Therefore, an additional hypergraph data structure is kept to define the shape of the polytope, where each valid simplex within the polytope can only be  $g$ -dimensional, corresponding to a hyperedge relating  $q$  many vertices within the hypergraph. Note that, for a simplex to be valid it should at least receive some amount of projections.

Chapter 3 includes a solution to /c-polytopes problem, consisting of subdivision, pruning, and merging subroutines. Subdivisions are needed when the edge size of a simplex is above the limit. Similarly, pruning is needed when the simplex does not require any or enough projections. Topologically important merging process is needed to fix excessive pruning and subdivisions. A detailed complexity analysis of /c-polytopes gives out a complexity of  $\mathcal{O}(m^3)$  again.

Most importantly, performance of /c-polytopes is compared against  $k$ -means, A;—flats, Gaussian mixture models (Reynolds, 2015), kernel A-means (Dhillon et al., 2004), and ensemble A-means (Iam-on and Garrett, 2010). It appears that ensemble /c-means is the most general method in literature and proposed method matches the performance of ensemble A-means even surpassing it at certain cases. It is also experimentally validated that, if the minimum intercluster margin is known, A-polytopes is also able to recover the number of clusters accurately in another set of experiments. The chapter concludes with a discussion considering the case in which clusters are not homogeneously  $\wedge$ -dimensional but heterogeneous and also a discussion on supervised

setting leading to the formulation in Chapter 4.

### *1.2.3 Chapter 4: Evolutionary simplicial learning*

Chapter 4 generalizes A-polytopes formulation, which is only about clustering, into a general framework that can be applied to many other tasks, such as one-class learning, or multi-class classification. In its final form, called evolutionary simplicial learning, the proposed framework learns more general piecewise linear constructs, namely an arbitrary union of simplices to the data at hand through evolution.

Chapter 4 starts with an introduction to one-class learning. It is possible to categorize one-class learning methods by the type of the model targeted. In one approach, enclosing hypersurfaces (i.e. decision boundaries) are targeted. As an opposing approach, graph based methods seek skeletons within. Most importantly, dictionary learning can also be regarded as an inner-skeleton method. In fact, in sparse representations based classifier (SRC) models, data is classified accordingly favoring the most reconstructive or representative dictionary (Wei et al., 2013). This form of SRC is defined as generative-only. While there are parallels between inner-skeleton and generative methods, there is a relationship between decision-boundary and discriminative approaches. A method can both be generative and discriminative at the same time and discriminative dictionary learning methods are examples of such approach (Jiang et al., 2013; Song et al., 2019).

A crucial point is that SRC methods are not capable of learning linearly non-separable cases, as a linear generative dictionary learning method is incapable of distinguishing two classes within the same subspace. In other words, SRC is insensitive to intensity/magnitude of a pattern. In that case, simplicial learning is introduced both as a generative-only method and as a generalization of /c-polytopes concept of Chapter 3 that can learn any linearly non-separable case in theory. In this light, definitions of a polytope, a simplicial complex, and a simplicial are given as generalizations of piecewise linear constructs. Polytope refers to an intact object composed of homogeneously dimensional simplexes. A simplicial complex is a formal set of simplices satisfying following two conditions: (i) every face of a simplex from this set is also in this set and (ii) the non-empty intersection of any two simplices is a face of these two simplices. A

simplicial is instead defined as an arbitrary union of simplices, being the most flexible structure.

There are various recent studies involving these structures (Luo et al., 2017; Belton et al., 2018). Not surprisingly, Chapter 4 uses a sparse representations framework to formulate simplicial learning in an efficient manner. Three ingredients in simplicial learning, similar to  $\ell_1$ -polytopes formulation are, sum-to-one constraint, non-negativity, and group sparsity this time instead of block-sparsity. A single hypergraph data structure is kept, to keep track of groups, or valid simplexes within the simplicial, each hyperedge relating arbitrary number of vertices this time.

However, there is no direct restriction on the number of simplices or the dimensions of those simplices. Additional penalty terms are needed to keep number and dimensionality of simplices small for a compact solution. An evolutionary process seems feasible for this problem that looks combinatorial in nature. In this regard, a fitness function is needed to guide the search process. An optimization process only for the number and the dimensionality of simplices is not enough. Volume, or more formally content of the simplices must be considered also. Luckily, content of an arbitrary simplex can be calculated using Cayley-Menger determinant (Michelucci and Foufou, 2003). Because of allowed heterogeneous dimensionality, an important issue arises, namely how to effectively compare the content of a triangle and a line-segment as an example. In this light, an *approximated cumulative discrete content* formula is devised, introducing exponential penalty to content through dimensionality. However, a direct addition of this term to the sum of squared errors is not practical. Therefore, devising a formula involving logarithmic scale is chosen as the final option.

Having pinned down the fitness function, mutations and breeding processes then perform the actual search. Mutations are performed on the hypergraph data structure kept as an incidence matrix of zeros and ones. Breeding process extracts two subsimplicials from two simplicials each and merges them together. While breeding determines the core dimensionality, mutations instead fine tune the simplicial to the data at hand. An important implementation detail is that, while starting as a single point is enough for low dimensional datasets, in high dimensional cases a procedure involving fc-means is used as a subroutine to designate an initial simplicial.

Proposed method is tested in two phases of experiments. At first, the performance is evaluated in a one-class classification task for outlier detection. In 17 datasets considered, 12 competing methods are reported involving ESL. ESL not only presents best average Area Under the Curve (AUC) Receiver Operating Characteristics (ROC) performance it has the least standard deviation, and it seems as the most reliable method among considered ones for this performance measure. In second phase, multi-class experiments are performed. In challenging synthetic data sets created, involving linearly non-separable cases requiring intensity/magnitude distinction, ESL easily outperforms all other dictionary learning methods considered, such as Sparse Representation based Classification (SRC) (Wright et al., 2008), Label Consistent

K-SVD (LCKSVD1 and LCKSVD2) (Jiang et al., 2013), Dictionary Learning with Structured Incoherence (DLSI) (Ramirez et al., 2010), Fisher Discrimination Dictionary Learning (FDDL) (Yang et al., 2011), Dictionary Learning for Commonality and Particularity (DLCOPAR) (Kong and Wang, 2012) and Low-rank Shared Dictionary Learning (LRSDL) (Vu and Monga, 2016, 2017). In real-world tasks of digit classification (USPS (Hull, 1994) and MNIST (LeCun et al., 2010)), ESL performs as a superior generative method nearly performing at the capacity of Gaussian SVM. Chapter 4 concludes with a discussion of possible probabilistic and discriminative modifications.

#### *1.2.4 Chapter 5: The problem of orthogonality*

Even with all these enhancements, an important shortcoming remains in the case of inputs that contain 'spatial' information. In conventional consideration, each dimension of data is assumed to be independent from another, as a result of vectorization process. More technically, the bases are assumed to be orthogonal (as in  $n$ -dimensional Euclidean space). However, neighboring or close data cells in digital signal forms such as sounds, images, or videos most probably have certain dependency, which is referred to as 'spatial' information throughout Chapter 5. Convolutional neural networks have advantage in such cases as they preserve and process the 'spatial' information. Chapter 5 offers an overview of this orthogonality problem from the perspective of sparse representations through convolutional case, and provides further



insight for future studies on this topic.

Chapter 5 opens up with a discussion on how many possible spatial configurations a signal could have been in going back from a  $n$ -sized vector format. The result is  $dk(n)n!$  where  $dk(n)$  is the  $k$ -th Plitz function, which gives the number of ordered factorizations of  $n$  as a product of  $k$  terms, designating a  $/c$ -dimensional signal.

The chapter goes on to explain that orthogonal consideration is problematic for classical problems such as  $/c$ -means also. Considering  $/c$ -means to be applied on images, an orthogonal vectorized distance calculation between two images might not be natural. Similarly, in Computer Graphics domain, for natural interpolation between two rotation matrices, quaternions are used for spherical linear interpolation instead of naive linear interpolation (Jafari and Molaei, 2014).

These ideas lead to shift invariant formulation of  $/c$ -means as a more natural clustering problem of images. Convolutional sparse representations are utilized to formulate the shift invariant  $/c$ -means problem, and a solution to this problem is then given, through OMP and MOD as subroutines. Convolutional dictionary learning is listed as a generalization of this formulation, as a general unsupervised feature extraction method. Performance of convolutional dictionary learning as an unsupervised feature extraction method is validated many times in literature (Zeiler et al., 2010; Garcia-Cardona and Wohlberg, 2018). However, the main aim of this chapter is to provide an extensive comparison with classical orthogonal consideration. An important note is that, a shift from  $l_0$  to  $h$  constraint is applied in sparse representations to take care of computational complexity and information loss problems.

Two sets of experiments are performed using the SPORCO library (Wohlberg, 2017) for the needs of convolutional dictionary learning. A modified version of MNIST is created in which each image uniformly randomly received a shift in each axes from a mean of 2, 4, 8, and 16 pixels respectively. It is observed that, shift invariant  $/c$ -means formulation is very resilient to shifts, while  $/c$ -means, kernel  $/c$ -means, and ensemble  $/c$ -means performances drop to random guess performance in case of big shifts.

In another set of experiments, unsupervised feature extraction performance of CDL is measured. Competing methods are Histogram of Oriented Gradients (HOG) (Dalai and Triggs, 2005), Local Binary Patterns (LBP) (Ojala et al., 1996), and Gabor

Feature Extraction (GFE) (Haghighat et al., 2015), while dictionary learning has 3 versions to be tested. DL is global only method where atom size is that of image size. In patch-based dictionary learning (PDL), smaller patches are extracted in a sliding window manner in a local only manner. Convolutional dictionary learning (CDL) method can be regarded as a both global and local method. Experimental results suggest that CDL is superior to both DL and PDL when the data set contains enough samples. GFE, as an unsupervised simulation of first layers of a CNN perform the best, CDL is the second best, very close to the performance of HOG. A level below, DL, PDL, and LBP perform similarly.

The chapter goes on with certain possible variations on neural networks. A sparsely overlapping block-wise connected continuous layered network is proposed as an alternative to a 1D CNN. Geometric algebra (Wang et al., 2019) and multilinear (Lu et al., 2011) approaches to machine learning are mentioned, noting that these methods should also work for 1D signals to start with.

#### *1.2.5 Chapter 6: Conclusion and perspectives*

In Chapter 6, conclusions are drawn based on the main ideas and contributions of the work, and various future perspectives are presented for the domain of sparse representations to be applied on machine learning.

### *1.3 Conclusion*

Dictionary learning for sparse and redundant representations conventionally appears as a feature learning method in the domain of machine learning. However, using structural constraints, namely using block-sparsity, it can be cast as a clustering problem directly. In conventional form, learning of linearly nonseparable cases is not possible. Through additional magnitude constraints on sparse codes, one can arrive at superproblem of  $\ell_1/\ell_2$ -means, such as  $\ell_1/\ell_2$ -simplexes, or  $\ell_1/\ell_2$ -polytopes that can learn linearly non-separable cases. Simplicial learning is a further modification of  $\ell_1/\ell_2$ -polytopes concept for the problem of classification through the introduction of one class learning. Due to combinatorial nature of simplicial learning, an evolutionary approach is taken. Evolutionary simplicial learning can easily handle linearly non-separable cases

and surpass the capacity of classical dictionary learning methods considered.

Due to considering dimensions as orthogonal to each other via vectorization process, there are still problems for machine learning of signals. Convolution operator is a practical tool that can preserve spatial information in this regard. Through convolutional sparse representations a shift-invariant /c-means problem is formulated and solved. Furthermore, unsupervised feature learning capacity of convolutional dictionary learning is evaluated. With all these modifications and considerations, sparse and redundant representations framework appear to be an indispensable tool in advancing machine learning research.

## CHAPTER 2: CLUSTERING AND SPARSITY

### 2.1 Introduction

As a subdomain of machine learning, clustering is an approach to unsupervised learning. There is no labeling required, unlike classification tasks. In broad terms, clustering can be expressed as *exploring the unknown*. The wide range of clustering applications includes search engines, social networks, visual tasks such as image segmentation, and DNA analysis. Search engines need to cluster information in order to be able to retrieve relevant data in the times of querying. Social networks inherently appear in a clustered nature. Image segmentation is a visual application of clustering. Not surprisingly, molecular biology is a promising domain for clustering applications, due to its aim of discovering the unknown world (Kiselev et al., 2019).

Clustering can simply be defined as the task of grouping entities in terms of a similarity measure. Here, the critical issue is to understand what is meant by “similar”. Similarity is in a sense the inverse of a distance metric between two entities. The shorter the distance, the more similar the entities, and vice versa. It is important hence to note that, clustering results will be crucially dependent on the similarity notion chosen. A conventional distance metric is the squared Euclidean distance between two data points  $x$  and  $y$  which is defined as  $dist(x, y) = \|x - y\|^2$ . Many other similarity measures, e.g., (Borgefors, 1986; Maesschalck et al., 2000), could be utilized to tackle the broad range

of domain specific clustering problems.

Clustering methods are usually categorized under four main groups. The first group is based on the cluster formation methodology including top-down, bottom-up, and analytic optimization techniques (Gordon, 1987). A second group lists methods depending on the cluster model acquired such as hierarchical (Sibson, 1973), centroid (as in k-means (Lloyd, 1982)), distribution such as expectation maximization (Carson et al., 2002), density (Ester et al., 1996; Kriegel et al., 2011), subspace, group, and graph-based models (Felzenszwalb and Huttenlocher, 2004; Novak et al., 2010). Thirdly, depending on the relationship type between entities and clusters, hard or soft clustering can be distinguished by defining binary or fuzzy relations, respectively. A final clustering group, based on the nature of cluster-cluster relations, defines the distinction between overlapping versus disjoint partition groups in general.

### 2.1.1 Challenges in Clustering

Clustering problem is not a trivial task, especially in the case of high dimensional data, found in most real-world applications. Conventional clustering methods usually fail in such scenarios. This phenomenon is referred to as *the curse of dimensionality* (Parsons et al., 2004). The problem here can be described with a synthetic example where there is a set of data originally in a low-dimensional space, which is gradually expanded with irrelevant information within some additional dimensions. As such dimensions are incrementally added, the inherent distribution of original data will gradually become obscure because of the increased volume, and that statistically sound subset becomes sparser in higher dimensions. This is especially problematic in the case of clustering, which employs some conventional distance metrics, as with each additional dimension, such functions will lose their discriminative power.

In addition, large amounts of data does not mean that learning algorithms will be successful. *The problem of overfitting* (Hawkins, 2004) usually occurs when the model being captured is excessively complex because of very high dimensional feature space. Also, the data at hand may not be very representative of the whole ground-truth model. In that case, learning algorithms tend to fit a model to data samples at hand, thus missing the true underlying structure. In other words, some kind of memorization

occurs instead of learning.

Noise and outliers are additional challenges to be tackled in practical signal processing and learning tasks. Especially, non-Gaussian noise is common in applications that involve measurements. Outliers, on the other hand, are inconsistent observations among the general population. Combined with certain output constraints, these peculiarities pose great challenges for problems involving both linear and non-linear systems. The effects of these additional considerations are best investigated in some recent studies as (Stojanovic et al., 2016; Stojanovic and Nedic, 2016).

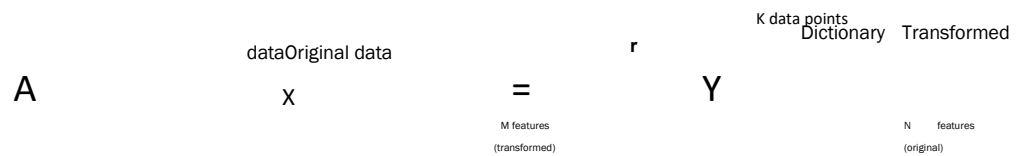
### 2.1.2 Remedies for Challenges

Because raw data is usually in a crude form, as explained, clustering approaches require a preprocessing step to cope with high dimensions and undesired sampling issues. Various preprocessing techniques have been proposed to increase performance in cases of high dimensions, e.g., (Hinton and Salakhutdinov, 2006; Yan et al., 2007; Baudat and Anouar, 2000; Jolliffe, 2002). They generally reshape the sample space through transformations or eliminations to observe the dataset in a refined way that would be more suitable for further processing. In an example, Principal Component Analysis (PCA) (Jolliffe, 2002) is a method that transforms sample attributes into a form that would have the highest variance, thus more suitable for discrimination tasks with an additional benefit of reduced dimensions. In general, this concept can directly be generalized as *feature transformation*. However, it is important to note that such techniques do not discard irrelevant features, i.e., all features are preserved and reshaped through (non)linear combinations. As an extended approach, dimensionality reduction via feature selection performs elimination of irrelevant features and considers what seems to be the most important subset of features abiding by certain optimality criteria. Both of these techniques help future classification or clustering tasks to achieve more accurate representations in return.

Although feature selection can simply be used as a solution to high dimensional problems, elimination process however might lead to some loss of important information that have strong meaning in different context, i.e., in different subspaces. In this light, *subspace search* (Parsons et al., 2004), a combinatorial approach to subset selection, can be performed as an extension of feature selection where many subsets of

features are distinctively analyzed while keeping original dimensions intact. A class of clustering methods has been proposed which includes searching for clusters in subspaces rather than the original space, thus referred to as *subspace clustering* (Parsons et al., 2004). Considering data points in isolated but relevant dimensions eliminates the interference of irrelevant dimensions, hence provides a solution to the clustering problem. Observing data in many alternate subspaces can provide means to clustering certain groups in certain subspaces, and the rest in different ones. The whole data can effectively be partitioned through merging the solutions in these subspaces, in a way





**Figure 1. Sparse and redundant representations implement a transformation  $Y = AX$  that increases the dimensionality of feature space from  $N$  to  $M$  and searches for suitable subspaces within the new  $M$ -dimensional feature space.**

that would not be possible by observing all dimensions at once. In general, subspace clustering problem can be formulated by defining the number of subspaces, subspace dimensions and a corresponding basis when supplied with a set of data points that lie within a union of subspaces.

Note that it is also possible to take an extended approach to subspace search within expanded dimensions through *sparse and redundant representations* (Elad, 2010), which implements a transformation that increases the dimensionality of feature space, as illustrated in Fig. 1, and then searches for suitable subspaces within this new feature space. This chapter focuses on the sparsity-based clustering methods, considering sparse and redundant representations as a both feature transformation and a “structure” of clustering with the help of adaptive (learned) overcomplete dictionaries.

Having considered the challenge of high dimensional data, it is also important to mention certain techniques to overcome the problem of overfitting. In the case of clustering, a common way to deal with overfitting is to minimize within cluster variance (Demiriz et al., 1999). More generally, overfitting occurs when the model accommodates more parameters than needed (Everitt and Skrondal, 2002). In the sparse and redundant representations framework, the sparsity measure directly corresponds



*2.2 Sparse Representations and Clustering*  
to parameter quantity, and can be manipulated easily. For example, by keeping the sparsity constraint strict enough, a model based on a few parameters can be formed, exhibiting less overfitting.

In the case of noise and outliers, it has been shown that under certain conditions, it is highly probable that sparse and redundant representations admit a local minimum around the reference signal-generating model (Gribonval et al., 2015). This means sparse representations are indeed resilient to noise and outliers when certain criteria are met, such as appropriate scaling of dimensions, number of measurements, and model parameters. In practical applications, the nature of noise may not be Gaussian, but exhibit high levels of outliers. There are successful studies which overcome such situations with flexible structures for noise handling, e.g., a method based on a hybrid norm for minimizing the data fitting error term (Barazandeh et al., 2017), a nonparametric scheme minimizing some norms of residual and original signals (Mayiami and Seyfe, 2012). Furthermore, sparse and redundant representations are widely used in signal denoising applications, which provides the potential for robust models, even in the presence of (non-)Gaussian noise and outliers (Elad and Aharon, 2006; Shao et al., 2014; Zhu and Vogel-Heuser, 2014).

The rest of this chapter is organized as follows. Section 2.2 first overviews the problem of sparse representations, and then relates the clustering problem to the sparsity constraint, namely, to *sparse coding*. Following this, Sec. 2.3 introduces the principles of *dictionary learning* for sparse representations, and then connects the clustering problem to learned dictionaries. Section 2.4 then discusses related concepts to sparsity-based clustering. Finally, Sec. 2.5 mentions shortcomings of the standard formulation proposed in this chapter for general machine learning, and paves way to upcoming chapters.

### *2.2.1 Sparse representations: An overview*

Sparse representations have become a key research topic with various applications in signal and data processing, e.g., denoising (Elad and Aharon, 2006; Protter and Elad, 2009), modeling (Peyre, 2009), restoration (Mairal et al., 2008b,c), compression (Bryt and Elad, 2008; Peotta et al., 2006), and even more (Fadili et al., 2007; Mairal et al., 2008a; Liao and Sapiro, 2008). Put simply, sparse representations

represent most or all information contained in a data with a weighted linear combination of a small number of elements or *atoms* chosen from an overcomplete or redundant basis or *dictionary*. Such a dictionary is a set of atoms whose number is much larger than the dimension of the data space. Any entity then admits an infinite number of representations, and the sparsest such solution has interesting aspects for various data processing tasks.

The main objective is to obtain a sparse approximation of a given input data  $y \in R^N$ . Given a full rank matrix  $A \in M^{N \times M}$ , one tries to optimize the solution of

$$y = Ax \text{ subject to } \min \|x\|_0 \quad (2)$$

where  $x \in R^M$  denotes the sparse representation of  $y$  and  $\|x\|_0$  is the  $\ell_0$ -norm of  $x$ , i.e., the number of non-zero components in  $x$ . The matrix  $A$  is the *dictionary* and its columns (*atoms*) are assumed to be normalized in any norm.

In general, sparse representations for any set of data can be imposed in the form of a matrix factorization as  $Y = AX$ , where  $Y$  denoting the original data with  $N$  features and  $K$  samples, and  $X$  as the sparse representation matrix of  $Y$  in the new  $M$ -dimensional feature space as depicted in Fig. 1. Assuming that  $A$  is fixed, the  $\ell_0$ -norm constraint on the columns of  $X$  forces each data sample to use only a small number of feature templates (atoms). Hence, sparse codes, namely the columns of  $X$ , together with the atoms they use, define a subspace.

In practice, the whole problem can be relaxed as an approximate convex optimization while fixing  $A$  and solving for  $x^*$  for each  $y_j$   $M$  independently, by minimizing the total approximation error over all samples by

$$\arg \min \sum_{j=1}^K \|y_j - Ax_j\|_2 \text{ subject to } \|x_j\|_0 \leq k \quad \forall i, \quad (3)$$

which is known as *the sparse coding problem*. Here, the parameter  $k$  defines the maximum sparsity allowed for the representation of  $y_i$  during the sparse coding process.

## 2.2 Sparse Representations and Clustering

There is no known technique for obtaining the exact solution under general conditions on the fixed dictionary  $A$ , except for the exhaustive combinatorial approach. Searching for this sparsest representation is hence unfeasible. This problem is computationally intractable (Davis et al., 1997) and thought to be NP-hard (Tillmann, 2015). A wide variety of pursuit algorithms (Chen et al., 1998; Mallat and Zhang, 1993; Pad et al., 1993; Blumensath and Davies, 2008) have been introduced as heuristic greedy methods aiming at approximate solutions with tractable complexity.

For the  $\ell_0$ -norm constraint, greedy approaches are the most appropriate as the above problem in this form is NP-hard. Matching Pursuit (MP) (Mallat and Zhang, 1993) and Orthogonal MP (OMP) (Pati et al., 1993) are most widely used examples to these iterative methods. On the other hand, it has been shown that for many high dimensional cases,  $\ell_1$ -norm constraint (instead of  $\ell_0$ -norm) is sufficient to ensure the sparsest solution (Donoho, 2006). Note that the very same problem with  $\ell_1$ -norm constraint can then be solved via regular linear programming tools, such as interior point (Nesterov and Nemirovskii, 1994) or regression shrinkage (Tibshirani, 1996). Basis Pursuit (BP) (Chen et al., 1998) is the generalized term for  $\ell_1$ -norm constrained version, as an approximation to the original problem.

Note that, since the transformed feature space in  $X$  has higher dimensions than that in  $Y$ , this feature transform can also be coined as “dimensionality expansion”, as opposed to dimensionality reduction such as in PCA. This is an advantage because, through dimensionality expansion, it is possible to utilize the subspace clustering approach at large.

Sparse coding can be thought of as a method of information localization. In this sense, sparse representations and the clustering problems are usually complementary, as clustering itself includes a form of information localization. However, note that the sparsity constraint alone does not imply clustering. For example, random sparsity is an expression of information being localized to a certain

extent, but clustering would not be evident at all for such a case. Thus, sparsity property needs indeed to be structured in order to be significant in informative sense. By nature, in most real world examples sparsity property and clusters are usually observable together. As an example, in social networks, not everyone is friends with everyone else, and people appear to be in certain friendship groups. Similarly protein-protein interactions in molecular biology are selective, while proteins of same functional domain and cellular location tend to cluster (Schwikowski et al., 2000). A striking sparsity example can be given for the brain. The brain, that fundamentally based on compartmentalization, not only has spatial but also temporal sparsity. Neurons are active in relatively small number of time periods, and also, the activated population of neurons are spatially sparse, i.e., only a small portion of neurons are active at any time (Barnes et al., 1990).

At this stage, there are two possible directions towards a solution for the main topic, namely for the clustering problem. Firstly, it is possible to supply sparse representations (i.e., sparse codes) acquired to any existing data clustering method -as extracted features- to be further processed as exemplified in (Elhamifar and Vidal, 2013) . In this simple case, sparse representation framework remains as a tool of feature transformation and/or selection, as a preprocessing step for clustering. Secondly, it is possible to formulate the sparsity concept as a clustering problem directly through additional structural constraints on sparse and redundant representations.

#### *2.2.2.1 Sparsity as a feature transform*

The first option is to use the transformed feature space, namely sparse codes, as an input to any existing data clustering algorithm. As a special case, if the dictionary is chosen as the data itself, i.e.,  $A = Y$ , the result is a formulation as  $YX = Y$ . In this form,  $X$  contains information about a kind of *self-similarities* among the original

data. However, the diagonal entries of  $X$  has to be forced to be zero to prevent the trivial solution of  $YI = Y$ , where  $I$  represents the identity matrix with suitable dimensions. The columns of the dictionary are usually normalized, arriving at a final formulation as  $YX = Y$  where  $Y$  denotes  $Y$  with normalized columns. In an example, this logic is utilized to solve the problem of segmenting multiple motions in videos (Elhamifar and Vidal, 2009, 2013). After solving for  $X$  through  $\ell_1$ -norm constraint optimization, a similarity matrix is further calculated by  $|X| + |X^T|$ , which is then processed by spectral clustering for final segmentation. Experiments on chosen video sequences show that this approach is exceptionally successful in this clustering task.

If above  $YX = Y$  is solved for  $X$  with a greedy approach, such as MP or OMP with an  $\ell_0$ -norm constraint where  $k = 1$ , then the non-zero coefficient with the index  $j$  within  $X_j$ , will show that  $y^*$  and  $y$ , are the most similar in terms of directionality; in other words  $y^*$  and  $y_j$  are highly correlated in terms of angular similarity. This can be regarded as an alternative similarity measure to Euclidean distance. Note also that for any sparsity constraint with  $k > 1$ , this formulation can be generalized as directional decomposition of the data.

As a relevant note, there is in fact a whole field of directional statistics (Mardia and Jupp, 2009; Mardia, 2014), in which data points are represented as scaled directions -contrary to points in cartesian coordinates- and their distributions are examined from that perspective. In that regard, Von Mises-Fisher probability formulation deals with distributions on circles, spheres, or in general  $n$ -dimensional hyperspheres (Fisher, 1953). In relation to the clustering problem, cosine similarity or angular distance as its inverse, can be used alternatively. Spherical  $k$ -means (Zhong, 2005) aims to maximize the cosine similarity objective, thus it is equivalent to  $k$ -means clustering on a unit hypersphere. Von Mises-Fisher directional distributions can also be used as a probabilistic approach for clustering (Banerjee et al., 2005; Gopal and Yang, 2014).

$0$					
		$0$	$0$		$0$
$0$		$0$	$0$		$0$
$h_3$		$0$	$j_3$		$-3$
$h_4$		$0$	$0$		$u$
$h_5$		$0$	$0$		$'_5$
		$0$	$0$		$0$
$0$		$0$	$0$		$0$
		$b$	$b$		$0$
		$\wedge_8$	$0$		$0$
$0$		$j_9$	$j_9$		$0$
		$0$	$0$		$0$
$0$		$0$	$0$		$0$

Figure 2. An example sparse coding case: (left) a portion of the dictionary is shown with colored columns (atoms), and (right) example sparse code-patterns colored with the same color as the atoms used during coding. There are two disjoint subspaces (green and blue code-patterns), and there also exists non-structural code-patterns (green-yellow and blue-yellow).

### 2.2.2.1 Structured sparsity as a means of clustering

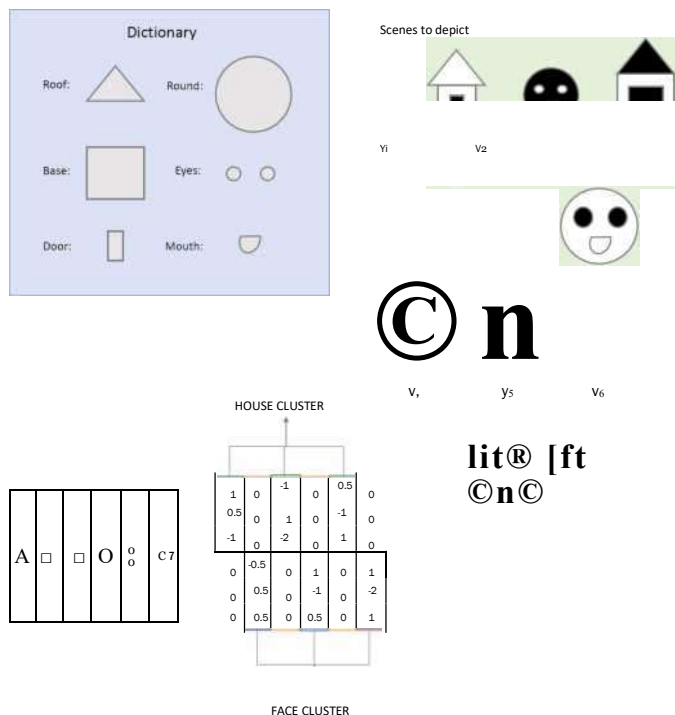
Instead of using sparse codes as features for the existing clustering algorithms, sparse code appearance patterns can directly be utilized for clustering. While noting that there is no structural constraint on code-patterns in the conventional sparse coding approaches, additional structural code-pattern constraints can easily be injected and then manipulated so that subspaces can directly be designated as clusters. As a simple example, Fig. 2 depicts a coding case where, as well as two disjoint subspaces as desired (green and blue code-patterns), there also exists non-structural patterns (green and yellow, blue and yellow). Structural constraints can enforce the condition that all sparse code-patterns appear in disjoint subspaces, which will naturally designate structured clusters.

As a structured sparsity technique, *group sparsity* enforces grouping of the elements belonging to the sparse-code vectors by allowing coefficients to fill the vector group-by-group. The sparse code is conceptually partitioned into overlapping or disjoint groups, and an additional norm constraint is used on this group level. In this kind of structure, there is a cascade of norm constraints, usually two-layered, as opposed to a single, general one in the conventional sparse coding approaches. As an

example,  $l_1$ - norm constraint based well-known *lasso* (Tibshirani, 1996) method can be extended to *group lasso* (Meier et al., 2008) with an  $\ell_2$ -norm constraint on the group level.

Considering the sparsity concept as a cascade of norm constraints forced on sparse codes on multiple levels leads to the possibility of multiple norm combinations, and also to other structural variations. An example is *sparse group lasso* (Friedman et al., 2010), which extends group lasso through a global  $\ell_1$ -norm constraint in addition to  $\ell_1$ -norm group sparsity and  $\ell_2$ -norm within group constraints. Such an enforcement yields sparsity both on the group and global levels, whereas group lasso alone does not enforce sparsity within a group. As a second example, strong group sparsity (Huang and Zhang, 2010) has  $\ell_1$ -norm within group constraint, and the support selected is restricted to that lying within the smallest possible subset of non-overlapping groups. Through a generalization, any structure can further be imposed on the sparse code set. In a recent study (Huang et al., 2011), group sparsity was extended through a subset imposing cost function while defining the coding complexity for that sparse subset. Note that by manipulating such cost functions, a range, including block, hierarchical or even graph sparse code-patterns can be enforced. It is important to keep in mind that structured sparsity is a sparse coding approach that will work particularly well if the data itself has that specific structural nature (Huang and Zhang, 2010). Without initial structure, the structured coding will much be less meaningful.

Most relevant to the clustering problem, *block sparsity* (Eldar and Mishali, 2009; Eldar and Boleskei, 2009; Elhamifar and Vidal, 2009) is a specific case of disjoint group sparsity where groups appear in blocks. A block sparsity of level 1 corresponds to some designation of disjoint subspaces. In such case, these subspaces can directly be assigned as clusters and that will correspond to a non-overlapping subspace clustering scheme as depicted in Fig. 3.



**Figure 3.** A block sparsity example illustrating the designation of disjoint subspaces which are assigned as clusters: house (green) and face (blue) clusters. The dictionary and the data represented in the sparse domain are shown on the top-row.

## 2.3 Dictionary Learning and Clustering

### 2.3.1 Dictionary learning: An overview

A crucial question in sparse representations is the choice of the dictionary. Possible choices include various sets of analytic waveforms such as overcomplete DFT, DCT, wavelets. However, both the sparsity and the quality of the representation depend on the used dictionary, and most importantly its suitability for the data and the problem at hand. Therefore, the underlying main idea of dictionary learning for sparse representations suggests that the data can be better approximated sparsely as a weighted linear combination of a set of *prelearned* dictionary atoms, rather than off-the-shelf overcomplete bases or dictionaries, e.g., (Elad and Aharon, 2006; Protter and Elad, 2009; Peyre, 2009; Mairal et al., 2008b,c; Bryt and Elad, 2008; Fadili et al., 2007). The



sparsity constraint associated with the learning problem generally leads to a solution which can fit any practical application by means of pursuit algorithms with  $\ell_0$ -norm and  $\ell_1$ -norm sparsity measures.

The objective is to obtain an explicit dictionary matrix  $\mathbf{A}$  which is optimally representative of a given set of training samples under some strict sparsity constraints. Formally, given a set of training data with  $N$  features and  $J$  samples stored in the columns of a matrix  $\mathbf{T}$ , the search for an optimum dictionary  $\mathbf{A}$  involves solving the constrained minimization as

$$\arg\min_{\mathbf{A}, \mathbf{Z}} \|\mathbf{T} - \mathbf{AZ}\|_F^2, \text{ subject to } \|\mathbf{z}_j\|_0 \leq k \quad \forall j \quad (4)$$

where the sparse matrix  $\mathbf{Z} \in \mathbb{R}^{M \times J}$  has its columns  $\mathbf{z}_j$  as the sparse representation vectors of the corresponding training samples  $\mathbf{t}_j$ . Note that the constraint on  $\mathbf{A}$  is implicitly assumed to be valid to obtain unit norm atoms. Here  $\|\cdot\|_F$  denotes the Frobenius norm.

The problem in Eqn. 4 is combinatorial and highly non-convex, and thus a local minimum can be expected (Rubinstein et al., 2010a). Alternatively, this formulation can be rewritten as a joint optimization with respect to the dictionary  $\mathbf{A}$  and sparse vectors  $\mathbf{z}_j$  while including the sparsity constraint in the formula as a penalty term

$$\arg\min_{\mathbf{A}} \sum_{j=1}^J \left\{ \arg\min_{\mathbf{z}_j} [\|\mathbf{t}_j - \mathbf{A}\mathbf{z}_j\|_2^2 + \alpha_j \|\mathbf{z}_j\|_0] \right\}$$

which is not jointly convex but convex with respect to one of its variables when the other one is fixed (Elad and Aharon, 2006).  $\alpha_j$  here represents the sparsity regularization parameter for  $\mathbf{t}_j$ .

In this way, the whole problem can be factorized into two *approximate* convex optimization steps as: a) *sparse coding*: optimizing  $\mathbf{Z}$  by fixing  $\mathbf{A}$ ; b) *dictionary update*: optimizing  $\mathbf{A}$  by fixing  $\mathbf{z}_j$ . A solution can be reached by iteratively solving these two steps.

While sparse codes  $z_j M_j$  can be calculated in the sparse coding step as discussed in Sec. 2.2, the problem is then to optimize  $A$  by minimizing the representation error of the training samples. This optimization is known as *the dictionary update problem*, and it can be formulated as

$$\arg \min_{A \in \mathbb{R}^{n \times m}} \sum_{j=1}^J \|z_j - A z_j\|_2^2 \quad (6)$$

The above described optimization problem can be solved using various techniques. Non-parametric dictionary learning methods, such as Method of Optimal Directions (MOD) (Engan et al., 1999) and K-SVD (Aharon et al., 2006), have been developed, resulting in non-structural learned dictionaries. There are also parametric learning structures for such as *translation invariant dictionaries* (Blumensath and Davies, 2006; Jost et al., 2006; Aharon and Elad, 2008; Engan et al., 2007), *multiscale dictionaries* (Mairal et al., 2008c; Sallee and Olshausen, 2003), *unions of orthonormal bases* (Lesage et al., 2005; Sezer et al., 2008) and *sparse dictionaries* (Rubinstein et al., 2010b). Moreover, for various data and signal processing tasks, the literature provides online learning algorithms (Mairal et al., 2010; Skretting and Engan, 2010), task-driven learning approaches (Mairal et al., 2011), tree-structured hierarchical methods (Monaci et al., 2004; Nakashizuka et al., 2009; Jenatton et al., 2011), and iteration- tuned schemes (Zepeda, 2010; Zepeda et al., 2011).

### 2.3.2 Dictionary learning linked with clustering

As mentioned above, a chosen fixed dictionary may not always be appropriate for clustering the data at hand, especially when the data under investigation is of an unknown nature. This situation is exemplified in Fig. 4, which builds on top of the previously selected example, depicting that sparse codes are not structured, but lie apparently on intersecting subspaces. There is no obvious cluster-like appearance in sparse codes. However, it is possible to acquire disjoint subspaces through adapting the dictionary to the data at hand, by learning a more suitable dictionary.

Constraining block sparsity structure onto sparse codes using the sparse coding step, followed by a dictionary learning step, basically corresponds to learning a block- sparse model for the data. This is a form of non-overlapping clustering,

where distinct subspaces are defined by adaptive subdictionaries together with the supporting block of

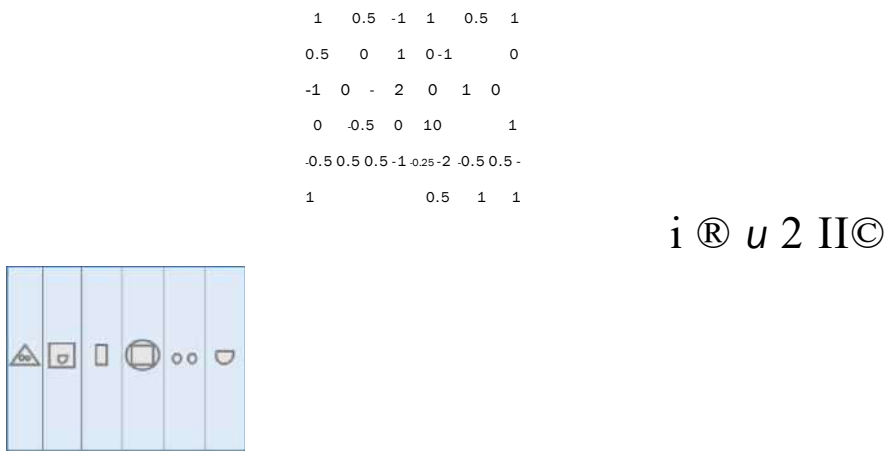


Figure 4. The chosen dictionary may not always be appropriate for clustering the data. An example sparse decomposition depicts that sparse codes are not structured, but lie on intersecting subspaces.

sparse codes. Examples from literature prove that such an approach is a successful alternative to self-similarity (Sprechmann and Sapiro, 2010). As an example in (Ramirez et al., 2010), subdictionaries are modeled to learn a subspace for each cluster in an image segmentation task. Initially, data samples are assigned to best representing subdictionary according to a certain representation quality that includes a data fidelity term and a sparsity promoting term. Then, these assignments are fixed, and solutions for better adapted subdictionaries are calculated with an additional incoherence term. This proves to be an efficient and effective solution for the image segmentation problem targeted. A successful denoising application example can also be given in (Dong et al., 2011). Here, the concept of double-header  $\ell_1$ -optimization is introduced with an additional  $\ell_1$ -norm restriction that enforces best representing centroid for each cluster through an adaptive dictionary. Simultaneous centroid enforcement and sparse coding create a noise-resilient structure. Encouragingly, this denoising application is reported to match the state-

of-the-art BM3D (Dabov et al., 2007) performance. In fact, denoising is a very suitable domain for such clustering formulations. Yet in another denoising study in (Huang et al., 2014), authors successfully aim at the decomposition of images into multiple semantic layers through unsupervised clustering based on self-learning, allowing detection and removal of undesired patterns such as Gaussian noise and rain strikes.

## 2.4 Related Concepts

### 2.4.1 Signal processing perspective

Compressive sensing (CS) aims at reducing the number of measurements needed to describe a signal while exploiting its compressibility. It can mathematically be expressed by

$$z = \Phi y = \Phi \Psi x \quad (7)$$

where  $\Phi \in \mathbb{R}^{N \times M}$  is the stable measurement matrix with  $N < M$ , and it is responsible of dimensionality reduction from  $y \in \mathbb{R}^M$  to  $z \in \mathbb{R}^N$  such that  $z$  designates less number of measurements taken. The main goal is to recover the original compressible signal  $y$ , or equivalently the sparse signal  $x$ , from  $z$ . Note here that  $\Psi \in \mathbb{R}^{M \times M}$  is an orthonormal sparsifying basis for obtaining  $k$ -sparse representation signal  $x$  such that  $y = \Psi x$  and  $x = \Psi^T y$ .  $k$  largest coefficients in  $x$  are kept while discarding the smallest for  $k \ll M$  (Baraniuk, 2007). The solution to this problem involves two steps. First, a suitable  $\Phi$  has to be designed, and then a reconstruction algorithm is needed to recover  $y$  from  $z$ . For stability, a sufficient condition is that  $\Phi \Psi = \Phi \Psi^T$  satisfies the restricted isometry property (RIP) (Candes et al., 2006). An alternative approach for stability is to ensure that the matrix  $\Phi$  is incoherent with the sparsifying basis. However, in practice, the signal  $y$  at hand may not be sufficiently sparse in an orthonormal basis, but in a redundant and overcomplete dictionary. Through a generalization, then can be replaced with a highly overcomplete and coherent dictionary  $A$  tying the gap between CS and sparse representations (Candes et al., 2011).

Considering compressible signals in a “structured nature” paves way to

model-based CS. These methods significantly decrease the bound of required measurements to  $N = \mathcal{O}(k)$  for tree-sparse and (in the limit) for block-sparse signals, whereas standard CS models can robustly recover  $k$ -sparse compressible signals from  $N = \mathcal{O}(k \log(M/k))$  measurements. Such approaches to CS have helped to decrease the required amount of measurements for robust recovery of signals in applicable domains. Similar methods can also be used for CS recovery of clustered signals (Baraniuk et al., 2010; Cevher et al., 2009; Yu et al., 2012).

Up to this point, sparse and redundant representations have been considered through a single measurement vector (SMV) framework, in which each signal is considered individually, even though sparse representations solution is obtained for multiple signals. In the multiple measurement vectors (MMV) approach, on the other hand, multiple signals are simultaneously considered by processing multiple sparse vectors together while selecting a column (an atom) from the dictionary  $A$ . The optimization of MMV can be formulated as

$$\arg \min_{\mathbf{X}} R(\mathbf{X}) \text{ subject to } \mathbf{Y} = \mathbf{A}\mathbf{X} \quad (8)$$

where  $R(\mathbf{X})$  represents the number of rows in  $\mathbf{X}$  containing non-zero entries (Cotter et al., 2005; Chen and Huo, 2006).

The perspective of MMV is especially powerful when solutions have an initial common sparsity profile. However, dictionary learning with MMV approach will be extremely ill-posed because all-zero rows in  $\mathbf{X}$  may cause corresponding dictionary atoms either to disappear or to diverge during the dictionary update step. However, this approach can be successfully used to discard certain atoms from a highly overcomplete dictionary to obtain a more compact representation. As a final relevant note, algorithms similar to ones used for MMV recovery can be adapted for the block sparsity structure, thus can be linked to clustering (Davenport et al., 2012; Yuan and Lin, 2006; Eldar et al., 2010; Eldar and Mishali, 2009).

#### 2.4.2 Machine learning perspective

Three conceptually different machine learning methods are reviewed in

relation with sparse representations. These methods include support vector machines to be mentioned along large margin modeling (Steinwart and Christmann, 2008), decision trees as examples of symbolic machine learning (Quinlan, 1986), and neural networks as a connectivist approach.

Large margin modeling can be defined as finding hyperplanes which maximize the margin between classes (Cortes and Vapnik, 1995; Tsochantaridis et al., 2005; Cevikalp et al., 2010). Such a model is composed of a bias, weight vectors and support vectors. Support vectors are selected as data points which lie closest to hyperplanes, and these are sufficient to express the whole data set. In other words, support vectors lie on the margin and, for certain applications, carry all the relevant information about the data. Thus, the solution is sparse in nature. It has been shown that a slight modification of  $\ell_1$ -norm sparsity optimization method, namely Basis Pursuit, is equivalent to Support Vector Machines (SVMs) (Steinwart and Christmann, 2008), which are large margin formulations (Girosi, 1998). Building on top, large margin clustering is also possible through maximizing inter-cluster margins (Xu et al., 2004; Zhang et al., 2009). Therefore, drawing parallels between the two variants of this approach can lead to a more generalized theory that is able to capture the gist of the sparsity concept analytically, since both variants can be thought as mathematically constructive methods.

Symbolic machine learning is traditionally associated with ID3 decision tree learning (Quinlan, 1986). In general, the symbolic approach to machine learning can be thought as inductive learning, in which certain rules are inversely deduced from the observed data. However, symbolism has a broader presence in the AI world in general, as a means of high-level abstraction over numerical units, often introducing human-readable representations (Haugeland, 1989). In line with this definition, model-based clustering can be classified as an approach to symbolism. For example, centroids in k-means, as rather shallow symbols, with a distance rule for assignments, define an abstract object that provides partitioning. Similarly, large margin modeling can be seen as another shallow symbolic approach; in this case, support vectors with their strict boundaries provide an abstraction layer. In this sense, symbols, as abstracted objects, provided with rules for decisions can be regarded as

sparse representations, since this approach allows a relatively small number of symbols to express enormous amounts of data. It is important to note that the shallow analogies given here may not be common, as symbols generally need to be very high-level abstractions, as in (Gowda and Diday, 1992,1991).

Connectivist approaches, such as traditional neural networks (Du, 2010), tend to process data in pure numerical units. Perceptron is a generalization of a single neuron cell that works on the numerical unit level (Rosenblatt, 1958). However, with only a single layer, perceptrons are not capable of learning the nonlinearity. A multi-layer generalization of perceptrons solves this problem, while also introducing a possibility for sparsity through the activation function (Rosenblatt, 1961). However, such generalization still depends on numerical units for computation. Convolutional Neural Networks (CNNs) provide an abstraction over multi-layer structure, in which a degree of “symbolism” is introduced, as apparent from the human-readable filters that are formed within the nodes (Krizhevsky et al., 2012). Note that connectivism, rather than enabling deep understanding, simply replicates the evolved structure of the human brain, unlike analytic approaches. As a recently popularized approach, clustering with deep learning (Schmidhuber, 2015; Hershey et al., 2016) at this stage may be successful, but currently it is not sufficient to provide a deep analytic understanding of inner-working procedures of sparse structures and clustering peculiarities.

## ***2.5 Shortcomings and a proposal***

One shortcoming of conventional dictionary learning for sparse representations is about initialization. In cases where systems are restricted to random initializations, a supposedly optimal state update based on such an improper dictionary might hamper the system from start conveying undesired effects to later iterations. Such shortcoming is addressed in a standalone manner in Appendix A, in which intermediate error codes are used to boost dictionary learning process especially in cases of random initializations.

However, there is a deeper problem that haunts sparse representations to be

applied as a standalone machine learning tool itself, besides being a feature transformation method. This problem is related to atom normalization, mentioned along with dictionary learning step. If normalization is not applied, the system diverges eventually. Because of normalization and lack of positional information, conventional dictionary learning, as a standalone tool, is not able to learn linearly non-separable datasets. For example, assuming two clusters as in Fig. 5, conventional sparse representations will not be able to learn the example given as atoms when normalized become unit directions and most importantly lose positional information. Two solutions are proposed to overcome this problem, namely *superatoms* or structural centroids.



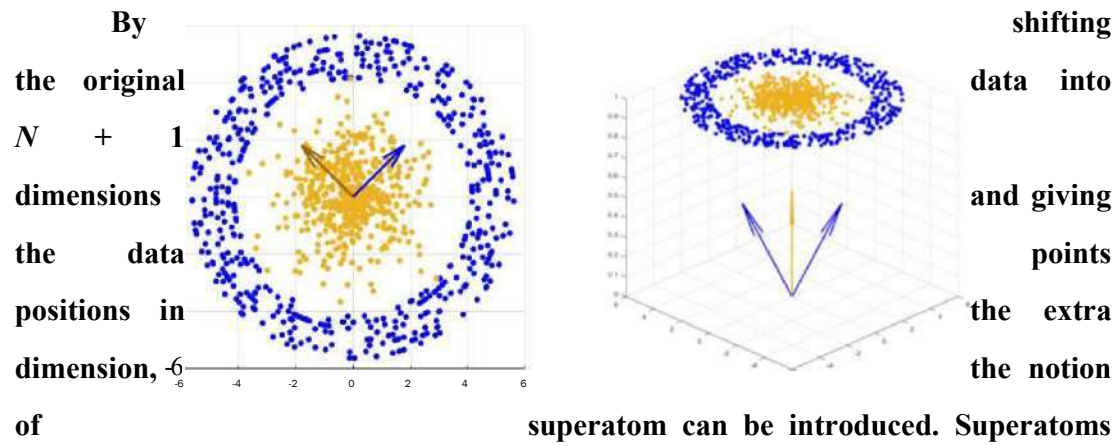


Figure 5. On the left, a drawback of normalizing the atoms is presented. Two classes cannot be discriminated. Superatoms on the right overcome this problem. will be normalized rays passing through the origin as exemplified on the right of Fig. 5. When sparsity is one, they are lines piercing the data space, effectively are points. When number of nonzeros are two, it corresponds to shooting planes intersecting the data space as an arbitrary line with possible offsets. By this procedure, positional behavior can be attained.

However, a neater and a more systematic solution is possible by restricting magnitudes of sparse codes, instead of normalizing the atoms (or superatoms). This is the topic of next chapter, where a generalization of  $k$ -means is systematically formulated within a sparse representations framework in which a centroid can now be structural, namely can be any piecewise linear construct, such as a line-segment, a triangle, a polygon, or a connected composition of these, instead of only being a point.

## CHAPTER 3: A-POLYTOPES CONCEPT

### 3.1 Introduction

It has already been proven that under certain circumstances dictionary learning for sparse representations is equivalent to conventional /c-means clustering. Through additional modifications on sparse representations, it is possible to generalize the notion of centroids to higher orders as noted in Sec. 2.5. Using higher dimensional, nonconvex prototypes may alleviate the curse of dimensionality while also enabling to model nonlinearly distributed datasets successfully. In this light, a systematic generalization of /c-means is targeted in this chapter.

A motivation to generalize /c-means emerges when one considers its shortcomings irrespective of its implementation, i.e., even assuming a globally optimal solution. First of all, it is possible to model only spherically shaped clusters; therefore, it will fail when clusters have nonconvex shapes. As another shortcoming, the number of clusters to search for has to be supplied by the user. A generalization in this context has to overcome one or both of these drawbacks without deviating too much from the original problem formulation.

To address the first issue, kernelized generalizations of /c-means have already been proposed (Scholkopf et al., 1998; Dhillon et al., 2004). However, cluster model shapes then depend on the kernel function chosen and this type of generalization fails to address the second issue. Ensemble clustering approaches (Fred and Jain, 2002; Hore et al., 2009; Iam-on and Garrett, 2010), based on multiple runs of /c-means can additionally model arbitrarily shaped clusters, but still do not provide a solution to the number of clusters issue. Another generalization attempt is /c\*-means (Cheung, 2003) where ellipse-shaped models can be learned while also discovering the cluster count. On the other hand, there have been studies that specifically target the second issue. In X-means (Pelleg and Moore, 2000), certain information criteria are used to find the inherent number of clusters using /c-means as a subroutine. Mean shift clustering (Cheng, 1995) is another related generalization where number of clusters

is not pre-specified, but it runs much slower. These generalizations basically fail to address both issues in a unified way. Note here also that although these proposals include  $\ell_1$ -means as a subproblem, they do not utterly generalize the original problem definition. In other words, they are not superproblems of  $\ell_1$ -means, but approach the  $\ell_1$ -means problem in generalized manners.

Through elevating the notion of centroids (central prototypes) to higher dimensions, it is possible to generalize the problem definition itself. In the related  $\ell_1$ -planes approach (Bradley and Mangasarian, 2000), planes are chosen as central prototypes instead of points. More generally,  $\ell_1$ -dimensional flats (e.g., 0-flat is a point, 1-flat is a line, 2-flat is a plane) can be chosen as central prototypes (Tseng, 2000; Canas et al., 2012). One may notice that  $\ell_1$ -means is a  $\ell_1$ -flats problem, because a centroid point is a 0-flat, namely a zero-dimensional flat. Hence,  $\ell_1$ -means might as well be called  $k$ -0-flats.

The fact is that a central prototype can be of higher dimensions having an arbitrary shape. By replacing the keyword “the centroid point” with any geometrical construct one can arrive at many other formulations such as  $\ell_1$ -lines,  $\ell_1$ -triangles,  $k$ -rectangles, or even  $k$ -polygons. To consider these novel concepts in a structured way (instead of going case-wise), one needs to adopt a unifying framework. An interesting aspect of this line of generalization is its close relation with sparse and redundant representations. Mathematically, it can be shown that both  $\ell_1$ -means and  $\ell_1$ -flats are of the form of dictionary learning with additional constraints on sparse representations (Szlam and Sapiro, 2009). Thus, a modified sparse representations framework appears to be a promising candidate for unification as already mentioned in Sec. 2.5

In this chapter, two superproblems of  $\ell_1$ -means, namely  $\ell_1$ -simplexes and  $\ell_1$ -polytopes are formulated through a novel sparse representations framework. This framework, not only generalizes  $\ell_1$ -means to provide solutions to both issues mentioned above, but also introduces a new geometric perspective to sparse representations by breaking it away from its subspace-centric viewpoint through the usage of positional and bounded (possibly nonconvex) spaces. This conceptual breakthrough has a superior value on its own although a rather naive solution to  $\ell_1$ -polytopes is presented. In this quest, structured sparsity, more specifically block-

**sparsity concept mentioned in**

previous chapter has to be mathematically formulated.

### 3.1.1 Formulating block-sparsity

In the conventional sparse representations framework, a constrained optimization problem is solved as already given in Eqn. 4. As noted before, there is no structural constraint on code-patterns in the conventional sparse coding approaches. Most relevant to the clustering problem, *block sparsity* (Eldar and Mishali, 2009; Elhamifar and Vidal, 2009) is yet a specific case of disjoint group sparsity where groups appear in blocks. This optimization problem is formulated in Eqn. (9) as

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{d}} & \|\mathbf{y} - [\mathbf{A}_1 \dots \mathbf{A}_K] \mathbf{x}\|_2 \quad \text{s.t.} \\ & \mathbf{d} \in \{1, \dots, K\} \\ & (\mathbf{I} - \mathbf{V} \mathbf{V}^T) \mathbf{A}_k \mathbf{x} = \mathbf{0} \quad \text{for } k \neq \mathbf{d} \end{aligned} \quad (9)$$

where  $k^*$  indicates the index of the optimal sub-dictionary for the data point  $\mathbf{y}$ . The constraint  $k \neq k^* \Rightarrow \mathbf{x}_k = \mathbf{0}$  ensures the block sparsity, and  $\mathbf{x}_k$  represents sparse coefficients other than the assignment block, i.e., that are all forced to be zero. In fact, a sub-dictionary here can be thought of as a central prototype that claims an entity (or not) depending on its reconstruction error, much like a centroid claiming the closest points to itself. Therefore, sparse coding with block sparsity determines which entities will be assigned to which central prototypes. To extend this analogy, dictionary update hence corresponds to updating the central prototypes (i.e., sub-dictionaries) which follows the assignment step.

### 3.1.2 K-means within a sparse framework

A close relationship between sparse representations and  $k$ -means clustering has been drawn in *k-SVD* (Aharon et al., 2006). As formulated in Eqn. (10), the sparse code  $\mathbf{x}_j, \mathbf{v}_i$ , has only one non-zero entry because of the constraint  $\|\mathbf{x}_j\|_0 = 1$ , corresponding to the simplest case of block sparsity, namely blocks of size 1. If those sparse-code entries are forced to be positive and sum-to-one, then it is a direct formulation of the classical  $k$ -means clustering problem. Dictionary elements (atoms) here directly designate centroids.

$$\min_{\mathbf{x}} \|\mathbf{y}_i - \mathbf{A} \mathbf{x}\|_2 \quad \text{s.t.}$$

$$\begin{aligned} & \mathbf{A}^T \{\mathbf{x}_i\} \mathbf{i} \\ & \|\mathbf{x}_j\|_0 = 1 \wedge \|\mathbf{x}_i\|_1 = 1 \wedge \mathbf{0} < \mathbf{x}_i, \mathbf{V}^*. \end{aligned} \tag{10}$$

In the recent years, many different paths based on above equation have been taken to ameliorate its drawbacks by either relaxing these restrictions or introducing additional terms (Yu et al., 2009; Wang et al., 2010; Zhang et al., 2015). On the contrary, the proposed study tries to generalize this equation to higher-order blocks while respecting all these original constraints.

### 3.2 *K-simplexes*

For a general block-sparse formulation in Eqn. (9),  $\mathbf{x}_i, \mathbf{V}_i$ , is restricted to be block sparse with a block size of  $q$  and there are  $k$  such blocks. Hence, this setup directly corresponds to  $/c$ -subspaces of dimension  $q$  (Szlam and Sapiro, 2009). In other words, central prototypes are subspaces. By definition, a subspace is a flat that passes through the origin, and  $/c$ -subspaces will most probably fail if there are more than one cluster within the same subspace. In that case, flats having arbitrary offsets will have more flexibility for the problem of clustering. On the other hand, there has not been a keen consideration on how to exactly formulate  $/c$ -flats within the sparse representations framework. One of the contributions of this chapter is to show that this is possible through introducing a sum-to-one constraint, i.e.,  $\mathbf{1}^T \mathbf{x}_i = 1$ , within the block-sparsity setting, and this will pave the way to formulating  $/c$ -simplexes.

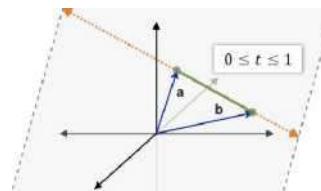
**Table 1. Clustering frameworks and respective constraints.**

	$\ \mathbf{x}_i\ _0$	$\mathbf{1}^T \mathbf{x}_i = 1$	$\mathbf{x}_i \geq \mathbf{0}$	$\mathbf{0} < \mathbf{x}_i$
$k$ - means	$= 1$	/	/	/
$/c$ -subspaces	$< q$	X	X	X
$Zc$ -flats	$< q + 1$	/	X	X
$Zc$ -simplexes	$< q + 1$	/	/	/

(c) Simplex

$$t_1 a + t_2 b$$

$$t a + (1 - t) b$$



(a) Subspace

(b) Hat

**Figure 6. A simple example of how additional constraints on sparse codes affect the solution of sparse representations, (a) The conventional sparsity constraint together with (b) sum-to-one ( $t_1 + t_2 = 1$ ) and (c) sum-to-one and nonnegativity ( $t_1 + t_2 = 1$  and  $t_1, t_2 > 0$ ) constraints.**

Consider that there are two arbitrary vectors  $a$  and  $b$  in an arbitrary space of dimension  $2Z^n$ . These vectors basically correspond to two specific points in a two-dimensional subspace of  $2Z^n$ . Within this subspace, the expression  $t a + (1 - t) b$  traces the line passing through both  $a$  and  $b$ , where  $t$  is any real number. In a more specific case, if  $t$  and  $1 - t$  are restricted to be non-negative, then the aforementioned equation represents a local line-segment connecting these  $a$  and  $b$  points. Note here that if this expression is written in the form of a matrix-vector multiplication as  $[a \ b] [t \ (1 - t)]^T$ , it is apparent that column vectors  $a$  and  $b$  correspond to dictionary atoms and the vector corresponds the weighting coefficients of the sparse representations framework, where sum-to-one constraint is also satisfied. A depiction of these cases is given in Fig. 6.

Let us now consider three linearly independent vectors and sum-to-one constraint on the weighting coefficients. In this case, a plane passing through these three vector points will be traced, and a triangle will be the geometric equivalent with an additional non-negativity constraint. Generalizing this, it is possible to observe that the constraint  $1^T x = 1$  replaces a  $(q-1)$ -dimensional subspace with a  $(q-1)$ -dimensional flat lying within that subspace, and further with a  $(q-1)$ -dimensional simplex within

that flat with the additional  $0 < x^*$ . On top of this generalization, in a block-sparse formulation, each block corresponds to one of these single geometric object systems. Hence, each sub-dictionary represents a prototype, and if there are  $k$  number of sub



dictionaries then one arrives at fc-subspaces, fc-flats and fc-simplexes, respectively. A summary of these clustering frameworks with their corresponding constraints are presented in Table 1. A-simplexes appear to be the most consistent generalization that respects the original constraints in /c-means.

In fact, when  $k$  number of  $g$ -dimensional simplexes are simultaneously to be fit to the data, the optimization problem takes the form of Eqn. (11) involving also block sparsity as

$$\begin{aligned} \underset{\substack{\mathbf{A} \in \mathbb{R}^{n \times (q+1)} \\ \mathbf{V} \in \mathbb{R}^{n \times k}}}{\text{argmin}} \quad & \|\mathbf{y}^* - [\mathbf{A}_1 \dots \mathbf{A}_k] \mathbf{x}\| \quad \text{s.t.} \\ & (\|\mathbf{x}^*\|_0 \leq 1 \wedge \|\mathbf{x}_j\|_1 = 1 \wedge 0 < \mathbf{x}_j, \forall i) \wedge \\ & (\mathbf{A}_k \in K^{n \times (q+1)} \setminus \forall k) \wedge (k \wedge k^* \Rightarrow \mathbf{x}^* = 0), \end{aligned}$$

where there are  $k$  simplexes being used in total,  $A_k$  denotes the  $n^{th}$  simplex and each  $A_k$  has  $q + 1$  columns therefore simplexes are of dimension  $q$ . Note that  $\|\mathbf{x}_j\|_1 = 1$  and  $0 < \mathbf{x}_j$  together imply  $\mathbf{1}^T \mathbf{x}_j = 1$  just as in the case of /c-means. As mentioned before, the restriction of  $K \wedge K^* \Rightarrow \mathbf{x}^* = 0$  ensures block sparsity where  $K^*$  is the closest simplex for the data point  $\mathbf{y}^*$ . For the case when  $q = 0$ , this boils down to the formulation of  $k$ -means given in Eqn. (10), logically zero-dimensional simplex being a point.

### 3.2.1 A solution to $kr$ -simplexes

The optimization problem given in Eqn. (11) is highly nonconvex. Therefore, similar to standard sparse representations, an iterative solution to A;-simplexes can be given through alternating between sparse coding and dictionary update. From the viewpoint of prototype-based clustering, one needs to be able to calculate the distance between a given data point and an arbitrary simplex (Golubitsky et al., 2012) so that the point can be assigned to the proper prototype, i.e., the assignment step. In sparse coding terms, this corresponds to orthogonal projection of  $\mathbf{y}^*$  onto the closest simplex, resulting in weighting coefficients representing the positive barycentric coordinates of the projection point. Note that sum-to-one constraint enforces invariance to translations and rotations. Invariance to rescaling follows from the formulation itself. After acquiring sparse representation vectors  $\mathbf{x}_j, \mathbf{V}_i$ , conventional



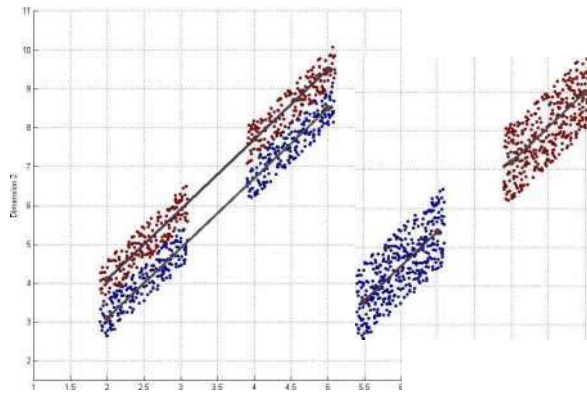


Figure 7. For the given two clusters: (left) a general solution to /c-simplexes is presented, i.e., there is no restriction on the sizes of simplexes, (right) a more natural clustering occurs when simplex sizes are restricted. (Best visualized in color.)

can be used, which corresponds to the prototype update.

### 3.2.2 Complexity analysis

Each data point  $y^*$  will be projected onto each simplex (i.e., there are  $k$  simplexes), then assigned to the closest one. Projection onto a single simplex is claimed to have a time complexity of  $O(n)$  where  $n$  is the dimension of data space (Duchi et al., 2008). In a sensible model, for each data point there must be at most one simplex, resulting in a bound  $k < m$  and  $m$  is the number of data points. Therefore, complexity of the coding phase is  $O(m^2n)$ . Updating prototypes through Moore-Penrose inverse has a time complexity of  $O(m^2kq)$ , where  $kq$  denotes the total number of atoms (columns) in  $A$ . This atom count can be at most  $m$  in a sensible model, arriving at a complexity of  $O(m^3)$ . Since overcompleteness implies  $n < kq < m$ , this phase seems as the bottleneck.

### 3.2.3 Drawbacks of $k$ -simplexes

Although it is promising when conceptually compared to fc-means or fc-flats, there are two major drawbacks of fc-simplexes for the clustering problem. The first

drawback is that simplexes can appear in many forms with different sizes. For a given data distribution, there will be infinitely many best-fit simplexes with varying sizes and these simplexes may not be apparent solutions to the clustering problem as depicted in Fig. 7. It is clear that an optimization only for minimizing the approximation error may not result in natural clusters. As a solution, an additional term that will penalize the size of simplexes can be introduced in Eqn. (11), which leads to a multi-objective optimization problem. An alternative approach would be to limit the size of simplexes, and this approach has been taken in this chapter as it does not require a joint optimization. For the rest of this chapter,  $r$  will denote the maximum allowed edge size of a simplex. The second drawback is that simplexes are convex structures. Therefore, a single simplex will not be enough to model a cluster if clusters appear in nonlinear forms. It is highly desired to have prototypes that can have nonconvex shapes. To be able to overcome this drawback, further generalization of  $/c$ -simplexes is possible through the concept of  $/c$ -polytopes.

### 3.3 $K$ -polytopes

This thesis sticks with the definition that a polytope is an intact object that admits an exact simplicial decomposition. For example, a polygon (a two-dimensional polytope) always admits a triangulation. Then in general terms, a polytope is a connected composition of many simplexes.

Having pinned down the formulation of  $A_i$ -simplexes, formulating  $/c$ -polytopes is simpler as given in Eqn. (12)

$$\begin{aligned} & \text{argmin}_{\mathbf{V}} \|\mathbf{y}_i - [\mathbf{A}_x \dots \mathbf{A}_K \dots \mathbf{A}^*] \mathbf{x}^*\| \text{ s.t.} \\ & \mathbf{A}_i, \{\mathbf{x}_i\}_i \\ & (\|\mathbf{x}_j\|_0 \leq q + 1 \wedge \|\mathbf{x}_j\|_1 = 1 \wedge 0 < \mathbf{x}_j, \forall i) \wedge \\ & (\mathbf{A}^* \in \mathbb{R}^{n \times p} \wedge (q + 1) < p, \forall k) \wedge (k \in K^* \Rightarrow \mathbf{x}_k = \mathbf{0}), \end{aligned}$$

where the constraint on sub-dictionaries  $\mathbf{A}_K$  in Eqn. (11) is replaced with a new term, which means that a prototype is still  $g$ -dimensional but has  $q + 1 < p$  vertices.

In contrary to  $/c$ -simplexes in which there is indeed  $(\wedge) = 1$  possible projection within a block (i.e., projection onto the simplex defined by  $\mathbf{A}_K \in \mathbb{R}^{n \times (q+1)}$ ),

Eqn. (12) suggests new prototypes being more general and there are ( many  $q$ -dimensional simplexes to test for within each single block. In simpler terms, many different polytopes can be defined over  $p$  number of vertices and those different polytopes should be distinguishable. Therefore, an additional structure is needed to define the shape of a polytope. This translates to keeping a set of valid simplexes within the polytope among all possible ( of them, for each  $A_{re}$ . In the remaining part of this chapter, the set of valid simplexes will be denoted as  $H_K, \forall k$ .

An important observation at this point is that the set  $H_K$  refers to a connected hypergraph. First of all,  $K_K$  indexes  $A_{re}$  and each entry within the set ' $H_K$  refers to a valid simplex without possessing any positional information. Vertices are abstracted out as nodes of the hypergraph and each valid simplex corresponds to a hyperedge that relates  $(q + 1)$  nodes. Note here that the most general form of  $/c$ -polytopes can be attained if a hyperedge is not only allowed to relate  $(q + 1)$  nodes but also fewer. However, this utmost generalization is beyond the scope of this chapter and only hyperedges that relate exactly  $(q + 1)$  nodes are considered. Such generalization will be the topic of Chapter 4.

If the hypergraph is not connected, it means there are actually two or more polytopes defined over  $A_K$ , and thus  $A_K$  is in fact composed of more than one block that indeed contradicts the hypothesis of  $k$  polytopes to start with. There is also another possibility that although a simplex is valid within a polytope, it may not receive any projections. Namely, there is no data point that is projected to that simplex, hence the simplex is redundant. Additionally, some of these simplexes might not be redundant but may receive just a few projections. Therefore, a new constraint can be applied to the set  $H_K$  such that the hyperedge  $h_K \notin 'H_K$  should claim at least  $A$  many data points. However, such constraint must not violate the condition that  $V_K$  being a connected hypergraph.

Let us denote a specific positional simplex as  $A_{Kih}$  that is defined by the hyperedge  $h_K$ , and as the number of data points those are claimed by that simplex. Any node of  $h_K$  simply corresponds to a vertex of  $A_{Kjh}$ . Let  $A^a_h$  and  $A^b_{K h}$  refer to two arbitrary vertices (or atoms)  $a$  and  $b$  of the simplex  $A_{Kjh}$ . Then the size of the edge connecting these nodes will be  $\| A^a_h - A^b_{K h} \|_2$  or simply  $\|a - b\|_2$ . Finally, let

**Table 2. Clustering success rates over 100 randomly initialized runs. Top value in each cell designates the mean, middle value is the maximum accuracy attained and the bottom value is the average running time.**

	A;-means	fc-flats	GMMs	kernel /c-means	ensemble fc-means	A;-polytopes
Crescents	88.6%	60.1%	76.9%	91.0%	100.0%	100.0%
	88.6%	91.3%	88.8%	99.8%	100.0%	100.0%
	0.01s	0.06s	0.01s	0.08s	4.56s	6.05s
Spirals	61.4%	51.3%	59.9%	61.3%	96.2%	99.9%
	61.9%	58.1%	65.5%	76.8%	100.0%	100.0%
	0.01s	0.14s	0.03s	0.12s	11.71s	23.12s
IRIS	92.0%	44.0%	73.9%	92.9%	96.1%	96.3%
	96.0%	93.3%	96.7%	98.0%	98.0%	98.0%
	0.01s	0.02s	0.02s	0.02s	1.38s	1.43s
MNIST	71.0%	43.1%	65.2%	70.5%	79.8%	78.2%
	77.8%	50.9%	82.5%	80.2%	83.7%	83.8%
	0.05s	0.23s	0.82s	0.68s	25.83s	32.83s

us denote the set  $S$  as the union of all  $H_K$  namely  $S = H_K$ , where  $S$  desirably be a hypergraph with  $k$  many connected components, each referring to the  $K^{th}$  individual polytope,  $K = 1 \dots k$ . The whole optimization problem then takes the form of Eqn. (13) such that

$$\begin{aligned}
 & \underset{A, \{x\}}{\operatorname{argmin}} Y'Hy^* - [A_x \dots A^* \dots A_{fc}]x < ||| \text{ s.t.} \\
 & e(||x_i||_0 < ? + 1 \wedge ||x_j||_1 = 1 \wedge 0 < X_j, \forall i) \wedge \\
 & (A^* \wedge n^{exp} \wedge (q + l) < p, \forall c) \wedge (/c \wedge K^* \Rightarrow xf = 0) \wedge \\
 & (A_{,,,,}, e^{\wedge x} (< i + i) \wedge A < vi \wedge ||A^*j - A^*, J_2 < r)
 \end{aligned}$$

where  $r$  denotes the maximum allowed edge size of a simplex as mentioned earlier.

### 3.3.1 An algorithm for $k$ -polytopes

Before presenting the proposed algorithm to tackle the problem, it can be immediately noticed that initializing the system with  $k$  many random polytopes will not be an effective strategy, although this scheme is usually applied in /c-means clustering (Celebi et al., 2013). Therefore, the proposed approach starts with a single random (/)-dimensional simplex which gradually evolves into  $k$  many meaningful polytopes, having evenly sized (/)-dimensional simplexes as faces. This is

accomplished through processes of subdivision, pruning and merging, depending on the parameters  $r$  and  $A$ .

While sticking with the conventional iterative process of sparse coding and dictionary update, two main problems arise: (i) the initial random simplex is probably not structurally correct, and (ii) dictionary update steps may introduce unevenly sized simplexes and some of these simplexes may become unnecessary. These two problems are addressed through an adaptive structured sparsity scheme as follows. Any simplex  $A_{Kjh}$  defined by the edge  $h_K \in U_K$  is subdivided appropriately when the size of at least one of its edges surpasses  $\tau$ , i.e., the simplex  $A_{Kjh}$  is divided into two simplexes if there exists at least one pair of vertices  $a$  and  $b$  such that  $\tau < \|a - b\|_2$ . As a result, this subdivision procedure introduces new simplexes  $A_{h'}$  and  $A_{Kyh}$  into  $A_K$ , and new valid hyperedges  $h'_K$  and  $h''$  into  $H_K$ . Similarly, if a simplex is not being used at all, or rarely used such that  $v_h^K < A$ , the corresponding hyperedge  $h_K$  is to be removed from  $\%_K$ . In addition, if there exists some vertices uniquely being used by the removed  $h_K$  then these vertices should also be removed from  $A_K$ . This procedure corresponds to the pruning stage.

Notice that the resulting polytopes will be grids of simplexes through pruning and subdividing, hence only a structurally approximate solution to the problem. For example, a T-shaped cluster in two-dimensions may not be effectively learned by the presented subdivision process. Another problem is that, pruning may introduce excessive number of clusters in cases of highly convoluted datasets. Through a merging process, close enough simplexes from different connected components (or even from the same component) can be stitched together depending on the parameter  $r$ , allowing more general and flexible shapes rather than simplex grids only. In this final form,  $r$  indeed regulates the desired minimum intercluster margin. This merging process will also allow the algorithm to recover the number of inherent clusters even in the absence of  $k$ . In other words,  $r$  can be adaptively determined when  $k$  is known;  $k$  can be adaptively discovered when  $r$  is known and fixed.

The algorithm can be traced in mind as follows. Initially there exists a single random simplex. After a few iterations of sparse coding and dictionary update, this simplex will extend through the data and eventually one of its edges surpasses  $r$ . In

this case, the simplex will be subdivided appropriately resulting in new simplexes. Some of these newly introduced simplexes will possibly not attain the required  $A$ , thus will be pruned. Pruning may result in more than necessary number of prototypes, where merging depending on  $r$  will then take place between close enough prototypes. In the end, a low dimensional skeleton-like structure will be learned as these steps are iterated. At this point, it is crucial to observe that the pruning procedure removes hyperedges from  $H_K$ , which may cause  $H_K$  to become disconnected. This actually corresponds to the block sparsity constraint in the system, and also equivalently to the emergence of more than one prototype.

The main objective of the algorithm is to determine  $k$  best-fit prototypes with the given set of constraints. However, upon convergence there may be  $C$  prototypes, or equivalently  $C$  number of connected components in the hypergraph  $S$ . If  $C \wedge k$ , the parameters  $r$  and  $A$  have to be readjusted so that the system is forced to converge with  $k$  prototypes. In summary, it is possible to evolve the initial random simplex to a proper structure as described above. This scheme corresponds to an adaptive structured sparsity approach, where the dictionary  $A$  may be growing or shrinking. The pseudocode of the overall algorithm and its block-diagram as well as the details of subdivision, pruning and merging methods are available in Appendix B.

### 3.3.2 Complexity analysis

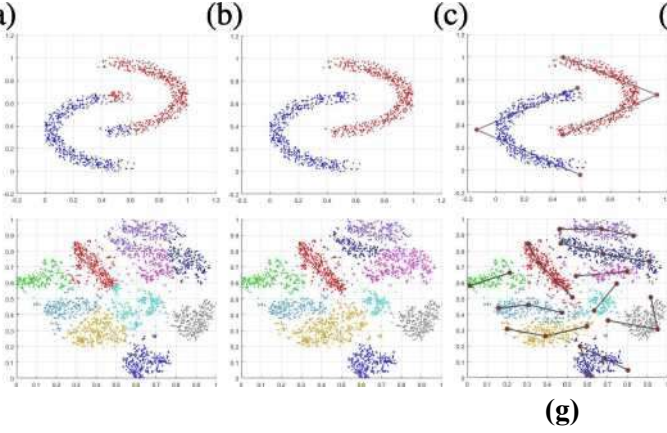
At first, let us consider the complexity of a single iteration. The total number of simplexes is the number of hyperedges in  $S$ , denoted by  $|E|$ . As noted above there is a bound as  $|E| < m$ , and this leads to a total time complexity of  $O(m^2n)$  for sparse coding. To update prototypes, i.e., the matrix  $A$ , least-squares optimization has a time complexity of  $O(m^2)$  where  $p_K$  is the number of vertices in  $K^{th}$  polytope. The sum  $c$  is simply equal to the total number of columns in  $A$  and  $n < p_K < m$ . Hence, updating prototypes has  $O(m^3)$  complexity.

In the subdivision process, available edges of all simplexes must be iterated. In a  $(q-1)$ -dimensional simplex, there are  $q(q+1)/2$  edges. Therefore, time complexity



of detecting oversized edges is  $O(mq^2)$ . A naive subdivision process can introduce at most  $2^{q(q+1)/2}$  new simplexes depending on  $r$ ; however it is assured that  $2^{q(q+1)/2} < m$  from the earlier bound on  $|E|$ . An exceedance of this bound signals to a necessity of updating  $r$ . Therefore, total time complexity for subdivision is  $O(m^2)$ . A similar analysis for pruning yields  $O(m)$  complexity, since there is no new simplexes introduced but only eliminated. Perhaps conceptually the most challenging procedure is merging.

Two simplexes having all their vertices distant from each other do not mean that they are not in proximity. Therefore, the distance between each pair of simplexes has to be calculated.



Surprisingly, there exists an iterative method to compute the distance between two arbitrary convex sets in linear time through Gilbert-Johnson-Keerthi algorithm (Lindemann, 2009). Then, the merging process has  $O(m^2n)$  time complexity same as the projection phase.

To sum up here, the bottleneck of the proposed algorithm is the prototype update phase with  $O(m^3)$  time complexity, which is also the bottleneck for conventional dictionary learning methods. Note that worst-case time complexity of fc-simplexes and /c-polytopes are equal, but /c-polytopes is slower within a constant factor because of additional procedures.

### 3.4 Experimental Results

A crucial point is that a hypergraph based data structure is needed to maintain  $S$  efficiently. Connected components have to be decided after an instance of convergence, where each component indicates the local structure responsible for its cluster. Due to conceptual difficulty of implementing the proposed system in high dimensional cases, in this study, only 1D and 2D polytopes are tested for 2D and 3D data spaces, respectively.

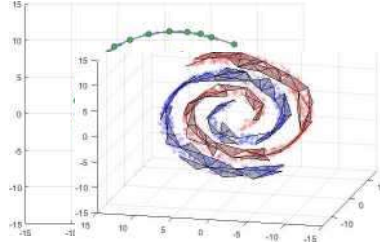
In a set of experiments,  $\ell_1$ -polytopes is compared to  $\ell_1$ -c-means,  $\ell_1$ -flats ( $\ell_1$ -lines), Gaussian Mixture Models (GMMs) (Reynolds, 2015), kernel  $k$ -means (Bishop, 2006) with a Gaussian kernel  $\sigma$  in the range 0.1 to 1.0, and ensemble  $\ell_1$ -c-means (Iam-on and Garrett, 2010) with an ensemble size of 30. Test datasets used are chosen as synthetically generated crescents and spirals, and as real world datasets: IRIS (Dheeru and Taniskidou, 2017) and MNIST (LeCun et al., 2010). A preprocessing is applied

(e) (f)

Figure 8. Example convergence states for (top) Crescents and (bottom) MNIST. (a) Ground-truth clusters, and clustering results with (b)  $k$ -means, (c)  $\ell_1$ -flats, (d) GMMs, (e) kernel  $\ell_1$ -means, (f) ensemble  $\ell_1$ -means, (g)  $\ell_1$ -polytopes. (Best visualized in color.)

to IRIS and MNIST with t-SNE (van der Maaten and Hinton, 2008) in order to reduce dimensionality. All tests are carried out on an Intel(R) Core(TM) i5-6600 CPU @ 3.30GHz Quad-core 8GB RAM machine using Matlab R2017a. Table 2 summaries resulting average and maximum clustering success rates together with average running times for each dataset over 100 randomly initialized runs. Figure 8 further depicts example convergence states for crescents and MNIST. It is obvious from these results that the proposed structure (being bounded and piecewise linear) is a powerful tool for such difficult cases in a clustering application.

Table 2 provides certain striking observations. First of all,  $k$ -flats is apparently not a method for generalized clustering as its prototypes are unbounded. GMMs provide better maximum accuracy compared to both  $k$ -means and  $k$ -flats; however, they



**Figure 9. if-polytopes aware of inter-cluster margins is capable of recovering exact number of clusters. Experimental results in (left) 2D and (right) 3D. (Best visualized in color.)**

are susceptible to random initializations as the mean accuracy suggests. Kernel  $k$ -means seems as a better candidate for generalized clustering but performs still poorly in cases of spirals and MNIST, possibly due to the kernel function chosen. Lastly, ensemble  $/c$ -means appears to be the most generalized method available in the literature. The proposed  $fc$ -polytopes is able to surpass ensemble  $fc$ -means in all test cases except the average accuracy in MNIST. This is possibly due to the fact that, the method becomes susceptible to random initializations when the cluster count gets higher.

In a second set of experiments, the ability of  $/c$ -polytopes to discover inherent number of clusters is analyzed. If the minimum intercluster margin is known, corresponding to desired  $r$  in the formulation, then the exact number of clusters can be recovered. Experimental results of some nontrivial clustering problems are presented in Fig. 9 for both 1D and 2D prototypes in 2D and 3D data spaces, respectively. It is apparent that the proposed method is able to recover the number of clusters successfully, even without being supplied with a parameter  $k$  but only with  $r$ . Hence, the proposed structurally adaptive approach results in a more expressive representation.

### 3.5 Discussion

In this chapter, novel problem formulations for  $/c$ -simplexes and  $/c$ -polytopes

are discussed and solutions to these optimization problems are proposed within the sparse representations framework. These formulations and solutions indeed result in a

more intuitive and geometric understanding of sparse representations, which may open doors to many studies that draw parallels between sparse representations and various other machine learning methods.

It is possible to further generalize the proposed formulation through an ensemble approach, which might possibly give even better results. Similarly, a generalization through kernels would greatly reduce the required number of vertices but further increase the complexity.

In an advantage, one can speculate that the proposed approach will mostly be immune to the curse of dimensionality. Note also that prototypes attained by this proposed method can be any nonconvex geometric objects. Hence, linearly nonseparable clustering problems are not an issue for the proposed framework.

A drawback of current formulation of fc-polytopes arises when the clusters are not homogeneously  $g$ -dimensional, but some parts of clusters are in lower dimensions, or even some parts may have dimensions strictly higher than  $q$  in nature. Such a problem needs to be addressed by allowing the polytopes to have simplexes that can adaptively change their dimensions depending on the nature of data assigned to these prototypes.

Finally, a surprising aspect of the proposed framework emerges when it is considered in a supervised setting. In general supervised settings, classification methods try to learn decision boundaries between more than one classes. However, the proposed method here is capable of learning one class at a time. In other words, for each class, a separate model can be learned without the need for opposing classes. Therefore, no learning is required from scratch when a new class is introduced, since learning a new model for the newly arrived class will suffice. These last two discussion items lead to the next chapter where /c-polytopes concept is further generalized through the introduction of simplicial learning as a one-class learning method.

## **CHAPTER 4: EVOLUTIONARY SIMPLICIAL LEARNING**

### ***4.1 Introduction***

At this point, it is proper to introduce one-class classification, as the fundamental form of the general classification problem, to bridge the gap between reconstructive signal processing and machine learning. Supervised machine learning in the form of classification inherently suggests the existence of more than one label. The concept of one-class learning, also known as one-class or unitary classification, emerges when there only exists a single label within the dataset, and one needs to discriminate it against all possible unseen labels (Moya and Hush, 1996). It is actually a special case of binary classification where there is the “in-class” label and also the “out-of-class”, but there is not any or enough number of “out-of-class” samples within the training dataset. Therefore, in the absence or weakness of the opposing class samples, conventional binary classification methods will have difficulties as they target the decision boundary in-between.

One-class learning methods can be categorized by the type of the targeted classifier model. There exist decision-boundary approaches which seek enclosing hyperspheres, hyperplanes or hypersurfaces in general (Khan and Madden, 2014). These methods can adjust the level-of-detail through the usage of parametrized kernels to cope with the over- or under-fitting problem. On the other hand, graph-based methods try to fit a skeleton with-in data in a bottom-up manner. As an example, a minimum spanning tree model can be utilized as a one-class classifier (Juszczak et al., 2009), in which the classification procedure relies on the distance to the tree. A generalization of graph-based approaches is attained through the concept of hypergraph, in which a hyperedge can now connect more than two data points or vertices. Hypergraph models not only allow custom but also lead the way to heterogeneous dimensionality. Such models are investigated in (Wei et al., 2003; Silva and Willett, 2008). As detailed in Sec. 4.2, simplicial learning through an extension of dictionary learning can be thought as the utmost generalization of the graph-based domain, in which vertices of a hypergraph can now move freely in space, taking the form of a simplicial. Note that in the present formulation, the targeted model is not necessarily a simplicial complex which is a much stricter construct that prohibits self-intersections (Barbarossa et al., 2018). The term simplicial refers here to an arbitrary union of simplices in the most general sense.

Most importantly, the dictionary learning concept can be categorized as an inner-skeleton method. However, the skeleton attained is not bounded in space but rather an infinite one, where each infinite linear bone is connected to all others at the origin. Technically speaking, a bone corresponds to a linear subspace of arbitrary dimensions. This conception will be indeed helpful when dictionary learning is considered within a multi-class classification framework. In its traditional multi-class formulations, the sparse representations based classifier (SRC) models a separate dictionary for each distinct class through a data fidelity term together with an  $\ell_p$ -norm regularization constraint on sparse codes ( $p = 0$  or  $1$  in general). Later, the test data is encoded sparsely and classified accordingly favoring the most reconstructive or representative dictionary (Wei et al., 2013). In the absence of other modifications, this form of SRC is known to be generative-only.

In a simplistic manner, one can draw parallels between inner-skeleton and generative formulations which discard the existence of other classes; on the other hand, also between decision-boundary and discriminative approaches which need the existence of opposing classes. Not surprisingly, a method can be both generative and discriminative at the same time. Discrimination, in this sense, rises from the fact that while learning a dictionary (or a model) for a class, the data points from other classes are also taken into consideration, i.e., distance to those other points are to be maximized. Some examples of discriminative dictionary learning methods can be given as (Mairal et al., 2009; Jiang et al., 2013).

There is a subtle but crucial point that goes unnoticed in SRC applications and this forms the backbone of this chapter. Corresponding to this upcoming point, XOR problem of neural networks dictates that a single layer perceptron is not capable of separating XOR inputs as only a single linear decision boundary is at hand. This has paved way to multilayer formulations that can solve linearly non-separable cases. As already



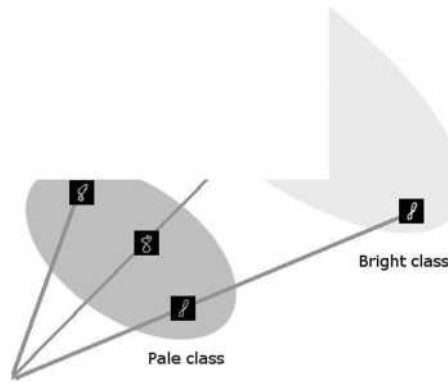


Figure 10. Conventional dictionary learning is incapable of distinguishing intensity/magnitude, or more technically two classes within the same subspace.

noted in Sec. 2.5, a similar problem haunts conventional dictionary learning methods silently. It is now time to define that problem in a more technical way. Consider the case as demonstrated in Fig. 10, in which there are two classes of digit 8. “Pale class” includes pale images, while “Bright class” contains exactly the same images but they are brightened up. In technical terms, there are two opposing classes lying on the same subspace in the eyes of linear dictionary learning methods. No matter how much discriminative they are, traditional techniques will be incapable of totally distinguishing these two classes. In other words, dictionary learning in its conventional form is insensitive to intensity/magnitude and it will never be able to solve problems requiring intensity/magnitude distinction.

#### 4.2.1 Definitions

Conventional dictionary learning basically tries to fit a union of subspaces to the data. Such subspaces are indeed infinite-extent and all crossing the origin without offsets, designated by the dictionary elements usually referred to as *atoms*. In Chapter 3,  $k$  many polytopes are fit to the data. Simplicial learning as an adaptation of both dictionary learning and  $k$ -polytopes concept aims instead at fitting more general bounded generic piecewise linear objects to the data.

Table 3 considers certain bounded generic piecewise linear objects. There are many not-equivalent formal definitions of the first construct, namely a *polytope* as discussed before. This study strictly sticks with the definition that “a polytope is an

intact object which admits a simplicial decomposition.” Hence, a polytope is made up of one or more simplices, whereas it is still in question that such simplices can be of different dimensions.

There are two possible ways to generalize the concept of polytope. In the first generalization, connectedness can be discarded leading to the fact that there is not a single object but multiple objects being considered at the same time. The second one allows the building-blocks namely simplices to have different dimensions, thus leading to heterogeneously dimensional objects. A formal name for such union of simplices is a *simplicial complex*, but restricted self-intersections are imposed for a rigorous treatment. By definition, a simplicial complex is a set of simplices satisfying the following two conditions: (i) every face of a simplex from this set is also in this set and (ii) the non-empty intersection of any two simplices is a face of these two simplices. Losing a bit of formalism, utmost flexibility can be reached by allowing such objects to intersect each other and themselves in arbitrary ways, and such final construct is simply named as a *simplicial* in the remaining part of this thesis, to refer to an arbitrary union of simplices in the most general sense. For a more rigorous treatment of these definitions and related concepts, readers might refer to the subject of algebraic topology.

#### 4.2.2 *Related work*

Simplex and simplicial complex based data applications are becoming popular in literature as data analysis receives more and more topological considerations (Luo et al., 2017; Huang et al., 2015; Belton et al., 2018; Tasaki et al., 2016; Patania et al., 2017). Moreover, utilizing simplices for data applications is not a completely new idea from the perspective of sparse representations (Wang et al., 2016; Nguyen et al., 2013). Quite similarly, in this chapter an adaptation of sparse representations framework is chosen that casts a union of subspaces to a union of simplices. A rigorous mathematical formulation is detailed in the following.

### 4.2.3 Mathematical formulation

There are three necessary modifications to make a successful transition from the traditional dictionary learning formulation to simplicial learning. As in  $k$ -polytopes, an additional sum-to-one constraint is needed on the sparse codes. In addition, the second necessary modification is an additional non-negativity on sparse codes again as in the case for  $/c$ -polytopes. This time, last modification on the road to simplicial learning is group sparsity (Yuan and Lin, 2006; Jacob et al., 2009) instead of block-sparsity. The possibility of this variation is already mentioned while formulating the concept of  $/c$ -polytopes as in Sect. 3.3.

While referring back to Sec. 4.1, when positional information is removed from a simplicial, the structure left then corresponds to a hypergraph, in which a hyperedge refers to a specific simplex within the simplicial. In relation to group sparsity, a hyperedge exactly corresponds to a group of atoms, hence a valid pattern of sparse codes.

**Table 3. Distinctions between the terms for generic objects.**

	May not be intact	Piecewise linear	Heterogeneous dimensionality	Arbitrary intersections
Polytope	X	/	?	✓
Simplicial complex	✓	✓	/	X
Simplicial	✓	✓	/	✓

As a consequence, a set of groups/hyperedges, or more technically a hypergraph data structure needs to be kept to define the shape of the simplicial parallel to  $/c$ -polytopes formulation. This hypergraph structure will be denoted as  $V = \{h_j\}$  where  $h_j$  designates the  $j^{th}$  hyperedge referring to  $j^{th}$  simplex within the simplicial. In accordance with this definition, simplicial learning with a structure imposed by  $V$  can be formulated in Eqn. (14) as follows,

$$\begin{aligned} & \underset{\mathbf{A}, \{\mathbf{x}_i\}, \{M_i\}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_1 \text{ subject to} \\ & \|\mathbf{x}_j\|_0 \leq q, \mathbf{A}^T \mathbf{X}_j = \mathbf{1}, \mathbf{0} < \mathbf{X}_j \leq \mathbf{A} \{k \neq h^* \rightarrow \mathbf{x}_k = 0, \forall A_j\}, \forall i, \end{aligned} \tag{14}$$

where  $h^*$  is the hyperedge indexing the closest simplex for the data point  $\mathbf{y}$  is  $q^* = |h^*|$

denotes the dimension of that simplex, and the  $(\|x\|_1 \leq 1, x \geq 0)$  constraint ensures the group sparsity such that only the optimal group (i.e., hyperedge referring to the closest simplex) in  $X_j$  is to be filled and other entries which are represented as  $x^*$  shall all be zero. Note here that groups can be not only overlapping but also of different sizes, hence leading to heterogeneous dimensionality. In this final form,  $T_i$  needs to be learned together with  $A$  but a further careful consideration is needed over the compactness of the simplicial in return.

In summary, as is, the optimization in Eqn. (14) is highly ill-posed since there is no restriction on the number of simplices to be used or the dimensions of those simplices. One could even choose a very high dimensional simplicial construct and zero-out the approximation error easily. Therefore, additional penalty terms need to be investigated based on the number and the dimensionality of simplices for a compact solution. Such a challenge appears to be highly combinatorial in nature and an evolutionary approach can be adopted after a careful consideration of an appropriate fitness function, as described and detailed in Sec. 4.3.

### 4.3 Evolutionary Approach

To obtain an optimal or a suitable simplicial in a heuristic manner, certain number of simplicials are to compete against each other on instances of the same dataset. Basically, an evolutionary approach includes a suitable fitness function to guide this search process, and sub-procedures such as *mutations* and *breeding* to perform the actual search.

#### 4.3.1 The fitness function

There are certain critical points to be carefully considered before designating the fitness function for the defined problem in this study. First of all, a straightforward optimization procedure for the number and the dimensionality of simplices will not be enough to attain a compact model desired. For example, consider that the data is distributed in the shape of a triangle with certain area. In this case, a triangle with the most compact area should be preferred as a targeted

model. However, one could fit a triangle to this data with correct angles but excessive area. In such a case the dimensionality or the number of simplices indeed do not change. In conclusion, one needs also to take the area (or volume), or more technically the content of the simplices, besides considering the number and the dimensionality of simplices. When the simplex is of dimension 2 (namely a triangle), the content is called the area, when the simplex is 3 dimensional (namely a tetrahedron), the content refers to the volume. Therefore, the term “content” generalizes area and volume concepts to higher dimensions.

The content of an arbitrary simplex can be calculated using Cayley-Menger determinant (Li et al., 2015). Let  $K$  be a  $g$ -dimensional simplex in  $R^N$ , and  $B$  denote  $(q+1) \times (q+1)$  distance matrix of vertices  $\{v_1, v_2, \dots, v_{q+1}\}$  such that  $B_{ij} = \|v_i - v_j\|_2$ . Then the content  $C_K$  of  $K$  is given in a relation in Eqn. (15) as follows,

$$(15)$$

where  $B$  is  $(q+2) \times (q+2)$  matrix obtained from  $B$  by bordering it with a top row of  $(0, 1, \dots, 1)$  and a left column of  $(0, 1, \dots, 1)^T$ .

Related to the content calculation here, another issue arises because of the allowed heterogeneous dimensionality in the optimization formula. The content of a line-segment (as an object) and a triangle (as an object) are incomparable in a general continuous setting since a triangle contains infinitely-many line-segments itself. To resolve this problem, an exponential term is introduced through *an approximated*

*cumulative discrete content* calculation as given in Eqn. (16) as follows,

$$\sum_{i=1}^{|H|} (1 + C_j)^{q_j} \quad (16)$$

where  $|H|$  denotes the number of hyperedges or equivalently the number of simplices,

$C_j$  is the content of the  $j^{th}$  simplex and  $q_j$  is the dimension of that simplex. As a content  $C_j < 1$  would complicate the exponentiation used,  $(1 + C_j)$  is needed in the discrete approximation.

Having pinned down with the above term which will be a component in the fitness function driving the evolutionary process, a fitness function candidate (in a minimization form) is given in Eqn. (17) as follows,

$$\sum \|y_i - \mathbf{A}x_i\|_2^2 + \alpha \sum (1 + C_j)^{q_j}, \quad (17)$$

where sum of squared error (SSE) used as the data fidelity term and approximated cumulative discrete content as to regulate the compactness of the representation,  $\alpha$  denotes the regularization parameter controlling the contribution of the compactness prior on the solution.

While initially experimenting above fitness function, it is observed that the parameter  $\alpha$  has a very broad optimality range, which changes drastically from dataset to dataset. This is due to the fact that there is a high dynamic range imbalance between two cumulative terms. Therefore, a variant of the defined fitness function is considered by transforming Eqn. (17) into the logarithmic scale in order to compress the dynamic range, leading to a more natural maximization setting formulated in Eqn. (18) as follows,

$$(18) \frac{\log_{10} \left( \frac{n}{\sum_i \|y_i - \mathbf{A}x_i\|_2^2} \right)}{1 + \beta \log_{10} \left( \gamma + \sum_j (1 + C_j)^{q_j} \right)},$$

where  $n$  denotes the number of data points and the parameter  $\beta$  regulates over- or under-fitting. When  $\beta = 0$ , the fitness function simply reduces to the data fidelity term favoring only for the reconstruction quality. Instead, a high  $\beta$  value forces the

simplicial to be compact. Empirical investigations suggest that a  $\beta$  value around 0.05

could be a global setting as it provides excellent results over all datasets considered in this study. Note that there might be no simplices at certain times of the evolution process. This would erroneously lead the sum of content to be zero, thus logarithm to be infinity. The parameter  $\gamma$  eliminates this possibility by fixing its value to 10. Hence, this parameter forces the lower logarithm to evaluate at least to value 1.

#### 4.3.2 Mutations and breeding

First of all, it is important to note here that the hypergraph  $H$  is kept in the form of an incidence matrix of zeros and ones, where the row count corresponds to the number of simplices and the column count matches to the number of vertices or rather the number of atoms (columns) in the dictionary  $A$ . Mutations can be easily applied on this binary matrix. In detail, there are four main processes that provide the background for evolution: (i) increasing/decreasing the dimension of a simplex, (ii) adding/removing a simplex to/from the hypergraph, (iii) subdividing a simplex and (iv) adding/removing a vertex to/from the dictionary. All of these mutation operations are performed randomly without any optimality consideration.

As an additional tool to assist the searching process, breeding of two simplicials is also undertaken in which both dictionary elements and hypergraph structures of those two simplicials are split and then merged appropriately in order to create a new simplicial representative of two parents up to certain extent. Details of the breeding procedure are depicted in Alg. 1. At first, hypergraph structures and the corresponding dictionary elements are extracted for these two simplicials  $S_1 = (H_1, A_1)$  and  $S_2 = (H_2, A_2)$ . Then random submatrices  $H_a \in H_1$  and  $H_b \in H_2$  from each hypergraph are attained together with the corresponding columns of these dictionaries, contained in matrices  $A_a \in A_1$  and  $A_b \in A_2$ . While vertices (atoms) are directly concatenated in  $A_{new}$  (line 7), hypergraphs are concatenated in a disjoint manner in  $H_{mw}$  (line 8). In short, two subsimplicials are extracted and then grouped together in a disjoint manner to form a new simplicial  $S_{new}$ . Such tool can be suitably employed to exploit the underlying dimensionality of the dataset since these splitting and merging processes may lead child simplicials to acquire a properly representative data- dimensionality in a very fast manner, much faster than mutation



**processes to perform**

#### Algorithm 1 Breeding Algorithm

```

1:  $(H_i, A_i)$   $\leftarrow$  get_structure( $S_i$ )
2:  $(H_2, A_2)$   $\leftarrow$  get_structure( $S_2$ )
3:  $H_a \leftarrow$  a random submatrix of  $H_i$ 
4:  $A_a \leftarrow$  the submatrix of  $A_i$  corresponding to  $H_a$ 
5:  $H_b \leftarrow$  a random submatrix of  $H_2$ 
6:  $A_b \leftarrow$  the submatrix of  $A_2$  corresponding to  $H_b$ 
7:  $A_r \leftarrow [A_a \ A_b]$ 
8:  $H_{new} \leftarrow H_a \cup H_b$ 
9:  $A_{new} \leftarrow A_r$ 

```

alone. Therefore, as a general observation, breeding determines the core dimensionality of the simplicial and mutations fine-tune the simplicial to the data. However, sufficiently high dimensional simplicials should be employed in the initialization stage for breeding to determine the core dimensionality.

#### 4.3.3 Implementation details

The algorithm to learn an evolutionary simplicial model on a set of data points  $\{y_i\}_{i=1}^n$  stored in the columns of a data matrix  $Y$  is given in Alg. 2. At first, the initial simplicial is to be generated from the given data points (line 1). It is observed that choosing a single point (i.e., centroid of the dataset) as an initial simplicial is sufficient for low-dimensional problems. Through mutations and breeding processes, the initial simplicial takes an appropriate form in a fast manner since the search space is relatively small. However, a procedure involving the fc-means algorithm (Jain, 2010) as a subroutine is employed to designate the initial simplicial for high dimensional problems. In such cases, starting from a single point greatly slows down the process of evolution since the search space is quite large. Hence, an initialization based on fc-means ensures that the starting simplicial is already a relatively fit one. A last point worth mentioning related to initialization here is that the initial simplicial  $S$  should satisfy the condition that the numerator of Eqn. (18) is positive, i.e.,  $\|y^* - Ax\|_1 < n$  to lead a meaningful evolution.

On line 6, the algorithm performs the projection of data points  $\{y^*\}$  in  $Y$  onto

**Algorithm 2 Evolutionary Simplicial Learning (ESL) Algorithm**

```

1: pop  $\leftarrow$  init_pop( $Y$ )
2: while not converged do 3:
    pop  $\leftarrow$  mutations (pop)
4:   pop  $\leftarrow$  breeding (pop)
5:   for all  $S$  in pop do
6:      $X$   $\leftarrow$  sparse_coding( $Y, S$ )
7:      $A$   $\leftarrow$  dictionary_update( $Y, X$ )
8:      $F$   $\leftarrow$  fitness( $A, H$ )
9:   pop  $\leftarrow$  sort and choose based on  $F$  values
10:  $S_{best}$   $\leftarrow$  pop(1)

```

each simplex of the simplicial  $S$  (Duchi et al., 2008; Golubitsky et al., 2012) which basically corresponds to the sparse coding optimization. The closest simplex for the data point  $y_i$ ,  $V_i$ , is determined through the minimum approximation error acquired after projecting  $y^*$  onto each simplex. The positive barycentric coordinates of the projection points corresponding to the sparse codes are acquired, and then the necessary spots of the sparse representation matrix  $X$  is filled accordingly.

On line 7, dictionary matrix  $A$  is updated using a direct least-squares solution. To optimize  $\arg\min_A \|Y - AX\|_f$ , by forcing its derivative to zero, the analytic solution is obtained with  $A = YX^+$  where  $X^+$  represents Moore-Penrose pseudo-inverse of  $X$ . Note that there is no evolutionary process for learning  $A$ , namely the vertices of the simplicial  $S$ . Instead, vertices are updated once exactly on this line at each iteration of the algorithm.

Finally, the surviving simplicials are determined based on the fitness scores they attain (line 10). Experimental trials suggest that keeping the population size at 10 is an efficient strategy, while an iteration count of 5 is sufficient instead of a full convergence. Notice here that the parent simplicials are to be kept in the population pool when their fitness scores are higher than their children's.

**4.4 Experimental results**

The proposed method is tested in two phases of experiments to evaluate its classification capabilities. In the first experimental setup, the performance is evaluated in a one-class classification task for outlier detection. Datasets contain certain degree of outliers in such outlier detection problems, and methods learn models -agnostic of data labels- in an unsupervised manner. In the second

classification task, the performance of the proposed method is evaluated in a multi-class setting. At this stage, seven synthetic multi-class datasets are generated in addition to two handwritten digit recognition datasets. The synthetic datasets are special in that they contain cases which require intensity/magnitude distinction, especially very challenging for conventional dictionary learning methods.

All experiments are performed on an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz 16GB RAM machine running on Microsoft Windows 10. Benchmark of outlier detection dataset named PyOD (Zhao et al., 2019) is run with Python 3.6 and the proposed ESL algorithm is implemented using Matlab 2014a on the same machine. All multi-class experiments are carried out on Matlab 2014a. DICTOL as the part of LRSDL project (Vu and Monga, 2017) is utilized for the implementations of other dictionary learning methods.

#### *4.4.1 Outlier detection*

In total 17 benchmark datasets are taken from ODDS Library (Rayana, 2016) for the one-class learning task. Information regarding these datasets in terms of number of samples, sample dimensionality and outlier percentages is summarized in Table 4 and interested readers might refer to (Rayana, 2016) for details about each individual dataset. Using these benchmark datasets, a random 60% to 40% train-test set split is repeated for 10 independent simulations and the mean Area Under The Curve (AUC) Receiver Operating Characteristics (ROC) results are reported in Table 5.

The proposed Evolutionary Simplicial Learning (ESL) method is evaluated against an extensive outlier detection benchmark named as PyOD (Zhao et al., 2019). The competing methods include Angle-based Outlier Detector (ABOD) (Kriegel

**Table 4. Information regarding the datasets used in outlier detection experiments.**

Dataset	#Samples	#Dimensions	Outlier Ratio (%)
arrhythmia	452	274	14.6018
cardio	1831	21	9.6122
glass	214	9	4.2056
ionosphere	351	33	35.8974
letter	1600	32	6.2500
lympho	148	18	4.0541
mnist	7603	100	9.2069
musk	3062	166	3.1679
optdigits	5216	64	2.8758
pendigits	6870	16	2.2707
pima	768	8	34.8958
satellite	6435	36	31.6395
satimage-2	5803	36	1.2235
shuttle	49097	9	7.1511
vertebral	240	6	12.5000
vowels	1456	12	3.4341
wbc	378	30	5.5556

et al., 2008), Clustering-based Local Outlier Factor (CBLOF) (He et al., 2003), Feature Bagging (FB) (Lazarevic and Kumar, 2005), Histogram-based Outlier Score (HBOS) (Goldstein and Dengel, 2012), Isolation Forest (IForest) (Liu et al., 2008), K Nearest Neighbors (KNN) (Ramaswamy et al., 2000), Local Outlier Factor (LOF) (Breunig et al., 2000), Minimum Covariance Determinant (MCD) (Hardin and Rocke, 2004), One-class Support Vector Machine (OCSVM) (Scholkopf et al., 2001) and Principal Component Analysis (PCA) (Jolliffe, 2002) and one of the most recent results obtained in (Weng et al., 2018) on the same benchmark (with an average of 20 runs for each dataset).

Last two rows of Table 5 illustrate the mean AUC ROC results over all datasets and their standard deviations. ESL not only presents the best average AUC ROC performance among all methods in the benchmark but also has the least standard deviation. One can conclude that it is the most reliable method among considered techniques for this performance measure. Moreover, ESL shows top AUC ROC performance in

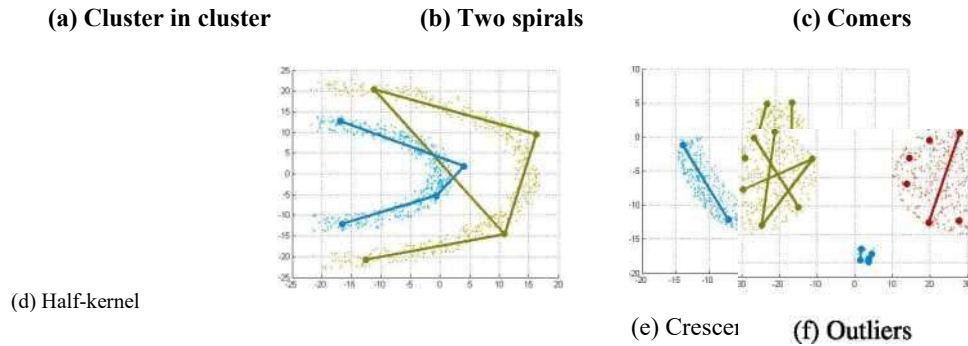


Figure 11. Examples of learned simplicials on synthetic datasets

three datasets. However, additional tests show that it does not have a noticeable advantage in Precision at  $n$  ( $P@n$ ) performance.

#### 4.4.2 Multi-class experiments

**Synthetic datasets:** For the multi-class classification task, six challenging synthetic datasets are generated by following the procedures in (MathWorks, 2019) and these datasets are depicted in Fig. 11. Four of these datasets contain binary classification tasks while the remaining two of them (*Comers* and *Outliers*) consist of four-class classification problems. In addition, a synthetically altered dataset (named as  $MNIST_8$ ) is included in the experimental setup, in which all samples of the digit 8 from the original MNIST (LeCun et al., 2010) are designated as the “Bright class” while a new “Pale class” is generated from all these original samples by dimming with a scale of 0.25 according to the previous discussion related to Fig. 10.

The proposed ESL algorithm in this setup is compared against Sparse Representation based Classification (SRC) (Wright et al., 2008), Label Consistent K-SVD (LCKSVD1 and LCKSVD2) (Jiang et al., 2013), Dictionary Learning with Structured

Incoherence (DLSI) (Ramirez et al., 2010), Fisher Discrimination Dictionary Learning (FDDL) (Yang et al., 2011), Dictionary Learning for Commonality and Particularity (DLCOPAR) (Kong and Wang, 2012) and Low-rank Shared Dictionary Learning (LRSDL) (Vu and Monga, 2016, 2017). Experimental results in terms of classification success rates are presented in Table 4.4.2. It is apparent that ESL easily outperforms all considered dictionary learning methods over all cases. This should not be a surprising result since all utilized synthetic datasets require intensity/magnitude distinction to various extents. On the other hand, some discriminative methods such as LCKSVD2, FDDL and LRSDL undergo meaningful learning (i.e., better than random) over some datasets. This observation leads to an important conclusion that discriminative modifications may alleviate insensitivity to intensity to a certain degree.

Fig. 11 depicts examples of learned simplicial models on six synthetic datasets. As it can be observed clearly, simplicials are bounded and they are composed of simplices (i.e., points and line-segments in these cases) with arbitrary offsets, providing an advantage over unbounded and without-offset dictionary learning models in all these classification tasks.

*Digit classification:* In most of the practical pattern recognition applications, the pattern or rather the direction of the feature vector utilized plays an important role on the success rate. For instance, a “star pattern” is a “star pattern” no matter how much bright or pale it is. Therefore, the advantage of simplicial learning over dictionary learning is expected to diminish in some real-world applications. This is observable in digit classification experiments featuring USPS (Hull, 1994) and MNIST datasets as reported in Table 7. In this set of experiments, ESL is compared to classification methods including Supervised Dictionary Learning (Mairal et al., 2009) with generative training (SDL-G) and with discriminative learning (SDL-D), Task-driven Dictionary Learning (Mairal et al., 2011): unsupervised (TDDL-G) and supervised (TDDL-D), FDDL, KNN, Gaussian SVM, Locality-constrained Linear Coding (LLC) (Wang et al., 2010) and Locality-sensitive Dictionary Learning (LDL) (Wei et al., 2013). LLC and LDL methods have the sum-to-one constraint on sparse codes, therefore they learn spaces with arbitrary offsets but learned models are still

not bounded (without the nonnegativity constraint).

As apparent from Table 7, ESL appears to be a successful generative-only method which performs nearly at the capacity of Gaussian SVM (i.e., a well-known and widely used discriminative classifier). However, it cannot outperform discriminative dictionary learning methods such as FDDL and TDDL-D in these datasets. A final note is that ESL can also be modified through discriminative elements. Discriminative methods SDL-D and TDDL-D have a 1.5 - 2% advantage over their generative counterparts SDL-G and TDDL-G. Hence, a successful discriminative version of ESL can then be projected to reach state-of-the-art, an estimation open to discussion or further investigation.

#### 4.5 Computational Complexity

Lets first dissect the loop starting on line 5 and ending on line 9 in Alg. 2 since this part mainly determines the time complexity. On line 6, each data point  $y^*$  is projected onto each simplex within the simplicial  $S$  and then assigned to the closest one. There are  $|H|$  simplices within a simplicial, namely the number of hyperedges in the corresponding hypergraph. An efficient projection onto a single simplex is claimed to have a time complexity of  $\theta(n)$ ,  $n$  is the dimension of data space (Condat, 2016). In a sensible model, there must be at most one simplex for each data point, resulting in a bound  $|H| < m$  and  $m$  is the number of data points. Therefore, complexity of the sparse coding phase is  $\theta(m^2n)$ . On line 7, dictionary update is performed by Moore- Penrose pseudo-inverse, having a time complexity of  $\theta(m^2v)$  where  $v$  denotes the total number of columns (atoms) in the dictionary  $A$ . Since overcompleteness implies  $n < v < m$ , this phase arrives at a complexity of  $\theta(m^3)$ . Lastly, line 8 includes the content calculation for each simplex. Since it involves calculating the determinant of a  $(q+2) \times (q+2)$  matrix and  $q$  is the dimension of the simplex, the complexity can be given as  $\theta(J^2j Qj)$  assuming that LU decomposition is employed for the determinant. An important remark here is that  $q^3$  is negligible compared to  $m^3$  and dictionary update is still the most expensive step within the loop. However, having very high dimensional simplices will slow down the algorithm. Note also that sorting, applying mutations and breeding are not computationally



expensive when compared against operations within the loop.

Lets now discuss the sensitivity of the algorithm to the ratio between actual dimension of the ambient space and actual inner size of the data. As noted before, mutations alone increase or decrease the dimensions of the model in a relatively slow manner. This is the main reason of breeding which may speed up the algorithm by creating children that are much less dimensional than their parents. Assuming that the starting simplicials have high enough dimensions, the breeding process uncovers the core dimensionality and then mutations will uncover local varieties. In short, it would take a long amount of time to recover the actual inner size of the data if only mutations were being used, but the algorithm can cope with this issue via the breeding process in a more effective way.

As a final note, the complexity of the proposed evolutionary approach is highly related to the population size. Therefore, the population size can be adjusted accordingly to satisfy the computational requirements versus the performance criteria. Moreover, the implemented Madab code in this study is experimental, hence even larger population sizes can be manageable with more optimized implementations.

#### *4.6 Discussion*

Dictionary learning through simplicials is more flexible than classical dictionary learning models since simplices are bounded and freely positioned in space. The proposed sparsity based evolutionary structure, called ESL is highly applicable if the characteristics of the problem at hand requires such successful localized models. In this study, a global fitness function is employed and there is no restriction on the local fitness of each individual simplex within the simplicial. If the local fitness of each simplex is considered and optimized individually, the resulting simplicial model might be in a more compact form. For example, the unnecessary simplex of the green simplicial in Fig. 11(c) would most probably be eliminated as it does not have any local fitness, thus lead to an increased accuracy of classification. Another point worth mentioning here is that the employed fitness function in Eqn. (18) is reminiscent of

Poisson distribution, in a multidimensional form (Inouye et al., 2017; Belyaev and Lumen'skii, 1988). Hence, other probabilistic considerations and also discriminative elements can be adapted to strengthen both theoretical and application aspects of the proposed framework.

As exemplified in this paper, simplicial learning can successfully address some weak points of conventional dictionary learning for the considered machine learning problems; it is a promising approach inherently capable of performing signal processing tasks and can become a general machine learning tool with many application domains.

Table 5. Mean AUC ROC results, and running times.

Dataset	ABOD	CBLOF	FB	HBOS	IForest	KNN	LOF
arrhythmia	0.769 0.31s	0.784 0.34s	0.778 0.68s	0.822 0.29s	0.801 0.49s	0.786 0.11s	0.779 0.09s
cardio	0.569 0.46s	0.928 0.16s	0.587 0.97s	0.835 0.01s	0.921 0.42s	0.724 0.19s	0.574 0.12s
glass	0.795 0.04s	0.850 0.05s	0.873 0.04s	0.739 0.01s	0.757 0.31s	0.851 0.01s	0.864 0.01s
ionosphere	0.925 0.07s	0.813 0.07s	0.873 0.08s	0.561 0.01s	0.850 0.33s	0.927 0.02s	0.875 0.01s
letter	0.878 0.42s	0.507 0.14s	0.866 0.88s	0.593 0.01s	0.642 0.42s	0.877 0.16s	0.859 0.11s
lympho	0.911 0.03s	0.973 0.05s	0.975 0.04s	0.996 0.01s	0.994 0.31s	0.975 0.01s	0.977 0.01s
mnist	0.782 8.61s	0.801 1.50s	0.721 55.79s	0.574 0.08s	0.816 2.24s	0.848 7.79s	0.716 7.43s
musk	0.184 2.61s	0.988 0.52s	0.526 14.49s	1.000 0.08s	1.000 1.52s	0.799 2.05s	0.529 1.93s
optdigits	0.467 2.96s	0.509 0.61s	0.443 14.60s	0.873 0.04s	0.725 1.24s	0.371 2.11s	0.450 1.94s
pendigits	0.688 1.71s	0.949 0.37s	0.460 4.44s	0.924 0.01s	0.944 0.72s	0.749 0.71s	0.470 0.66s
pima	0.679 0.15s	0.735 0.09s	0.624 0.12s	0.700 0.01s	0.681 0.34s	0.708 0.04s	0.627 0.01s
satellite	0.571 2.11s	0.669 0.63s	0.557 8.52s	0.758 0.03s	0.702 1.01s	0.684 1.23s	0.557 1.14s
satimage-2	0.819 1.91s	0.992 0.52s	0.457 6.52s	0.980 0.02s	0.995 0.80s	0.954 0.99s	0.458 0.87s
shuttle	0.623 17.36s	0.627 1.38s	0.472 70.16s	0.986 0.03s	0.997 3.07s	0.654 10.12s	0.526 13.85s
vertebral	0.426 0.05s	0.349 0.06s	0.417 0.04s	0.326 0.01s	0.391 0.30s	0.382 0.01s	0.408 0.01s
vowels	0.961 0.31s	0.586 0.11s	0.943 0.34s	0.673 0.01s	0.759 0.38s	0.968 0.09s	0.941 0.04s
wbc	0.905 0.08s	0.923 0.08s	0.933 0.09s	0.952 0.01s	0.931 0.32s	0.937 0.02s	0.935 0.01s
MEAN	0.703	0.764	0.677	0.782	0.818	0.776	0.679
STDEV	0.210	0.195	0.203	0.192	0.164	0.182	0.199

Table 5. (continued) Mean AUC ROC results, and running times.

Dataset	MCD	OCSVM	PCA	Weng et al. (2018)	ESL
arrhythmia	0.779 3.82s	0.781 0.05s	0.782 0.14s	0.801	0.826 8.51s
cardio	0.814 1.61s	0.935 0.09s	0.950 0.01s	0.969 -	0.884 36.19s
glass	0.790 0.06s	0.632 0.01s	0.675 0.01s	-	0.876 4.70s
ionosphere	0.956 0.35s	0.842 0.01s	0.796 0.01s	0.911	0.851 7.09s
letter	0.807 5.77s	0.612 0.08s	0.528 0.01s	-	0.776 22.63s
lympho	0.900 0.11s	0.976 0.01s	0.985 0.01s	0.987 -	0.984 2.72s
mnist	0.867 14.14s	0.853 4.91s	0.853 0.20s	0.929 -	0.803 171.54s
musk	1.000 57.93s	1.000 1.27s	1.000 0.25s	1.000 -	0.972 77.48s
optdigits	0.398 6.94s	0.500 1.45s	0.509 0.08s	-	0.746 103.72s
pendigits	0.834 4.82s	0.930 0.99s	0.935 0.02s	0.938	0.951 91.98s
pima	0.675 0.09s	0.622 0.01s	0.648 0.01s	-	0.626 13.19s
satellite	0.803 9.13s	0.662 1.41s	0.599 0.04s	0.750 -	0.705 105.03s
satimage-2	0.996 8.94s	0.998 1.14s	0.982 0.04s	0.976	0.995 83.80s
shuttle	0.990 16.11s	0.992 50.88s	0.990 0.05s	0.994 -	0.992 428.03s
vertebral	0.391 0.06s	0.443 0.01s	0.403 0.01s	0.580	0.413 4.96s
vowels	0.808 1.40s	0.780 0.04s	0.603 0.01s	-	0.881 21.31s
wbc	0.921 0.33s	0.932 0.01s	0.916 0.01s	-	0.924 6.03s
MEAN	0.808	0.793	0.774	n/a	0.835
STDEV	0.179	0.183	0.197	n/a	0.152

Table 6. Classification accuracy and computation time for six synthetic datasets, and for the proposed binary MNIST<sub>8</sub> problem.

Dataset SRC LCKSVD1 LCKSVD2 DLSI FDDL DLCOPAR									
Cf i	r- <=> °0 00	</> I-I ^	28 CM CM	0 00	Cf tb CO	28 CO CO 05	CA CC lb T-H	£8 X 05 05	CA lb n kO
iJ O3 Pi iJ	28 ko ko Ti<	CA lb CO CO	28 o lb 05 ko	00 CO	28 CO CO 05	CA 05 05	C C M	£8 CO CO 00	CA CM
28 00 05 lb CO	CA o b- CM	28 o T-H T-H ko	CA 05 T-H	CO	28 CO CO 05	CA T-H C	£8 CM CO	CA O CM	£8 lb CO 05 CM
28 CO ko ko ko	CA CM 00 ko	28 O lb CO ko	CA T-H	CO	28 o o 00 ko	CA 00 05	k	£8 CO	CA lb CM ko
28 00 CO CO Ti<	CA lb CO	28 o CO CM ko	CA lb 05	CO	28 o T-H ko	CA C 05	C	£8 CO CO 05	CA CO CM CO
28 ko ko ko	CA CM d	28 CO T-H lb	CA 05	d	£8 CO CO 05	CA C C d	d	28 CO CO 05	CA kO CM d
28 05 05 d ko	CA CM d	28 o T-H	CA T-H	CO	iS o CO	CA CO CM	d	£8 CO CO 05	CA CO CM d
28 lb CM ko	CA b- CM	28 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA 05 CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM
£8 lb 05	CA CO CM	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM	CO	£8 CO CO 05	CA CO CM

Table 7. Classification error rates of various methods on handwritten digit datasets, USPS and MNIST. ESL appears as a superior generative method, nearly performing at the capacity of discriminative Gaussian SVM on both datasets.

<b>Generative-only</b>					
<b>Dataset</b>	<b>SDL-G</b>	<b>TDDL-G</b>	<b>LLC</b>	<b>LDL</b>	<b>ESL</b>
<b>USPS</b>	<b>6.67</b>	<b>4.58</b>	<b>4.48</b>	<b>3.79</b>	<b>4.31</b>
<b>MNIST</b>	<b>3.56</b>	<b>2.36</b>	<b>-</b>	<b>-</b>	<b>1.85</b>

<b>Discriminative</b>					
<b>Dataset</b>	<b>KNN</b>	<b>SVM-Gauss</b>	<b>SDL-D</b>	<b>FDDL</b>	<b>TDDL-D</b>
<b>USPS</b>	<b>5.2</b>	<b>4.2</b>	<b>3.54</b>	<b>3.69</b>	<b>2.84</b>
<b>MNIST</b>	<b>5.0</b>	<b>1.4</b>	<b>1.05</b>	<b>-</b>	<b>0.54</b>

## CHAPTER 5: THE PROBLEM OF ORTHOGONALITY

### 5.1 Introduction

In traditional signal processing and machine learning problems, each data dimension (attribute) is assumed to be orthogonal to others. In other words, there is no distinction between cross-relations of dimensions. While signals carry information through a spatio-temporal configuration, assuming such orthogonality of signal dimensions is highly ill-posed even for ID cases. This phenomenon is depicted simply in Fig. 12.

Let us numerically analyze the severity of the problem of casting signals as vectors. Assume that an  $n$ -sized vector is received through the orthogonality consideration and it is known that the original form is an  $n$ -sized ID signal. If one tries to recover the original spatial configuration without further knowledge (i.e., which value was in which cell), all  $n!$  possible spatial configurations are equally likely. This problem becomes even more serious when the dimensionality of the signal itself increases. Consider an  $n$ -sized vector is received again but the underlying signal is now assumed to be an image. Not only there are permutations involved but also one needs to guess the height and width of the image. In general, for an  $n$ -sized vector

and a  $n$ -dimensional original signal, the number of possible spatial configurations that the signal could have been in is given as  $d_K(n)n!$  where  $d_K\{n\}$  is the  $n$ -th Piltz function, which gives the number of ordered factorization of  $n$  as a product of  $K$  terms (Sândor, 1996).

When the above described issue is undertaken, it is not hard to see that many conventional machine learning formulations are highly ill-posed from the perspective of real world signals. Let us now consider the case of fc-means to be applied on vectorized real world signals, and suppose images for simplicity. As fc-means originally assumes orthogonality of dimensions, it is easy to apply the usual Euclidean distance metric between vectors. However, it is indeed questionable whether it will capture the notion of distance between two images or rather the average of two images. An example in this light can be given from the domain of Computer Graphics. A direct linear

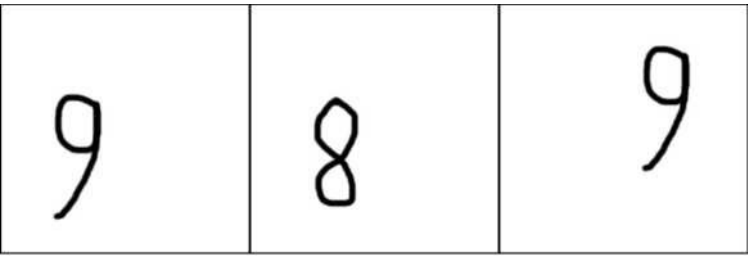
Figure 12. (Left) There is orthogonal<sup>x</sup> consideration. Every pairwise relation between dimensions is indistinguishable because of orthogonality. (Right) While considering spatial configuration of a 3-cell ID signal, the relation between cells  $x$  and  $z$  is obviously different from the other relations, i.e.,  $x$  and  $z$  are not neighbors.

interpolation between two rotation matrices is not natural, thus quaternions are utilized leading to a formulation called spherical linear interpolation (Jafari and Molaei, 2014). A similar consideration might also be superior in the clustering problem of images using fc-means. However, it is not trivial to cast a general image as a quaternion-like structure for further processing.

Let us try to prove that direct vectorized distance calculation is not natural for images by giving a more concrete example. Assume that there is a main image of the number 9 as exemplified in Fig. 13(a). The question here is which other image is more similar to this main image. Is it the number 8 in Fig. 13(b) having relatively same spatial position within the frame, or is it the number 9 in Fig. 13(c) with exact shape but linearly shifted in the frame? Vectorized distance measure will dictate that 8 is closer to the main image, which is definitely not natural. Therefore, a shift-invariant distance metric could be more powerful in this case.

For given two images  $I_a$  and  $I^*$ , the standard (vectorized) Euclidean distance is given in Eqn. (19). This formula can be enhanced with a shift-invariant adaptation as in Eqn. (20) where  $V_a$  denotes the image  $I_a$  zero-padded on its sides. Alternatively, a shift-invariant distance notion can also be given in terms of inverse of cross-correlation as in Eqn. (21). Nevertheless, even if a suitable distance metric is found to designate the closest centroid, it is not trivial to obtain the average of a cluster as the new mean





(a) Main image of 9    (b) The number 8    (c) Another image of 9

**Figure 13.** Vectorized distance will dictate that 8 is closer to the main image. However, it is indeed more natural to say that two images of 9 are more similar to each other.

for the next step.

$$\text{dist}(I_o, I_t) = \|I_a - I\|_k = \sqrt[k]{\sum_{i=1}^n |I_a(i) - I(i)|^k} \tag{19}$$

$$\begin{aligned} \text{dist}(I_a, I_b) &= \min_{y^i, i} \sum_{i=1}^n |I_a(i) - I_b(i+x, j+y)| \\ \text{dist}(I_o, I_b) &= \end{aligned} \tag{20}$$

$$\max(\text{corr}(I_o, I_b))' \tag{21}$$

In this study, *k*-means formulation will be considered within a sparse representations framework to provide a self-sufficient shift-invariant version. As noted in earlier chapters, the original fc-means problem can be expressed in a sparse representations framework as a dictionary learning problem. A shift-invariant version of /c-means can then be derived through a much recent convolutional dictionary learning formulation. It is not a surprise that a convolutional approach leads to a shift-invariant scheme, as convolution is an operator which breaks orthogonality assumption by considering neighboring data points group by group, forming a relation between spatial regions in the signal.

The chapter is organized as follows. Section 5.2 gives the mathematical description of the proposed shift-invariant fc-means concept, followed by a generalization through convolutional dictionary learning for classification. Section

**5.3 details exper**

imental setup and reports experimental results obtained from the proposed concepts. Later, Sec. 5.4 discusses many alternatives of convolutional-logic for spatio-temporal information preservation, including a spatio-temporal hypercomplex encoding scheme. Section 5.5 finally concludes this paper with a brief summary.

## 5.2 Convolutional Sparse Representations

It is possible to mathematically formulate the conventional fc-means problem in a sparse representations framework given in Eqn. 22 as follows,

$$\begin{aligned} \arg \min_{\mathbf{A}, \{\mathbf{x}_i\}_i} \|\mathbf{Y} - \mathbf{A}\mathbf{X}\|_2^2 \text{ subject to} \\ \|\mathbf{x}_j\|_0 = 1 \wedge \|\mathbf{x}_i\|_1 = 1 \wedge 0 < \mathbf{x}^*, \forall i, \end{aligned} \quad (22)$$

where the matrix  $\mathbf{A}$  is an over-complete dictionary and  $\mathbf{X}_j$  is the sparse representation of the data point  $\mathbf{y}$ ,  $\forall i$ . Each sparse vector contains only one non-zero component and this component is forced to be positive and sum-to-one. Dictionary columns as atoms (namely  $\mathbf{a}_{fc}$ ) designate centroids.

While Eqn. (22) represents a direct formulation of classical fc-means, it corresponds to the problematic orthogonality consideration as mentioned previously. A possible shift-invariant alternative of fc-means is given in Eqn. (23) as follows,

$$\begin{aligned} \arg \min_{\{\mathbf{a}_{fc}\}, \{\mathbf{x}_i\}_{i=1}^K} \|\mathbf{y}^* - \mathbf{a}^* * \mathbf{x}_{i,fc}\|_2^2 \text{ subject to} \\ (\mathbf{a}_{fc}^* * \mathbf{x}_{i-k} = 0) \wedge \|\mathbf{x}_{i,fc}\|_1 = 1, \forall i, k, \end{aligned} \quad (23)$$

where  $*$  denotes the convolution operator and  $k^*$  is the index of the optimal convolutional atom, or in other words the convolutional centroid that is assigned to the  $i^{th}$  data point. Notice here that the non-zero entry of  $\mathbf{x}_{i,fc}$  is not forced to be 1, but can now be anything. Therefore, this formulation is not only shift-invariant but also invariant to the magnitude of the pattern. However, this should then be complemented by an atom normalization process.

Because of the linearity property, atoms in  $\mathbf{A}$  can also be expressed in a large convolutional dictionary to be denoted by  $\mathbf{D}$  as depicted in Fig. 14. The local dictio-

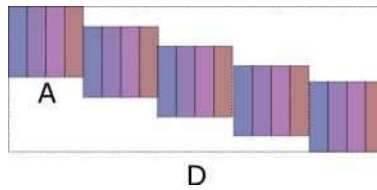


Figure 14. The local dictionary A consists of convolutional atoms, whereas the global dictionary D is filled with zeros outside the convolutional area.

nary A consists of convolutional atoms, whereas the global dictionary D is filled with zeros outside the convolutional area. In this regard, the mathematical optimization in Eqn. (23) evolves to Eqn. (24) where  $j$  denotes the index of the single non-zero element from the top and  $j$  modulo (number of clusters) determines the index of the assigned convolutional centroid  $k^*$ .

$$\begin{aligned} \underset{D, \{i\}}{\operatorname{argmin}} \quad & \|y_j - Dx_j\| \quad \text{subject to} \\ & (k^* = j \% \#fc) \quad A \quad \|x_j\|_0 = 1, \forall i. \end{aligned} \quad (24)$$

### 5.2.1 A solution to shift-invariant k-means

Since the optimization in Eqn. (24) is highly non-convex, an approximate iterative solution is employed alternating between *assignment to clusters* and *centroid update* akin to Lloyd's algorithm for the original fc-means problem (Jain, 2010). This procedure directly corresponds to *sparse coding* and *dictionary update* steps, respectively, in terms of sparse representations.

In this light, the data assignment step is solved with Orthogonal Matching Pursuit (OMP) (Pati et al., 1993) assuming D is fixed, to satisfy the  $f_0$ -norm sparsity constraint. On the other side, a straight-forward utilization of conventional dictionary update algorithms, such that Method of Optimal Directions (MOD) (Engan et al., 1999) or KSVD (Aharon et al., 2006), is not very obvious because the inherent subdictionary A composed of convolutional centroids is only to be updated in D. To solve this problem, each individual block of the overall sparse representation is extracted as an individual subproblem, on which MOD (i.e., least-squares) update is

applied. As the last step, the final updated subdictionary  $A$  is attained by averaging all of the resulting individual subdictionaries. To the best of the available knowledge, this naive solution to the centroid update problem is not extensively covered in literature, thus it can be coined as Method of Optimal Subdirections on Average (MOSA).

Experimental results indicate that this adaptation of shift-invariant fc-means provides better results when compared to its original version for datasets in which considerable shifts exist.

### 5.2.2 *Convolutional dictionary learning as a generalization*

Encouraged by the superiority of the shift-invariant fc-means formulation obtained through a convolutional sparse representation as an unsupervised task, the question is then to generalize this convolutional approach to other machine learning tasks such as classification. The claim is that an unsupervised feature extraction layer that is performed through convolutional dictionary learning as a generalization, can provide superiority over orthogonal-only consideration in also supervised tasks. This claim has already been validated in literature many times (Zeiler et al., 2010; Pu et al., 2016; Garcia-Cardona and Wohlberg, 2018) but an extensive comparison with the classical orthogonality consideration is usually missing.

In this regard, a shift from the strict  $\ell_0$ -norm constraint to a more lenient Minorai is considered. There are two main reasons behind this decision. First of all, it is unclear how to set the sparsity level in an  $\ell_0$ -norm formulation since denser choices drastically affect the computational complexity in greedy approaches and sparser solutions can lead to severe information loss. Importantly, most practical studies are based on  $\ell_1$ -norm in literature (Garcia-Cardona and Wohlberg, 2018).

With the above consideration, a final optimization for convolutional dictionary learning is given in Eqn. (25) by introducing the  $\ell_4$ -norm regularization into the formula via a Lagrange multiplier  $\lambda$ . Iterative solutions which alternates between convolutional sparse coding and dictionary update exist in literature (Garcia-Cardona and

Wohlberg, 2018).

$$\argmin_{\{a_{fc}\}, \{x_{i,fc}\}} \sum_i \sum_k \left( \frac{1}{2} \|x_i - \sum_k a_{fc,k} x_{i,fc,k}\|^2 + \lambda \sum_k \|a_{fc,k}\|_1 \right) \quad (25)$$

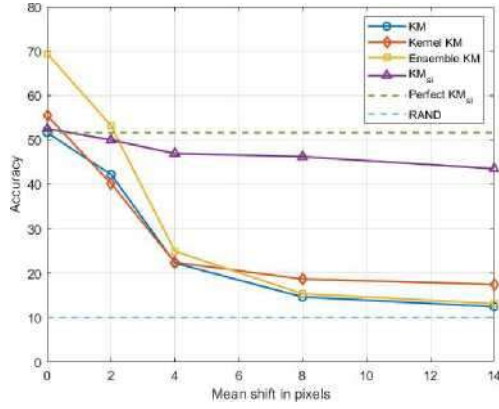
In fact, the aim of this chapter is not to devise new approaches to above optimization but to utilize it as an approach to the orthogonality problem. This unsupervised convolutional decomposition of a signal can be regarded as a feature extraction method that tackles the problem of orthogonality, where the extracted features for the  $i^{th}$  data point  $y^*$  are formed by concatenating the corresponding sparse codes, i.e.,  $z_i = [\{x^{\wedge} x\}, \{x_{ij2}\}, \dots]$ . Note that concatenation here still assumes orthogonality; however, there now exists a convolutional-logic before the orthogonality consideration which alleviates the main drawbacks of it from the start. The effectiveness of such a layer is to be experimentally tested against various other feature extraction methods in an extensive manner.

### 5.3 Experimental Results

In the following, two sets of experiments are performed corresponding to the discussions raised in Sec. 5.2.1 and Sec. 5.2.2. All experiments are carried on an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz 16GB RAM machine running on Microsoft Windows 10 using Matlab 2019a.

#### 5.3.1 Shift-invariant k-means

In this set of experiments, a dataset is formed by extracting first 1000 training images of each class from the MNIST handwritten digit database (LeCun et al., 2010), making a total of 10000 images. Four modified versions of this dataset are then obtained to test the shift-invariance property. First of all, empty images of sizes 32 x 32, 36 x 36, 44 x 44 and 56 x 56 pixels are initialized and original digits are inserted into these widened images with certain uniformly random shifts in  $x$  and  $y$  directions. Mean shifts in axes are chosen as 2, 4, 8 and 14 pixels, respectively, suiting the size of images. The clustering accuracy rates of fc-means (KM), Kernel KM (Dhillon et al., 2004),



**Figure 15.** Clustering accuracy (%) of fc-means (KM) based methods as a function of mean shift applied on MNIST. The proposed shift-invariant KM (KM<sub>si</sub>) is robust to shifts.

Ensemble KM (Iam-on and Garrett, 2010) and shift-invariant KM (KM<sub>si</sub>) on these cases are illustrated in Fig. 15.

Not surprisingly, the performance of KM<sub>si</sub> stays relatively stable in cases of varying shifts, whereas all other methods start to perform poorly when shifts are introduced. It is obvious that a mean shift of 4 pixels is enough to disrupt the functionality of classical methods for these datasets. Considering original images of sizes 28 x 28 pixels, this roughly corresponds to a mean shift of 14% of the whole image size. Note also that classical methods perform nearly poor as a random guess method (RAND) in cases of extreme shifts, e.g., 14 pixels or correspondingly 50%. KM has 12.47%, Kernel KM has 17.48%, Ensemble KM has 13.15% and KM<sub>si</sub> has 43.53% clustering accuracy in the case of 14 pixels shift applied on MNIST. This proves that neither kemelization nor ensembles can provide an efficient solution to the shift-invariance problem.

One may argue that a simple preprocessing step, which extracts a precise subimage of the digit in each image, would be enough to sustain shift-invariance for clustering these images; however, such a naive approach cannot be a general solution for natural images. On the other hand, the logic in KM<sub>si</sub> provides an automatic solution, which is both theoretically and practically sound, without any need for pre-processing. The simplicity and effectiveness of this clustering approach can further

pave way to more general techniques with the same logic applied on other machine learning tasks in some settings. In fact, a generalization of  $KM_{si}$  via convolutional dictionary learning can be utilized as a powerful unsupervised feature extraction method for classification that alleviates the drawbacks of the classical orthogonality consideration.

### 5.3.2 Convolutional dictionary learning

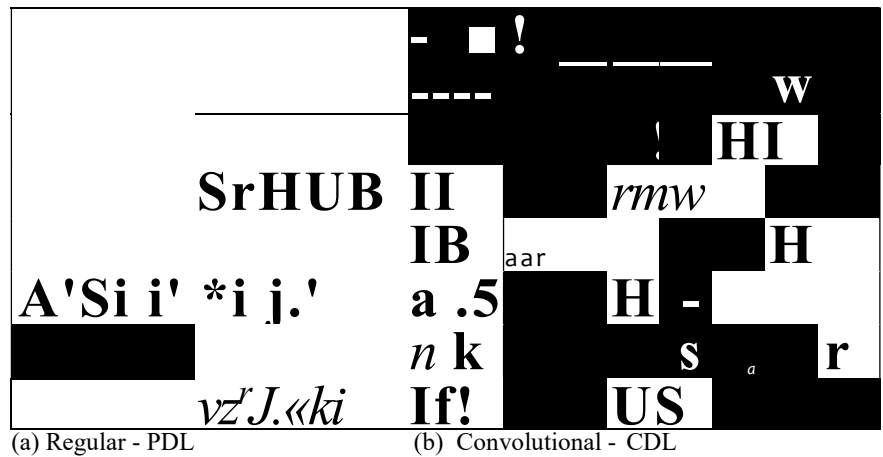
In this set of experiments, convolutional dictionary learning as an unsupervised feature extraction method is compared against various other well-known feature extraction schemes. An existing library called SPORCO (Wohlberg, 2017) is utilized for convolutional dictionary learning. In the following reported experiments, linear support vector machine (SVM) classifiers are employed after the feature extraction phase. The motivation behind the linear SVM usage is that, a successful feature extraction must transform the sample space into a linearly separable one as much as possible.

There are three employed versions of dictionary learning methods. The global- only dictionary learning (DL) operates over dictionary atoms of size  $28 \times 28$  pixels, namely atoms cover sample images globally. The patch-based dictionary learning (PDL) trains over dictionary atoms of size  $11 \times 11$  pixels, where local image patches are extracted in a sliding window manner. This type of approach can be regarded as a local- only one. Both DL and PDL methods are realized through regular dictionary learning iterative steps, i.e., sparse coding and dictionary update. In the proposed method, namely convolutional dictionary learning (CDL), atoms are of size  $11 \times 11$  pixels but now Eqn. 25 is in action instead. While considering the structure of the dictionary in a 2D form of Fig. 14, CDL can be classified as a both local and global approach. Effects of regular versus convolutional approaches are apparent in the learned atoms at the end of the training process as exemplified in Fig. 16. Notice that convolutional approach results in filters having Gabor-like appearance.

Other well-known methods that take spatial information in images into account are Histogram of Oriented Gradients (HOG) (Dalai and Triggs, 2005), Local



Binary Patterns (LBP) (Ojala et al., 1996) and Gabor Feature Extraction (GFE) (Haghighat et al., 2015). For HOG, a cell size of  $8 \times 8$  is chosen with 9 orientation histogram bins



**Figure 16. Patch-based versus convolutional dictionaries learned on MNIST. For a clear visualization, atoms are of size  $8 \times 8$ .**

and signed orientation is not used. For LBP, number of neighbors is 8 and radius of circular pattern to select neighbors is determined as 2. Rotation information is also encoded. The cell size is 5 and no normalization is performed. In GFE, a Gabor filter- bank of 15 filters is employed of size  $11 \times 11$  with 3 different scales and 5 orientations.

Another important categorization of methods is given through whether they perform dimensionality reduction or expansion. The last two methods to be mentioned, namely Autoencoders (AE) and Principal Component Analysis (PCA) both perform dimensionality reduction. Notice that HOG and LBP also accomplish effective dimensionality reduction while other methods instead go through an expansion process. A pooling procedure is closely tied to expansion in case of spatial methods, and is usually performed to reduce the computational cost with the advantage of certain rotation/position invariance. In methods with dimensionality expansion (DL, PDL, CDL, GFE), DL and PDL do not perform an additional pooling since they do not truly preserve spatial configuration. Although PDL takes local spatial information into account, there is no trivial way to perform a meaningful pooling on

top. On the other hand, CDL contains a max pooling layer and GFE has an average pooling layer, of cell sizes  $2 \times 2$  in both cases.

Table 8 summarizes all feature extraction methods in the benchmark. Note that

Table 8. Feature extraction methods in the benchmark.

	DL	PDL	CDL	HOG	LBP	GFE	AE	PCA
Learning	✓	/	/	X	X	X	/	/
Spatial	X	✓	✓	✓	✓	/	X	X
# Dimensions	5880	5880	2940	144	250	2940	100	100

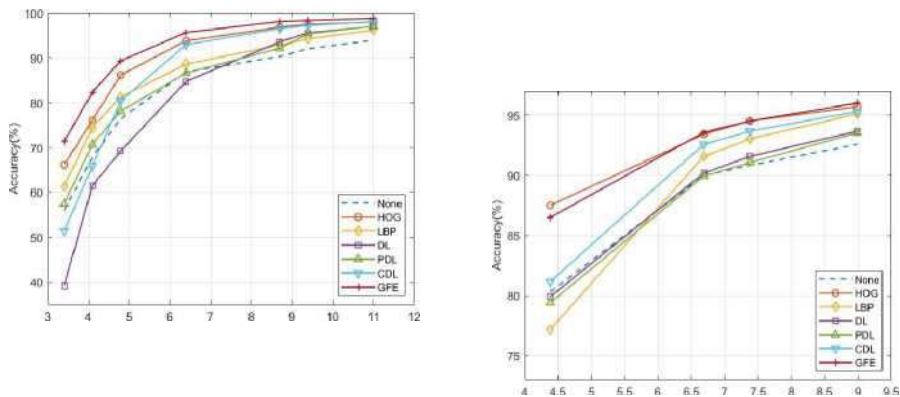


Figure 17. Classification accuracy (%) as a function of varying training sizes applied on (top) MNIST and (bottom) USPS.

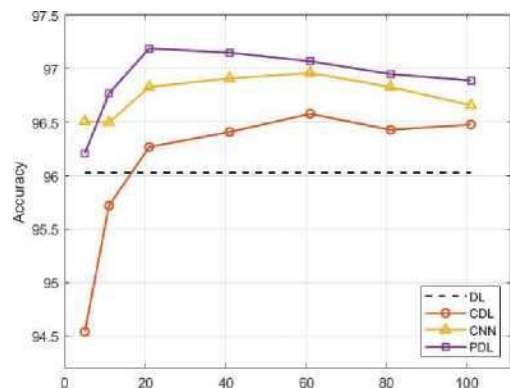
“Spatial” attribute appearing in this table is an antonym for the word “orthogonality” in the context of this study. For example, both PDL and CDL can be described as spatial methods since they process images by considering pixels within certain local neighborhoods. However, each pixel is indistinguishable from the others in DL because of the vectorization of the whole frame, resulting in an orthogonality consideration.

After having described all methods in detail, Fig. 17 depicts classification performance as a function of varying training sizes applied on MNIST and USPS (Hull, 1994) databases. As a global-only dictionary learning method, the inferior performance of DL in case of small training sizes is obvious. A similar behavior is also slightly observable in CDL as a both global and local dictionary learning approach. Although PDL does not perform poorly in small training sizes, it does not provide noticeable advantage over DL in the long run, while CDL outperforms both DL and PDL performing at the capacity of HOG when most of the dataset is used. HOG and GFE together compete for the top performance, whereas CDL performs a little

poorer but it

**Table 9. Classification accuracy (%) of feature extraction methods with linear SVM applied on the whole MNIST and USPS datasets.**

Dataset	DL	PDL	CDL	HOG	LBP	GFE	AE	PCA
MNIST	97.04%	97.07%	98.51%	97.99%	96.21%	98.80%	94.34%	94.15%
USPS	93.67%	93.52%	95.31%	95.71%	95.11%	96.01%	92.02%	92.72%



**Figure 18. Classification accuracy (%) as a function of different patch/kernel sizes applied on (preprocessed) MIT-BIH using linear SYM classifiers.**

is better than LBP. Most importantly, it is apparent that PDL cannot be an alternative to convolutional-logic at least for the 2D case.

Table 9 lists the final classification accuracy results with linear SVM applied on the whole MNIST and USPS databases. GFE is the top performing method as an unsupervised simulation of first layers of a convolutional neural network (CNN). Additionally, CDL and HOG compete for the second place.

The convolutional dictionary learning concept is further applied in a 1D setting. The MIT-BIH arrhythmia dataset (Moody and Mark, 2001), in which the signals correspond to electrocardiogram (ECG) shapes of heartbeats for cases unaffected (normal) and affected by different arrhythmias, is used. These signals are preprocessed and segmented, each segment represents a heartbeat, one of the five different classes (Kachuee et al., 2018).

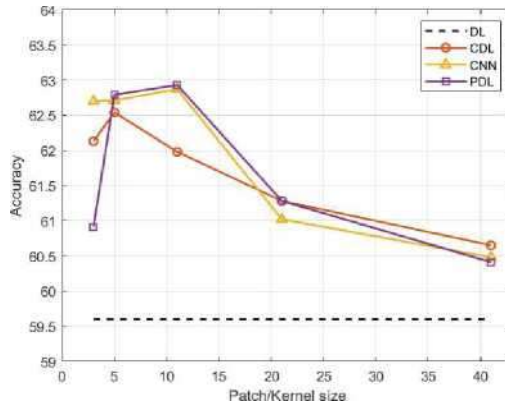
Preliminary experimentation suggests that the results could be highly dependent on the chosen patch/kernel size as CDL performs poorly for small patch/kernel sizes. These results are summarized in Fig. 18. In this figure, all methods are devised

to be resource-wise equivalent, i.e., they have equal dimensionality of features. DL, PDL and CDL algorithms have the same definitions as in 2D while they are translated into 1D equivalent versions. Finally, CNN here denotes a 1D convolutional neural network as a substructure of a regular 2D version. For a fair comparison, the architecture of CNN is composed of a convolutional layer, a batch normalization layer, a ReLU layer, a max pooling layer, a fully connected layer, a softmax and a classification layer. In other words, the convolutional-logic is applied once (without getting deep) before the classification stage.

The main observation here is that all spatially-aware methods (PDL, CDL, CNN) outperform the orthogonality consideration of DL, as long as the patch/kernel size is of enough size. It is apparent that a relatively small patch sizes cause CDL to perform very poorly. Such behavior is not observable for CNN which performs well for all kernel sizes chosen. The most surprising result is that PDL outperforms CNN nearly for all cases. However, note that CNN here does not have a deep architecture. The other surprising point is that CDL is the worst among all spatially-aware methods. It is possible that the employed SPORCO library may not be optimized for 1D settings.

To verify the generality of above results, another 1D problem from a different domain is chosen for the classification of electric devices according to their electric usage profile through raw data. The dataset is obtained from (Chen et al., 2015) and it contains 8926 train and 7711 test samples of size  $1 \times 96$ , with 7 possible classification labels. In parallel to Fig. 18, quite similar results are obtained in Fig. 19. With enough patch/kernel size, PDL performance is similar to that of CNN. All methods outperform the baseline of DL.

Inspired by all above experiments measuring the effect of patch/kernel size, the final simulation results on the patch/kernel effect (using the whole MNIST database) are depicted in Fig. 20. It is clearly observable that CDL nearly matches the performance of a shallow CNN, while PDL performs poorly in this 2D case. As a conclusion, one can expect PDL as an alternative to CNN in 1D and CDL in 2D, as long



**Figure 19. Classification accuracy (%) as a function of different patch/kernel sizes applied on the raw Electric Devices dataset using linear SVM classifiers.**

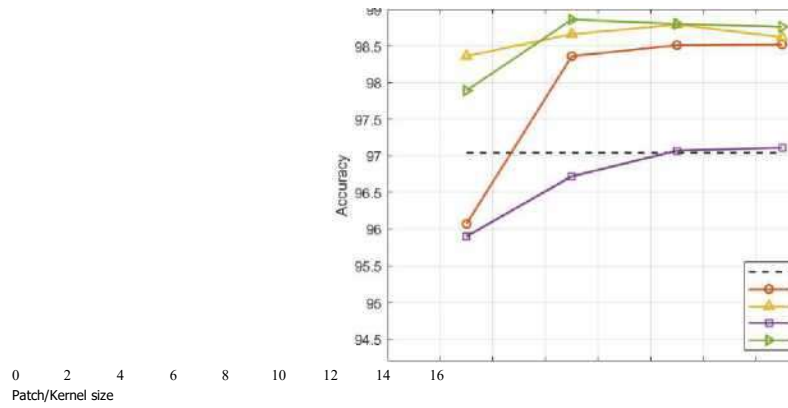
as patch/kernel size is sensible. Another note is that GFE followed by a linear SVM classifier is a viable unsupervised way of simulating a shallow CNN.

## 5.4 Discussion

### 5.4.1 Variations on neural networks

In fact, it is not a coincidence that Convolutional Neural Networks (CNNs) can surpass the capacity of Support Vector Machines (SVMs) especially for signal applications, as they tackle the problem of orthogonality with the convolution operator, whereas SVM formulation fully assumes dimensions to be orthogonal as in standard  $n$ - dimensional Euclidean space, much like the problematic original  $fc$ -means formulation.

More specifically, convolution with a kernel in input side of the layer corresponds to creating a hyperedge between input side nodes in question whether it be a 1D (Kiranyaz et al., 2015) or 2D or even 4D CNN (Choy et al., 2019). In other words, neighboring cells now occur in a relation, preserving the original spatial configuration. Then, as an alternative to convolutional approach, neighboring cells in the input or the output side of the layer can also be put in relation with real edges in-between, as another way of preserving the original spatial configuration that the input cells have.



**Figure 20. Classification accuracy (%) as a function of different patch/kernel sizes applied on MNIST using linear SYM classifiers.**

The idea of inserting edges in-between output nodes of a layer is problematic, as for performing any meaningful propagation a direction is needed for each edge. Inherently, all edges direct to the right in a conventional neural network architecture, but possibility of edges in-between in the same layer then forces to think of a neural network as a more general directed graph. With this new architecture, namely neural networks as general directed graphs, both forward and backward propagation must be reconsidered.

In fact, this line of logic leads to an alternative structure known as recurrent neural networks. Simulating the hearing process, connections between the nodes now form a directed graph along a temporal sequence. As an advantage, RNNs can process variable length sequences of input. It is possible to designate two broad classes of RNNs, where one is finite impulse and the other is infinite impulse. Finite impulse recurrent network is a directed acyclic graph that can be replaced by a conventional neural network, whereas infinite impulse recurrent network is a directed cyclic graph that cannot be replaced. Without the edges in-between as in an original fully connected multilayer perceptron two nodes in any hidden layer can be swapped in theory, not disrupting the topology of the network, or in other words not resulting with a noticeable change. These ideas are depicted in Fig. 21. Note that, it is possible to build upon basic recurrent neural network structure through bidirectional logic (Schuster and Paliwal,



**Figure 21. On the left graph there are edges within the hidden layer. On the right there is the classical fully connected neural network. Switching two nodes within the hidden layer makes no difference for classical neural network.**

1997), and long-short term memory concept (Sak et al., 2014).

However, empirical evaluation suggests that temporal convolution, or in other words, 1D convolutional logic surpasses the capacity of recurrent architectures in sequence modeling (Bai et al., 2018). It is still an open question whether regarding temporal dimension as a just another spatial dimension is the way to go or whether a hybrid approach is better. This is rather a deep issue related to properties of space and time. Instead, regarding neural networks of any structure as directed and possibly cyclic graphs, or in other words as neural graphs, might pave way to better understand the brain. Note that, this concept is rather different than graph neural networks, which use graphs as inputs (Scarselli et al., 2008).

Another generalization for neural networks is possible by considering infinite width neural networks (Arora et al., 2019). Recent results suggest that deep NNs that are allowed to become infinitely wide converge to models called Gaussian processes (Lee et al., 2017). However, such studies do not consider the case when there are in-between connections within the layers.

There is another futuristic approach that bypasses the problems of infinite width. To come up with such a structure, one needs to apply concepts of sparsity to neural networks in a creative manner. In this sense, the concept of partial propagation

**Figure 22. Concept of partial propagation. Partial propagation can be performed even when the hidden layer is composed of infinitely many nodes.**

is important. By partial propagation it is meant that the best output nodes are chosen in calculation that will minimize the final error, instead of blindly forward propagating the input on all output nodes.

Considering partial propagation in the domain of neural networks paves way

to interesting formulations. As exemplified in Fig. 22, having infinitely many nodes in the intermediate layer is not a problem when partial propagation is at hand. A conventional full propagation would be not possible in such a case. This can further lead to having an infinite but continuous layer (input or output), which is practically applicable with integration, as depicted in Fig. 23.

It is now important to give some mathematical formulation for cases depicted in Fig. 23 as single layered fully connected formulations.

In discrete to continuous case, the output layer can be regarded as a function of  $t$ . Assuming that there are 2 inputs the output can be given as  $x(i) = y_1 w_1(t) + y_2 w_2(t)$  where  $y_i$  designate the input values and  $W_i(t)$  designates weights corresponding to  $i^{th}$  input as a function of  $t$ . Continuous to discrete case is interesting because the input can now be a function itself, thus can pave way to the concept of *functional machine learning*. Let us denote input as a function of  $s$ , namely  $y(s)$ . Then  $i^{th}$  output is expressed as  $\int y(s) w_i(s) ds$ , where  $w_i$  is the weights function associated with  $i^{th}$  output this time. Last case is the most interesting one, as effectively it maps a function to

Figure 23. A generalization of neural network layer cases. From left to right, discrete-discrete (classical), discrete-continuous, continuous-discrete, and continuous-continuous cases are depicted.

another function. Let  $y(s)$  again denote the input. This time the weights can be formulated as a function of two variables namely  $s$  and  $L$ . As a result output becomes  $x(t) = \int y(s) w(s, t) ds$ . Forward and back propagation can be performed easily by mathematical tools capable of symbolic expression processing such as Matlab's Symbolic Toolbox. Preliminary experiments suggest that this line of generalization might provide superior capabilities at the expense of computational complexity.

Notice that, a continuous layer counts towards a layer in which there exist in-between edges with-in infinitely many nodes. However, this alone may not be enough to preserve the spatial configuration of the input layer. Therefore, additionally a sparsely overlapping block-wise connected continuous layered network is chosen as

the final architecture as an alternative to a CNN as depicted in Fig. 24. Higher dimensional analogous can also be devised. Experimentation with proposed structures were very limited due to conceptual and computational burden. Therefore, as a last resort to preserving spatial information in a simpler manner, multilinear approach mostly from the perspective of sparse representations is mentioned in the upcoming section as a promising analytical solution to the problem.

**Figure 24. A sparsely overlapping block-wise connected continuous layered network as an alternative to a 1D convolutional neural network in preserving the spatial information of a 1D signal.**

#### **5.4.2 Multilinear approach**

There are two main directions for approaching multidimensional signals in a multilinear fashion. In conventional linear algebra based representations, data points are always vectorized and thus treated as vectors, so such representation ignores the local spatial information of a multidimensional form. To overcome this, as the first solution, tensor representations treat data points as tensors in their original form, preserving the correlation in multiple dimensions (Roemer et al., 2014). As the most simplistic example of the other solution, one can use a complex-valued vector and encode additional information in the imaginary part. As an extended example, one can represent a color image as a matrix of quaternions (Xu et al., 2015). Utmost generalization of this second solution is reached through general frameworks based on what is called geometric algebra (Wang et al., 2019). These two multilinear approaches are to be examined thoroughly from the perspective of sparse representations.

The idea that images are not vectors, thus vectorization breaks the spatial coherency of images is investigated by Hazan et al. (2005). In fact, while linear algebra methods are organized under the hood of matrix factorization, this line of thought is centralized around tensor factorization instead as a generalization. Hazan et al. (2005) report that by treating training images as a 3D cube and performing a nonnegative tensor factorization (NTF) on it, they achieve higher efficiency,

discrimination and representation power than nonnegative matrix factorization (NMF). Specifically, they consider the following least-squares problem in Eqn. (26)

$$\underset{\{\mathbf{u}^m, \mathbf{v}^m, \mathbf{w}^m\}}{\operatorname{argmin}} \sum_{m=1}^M \|\mathbf{G} - \sum_{m=1}^M \mathbf{u}^m \otimes \mathbf{v}^m \otimes \mathbf{w}^m\|_F^2, \text{ subject to } \mathbf{u}^m, \mathbf{v}^m, \mathbf{w}^m \geq 0, \quad (26)$$

where  $\mathbf{G}$  is the 3D data cube mentioned to be factored into a sum of  $k$  rank-1 tensors and  $\mathbf{u}^m \otimes \mathbf{v}^m \otimes \mathbf{w}^m$  corresponds to three fold outer-product. Variations on such nonnegative tensor factorization formulation are also performed successfully on the problem of music genre classification based on tensor-based representation of audio signals (Benetos and Kotropoulos, 2008; Panagakis et al., 2009).

At this point, it is important to note that there are two main branches of tensor-based approaches corresponding to type of tensor decomposition they seek. As the first branch, there are studies (including the ones mentioned up to now) based on canonical polyadic decomposition (CPD), sometimes also referred to as CANDECOMP/PARAFAC (Kolda and Bader, 2009). As a generalization of methods mentioned, the equation takes the form of Eqn. (27)

$$\mathbf{A} = \sum_{i=1}^r \mathbf{X}_i \mathbf{A}_i^{(1)} \otimes \mathbf{A}_i^{(2)} \otimes \dots \otimes \mathbf{A}_i^{(d)} \quad (27)$$

where  $\mathbf{A}$  is  $d$ -dimensional tensor to be represented as a linear combination of  $r$  rank-1 tensors, or namely vectors. The most relevant example from literature is K-CPD (Duan et al., 2012), an algorithm of overcomplete dictionary learning for tensor sparse coding based on a multilinear version of orthogonal matching pursuit and CANDECOMP/PARAFAC decomposition. The framework is introduced in Eqn.(28).

$$\mathbf{y} = \sum_{i=1}^r \mathbf{X}_i \mathbf{A}_i = \sum_{i=1}^r (\mathbf{x}_i \otimes \dots \otimes \mathbf{g}_i \otimes \mathbf{a}_i) \text{ s.t. } \|\mathbf{x}_i\|_0 \leq k, \mathbf{x} = [\mathbf{x}_1, \dots, \mathbf{x}_r], \quad (28)$$

where  $\mathbf{y}$  is a single tensor element,  $\mathbf{x}$  being the sparse vector of a single tensor element. Note that, the dictionary becomes  $d + 1$  dimensional tensor if all atoms are thought together. Let  $\mathbf{A}^n = \{\mathbf{A}\}$  be the whole dictionary, and similarly  $\mathbf{y}^n = \{\mathbf{y}, -\}$  all of

the training data, then the equation to be optimized is given as in Eqn.(29)

$$\underset{\mathbf{A}^n, \mathbf{x}}{\operatorname{argmin}} \|\mathbf{y}^n - \mathbf{A}^n \mathbf{x}\|, \text{ subject to } \|\mathbf{x}\|_0 \leq k \quad \forall j, \quad (29)$$

This equation can be iteratively solved by alternating between sparse coding and dictionary update as Duan et al. (2012) propose. K-CPD surpasses conventional methods in a series of image denoising experiments. Most recently, this framework is also successfully utilized in tensor-based sparse representations of multi-phase medical images for classification (Wang et al., 2020).

The second and most prolific branch of tensor-based sparse representations is centered around the Tucker decomposition model instead, which is a more general model than CPD (Caiafa and Cichocki, 2013). However, before going into details of Tucker model, it is important to make the distinction between tensor sparse coding, and tensor dictionary update procedures much like in standard setting.

Literature for tensor sparse coding has three main claims. One branch claims that tensor sparse coding gives equivalent results to ID vectorized version, but time complexity and memory usage can be reduced significantly (Fang et al., 2012). Second branch specifically applies tensor sparse coding on positive definite matrices with increased performance (Sivalingam et al., 2010). Last branch applies structured sparsity on top of sparse coding, a multidimensional form of block-sparsity to be exact, and achieves significant gains (Caiafa and Cichocki, 2013).

In the branch that claims equivalence, Fang et al. (2012) has proposed 2D-OMP, where each atom in the dictionary is a matrix. It is reported that higher dimensional generalization is also possible by utilizing separable sampling. Similarly, Roemer et al. (2014) report that a simple way to redefine sparse coding in terms of tensors is not

found, and they choose to use conventional BP, or OMP methods for sparse coding. On the other hand, Sivalingam et al. (2010) report a novel tensor sparse coding approach for positive definite matrices, where vectorization will destroy the inherent structure of the data. In this case, the sparse decomposition is formulated as a convex optimization problem, belonging to a category of determinant maximization problems to be solved efficiently by interior point algorithms. By using region covariance descriptors as introduced by Tuzel et al. (2006), it is possible to convert images into positive definite form. Although capable of surpassing state-of-the-art at the time, the study does not address dictionary learning on top of sparse coding. Introducing block-sparse coding in tensor form and also offering the most comprehensive coverage of Tucker decomposition model through Kronecker structured dictionaries, Caiafa and Cichocki (2013) form the second and most prolific branch of tensor-based sparse representations.

Caiafa and Cichocki (2012) lay the foundations of Tucker decomposition model by defining Tensor-OMP algorithm that computes a block-sparse representation of a tensor with respect to a Kronecker basis. First of all, it is important to define Kronecker product of two matrices A (I by M) and B (J by N) as in Eqn. (30).

$$A \otimes B = \begin{pmatrix} a_{1,1}B & a_{1,2}B & \cdots & a_{1,M}B \\ a_{2,1}B & a_{2,2}B & \cdots & a_{2,M}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{I,1}B & a_{I,2}B & \cdots & a_{I,M}B \end{pmatrix} \quad (30)$$

The introduction of the model is given by the equation for the 2D case as in Eqn. (31)

$$Y = D_1 X D_2, \quad (31)$$

where  $D_1$  and  $D_2$  are two dimensional matrices associated with mode-1(columns) and mode-2(vectors). Matrix X is the sparse coefficient matrix. Note that Eqn. (31) can be rewritten in vectorized form as follows in Eqn. (32).

$$y = \text{vec}(D_1 X D_2) = (D_2^T \otimes D_1) x, \quad (32)$$

where  $\circ$  stands for Kronecker product. This equation states that, vectorized version of signal  $Y$  can be written as a standard linear combination of elements of a dictionary, which specifically has a Kronecker structure  $D = D_2 \circ \dots \circ D_i$ . Generalizing this methodology to any dimensions is possible through Tucker decomposition model of Eqn. (33) and its equivalent vectorized form in Eqn. (34). In this case,  $x^*$  stands for i-mode tensor matrix product.

$$y = X \underset{x^1}{x^1} D_1 \underset{x^2}{x^2} D_2 \dots \underset{x^v}{x^v} D_v \quad (33)$$

$$y = (D_v \circ D_{v-1} \circ \dots \circ D_1) x \quad (34)$$

In Caiafa and Cichocki (2013), authors report that a block-sparse structure imposed on  $X$  in its original form through sub-tensors provide significant results. They extend their earlier work by defining a Kronecker-OMP algorithm, that utilizes the Kronecker structure of the dictionary. They also propose N-BOMP (N-way block OMP), with block-sparsity imposed. It is important to note that, Tucker model together with block-sparsity restriction may work significantly well together as the higher dimensional block structure is meaningfully applied on the original sparse tensor  $X$  in the form of sub-tensors.

There are many studies in literature specifically based on this Tucker model of sparse representations with or without block-sparsity and additionally including dictionary learning. As an example, a two dimensional synthesis sparse model is sketched (Qi et al., 2013). Based on Tucker model, it is also possible to formulate existing popular least-squares based dictionary learning algorithms in tensor-based form, through T-MOD and K-HOSVD algorithms (Roemer et al., 2014). However, as Peng et al. (2014) demonstrate for multispectral image denoising, Tucker model has superior performance when applied together with group-block sparsity regularizer, corresponding exactly to the concept of block-sparsity in Caiafa and Cichocki (2013). In this way, a nonlocal sparse representation is possible for an originally patch based approach.

It is important to note that, certain parallels can be drawn between our earlier subject convolutional dictionary learning and tensor-based sparse representations. As an example, Huang and Anandkumar (2015) propose a novel framework for learning convolutional models through tensor decomposition and show that cumulant tensors have a CP decomposition, whose components correspond to convolutional filters and their circulant shifts.

However, tensor-based approaches (both CP and Tucker models) do not still provide a solution to ID case. Without loss of generality, let us assume that our signal is in the form of a column vector  $s$ . Since the signal is 1 dimensional there will be a single matrix of that single dimension in Tucker model. Therefore, Tucker model attained is the starting part in Eqn.(35). In this case, it is possible to show that  $s = X \cdot X^T \cdot D \cdot X = \sum_{i=1}^r X \cdot X^T \cdot D \cdot X$ . From the other domain, namely CP model perspective, we have  $s = \sum_{i=1}^r x_i \cdot x_i^T \cdot D \cdot x_i$  where  $x_i^*$  is the single sparse coefficient associated with  $i^{th}$  atom. In both cases, one arrives at standard formulation, namely Tucker and CP models are equivalent in one dimensional case.

$$s = X \cdot X^T \cdot D \cdot X = \sum_{i=1}^r X \cdot X^T \cdot D \cdot X \quad (35)$$

This brings up an important question onto the table. Although tensor-based approaches provide advantage when the signals are multidimensional, current formulations will not provide an edge for ID signals as this equivalence suggests. The remedy may come from seeing a ID signal, not as a ID vector, but more. For example, a ID complex vector can be formed by coding the cell positions in the imaginary part to overcome orthogonality problem in standard ID vector as depicted in Fig. 25. This paves way to performing sparse representations of complex valued data, or even quaternion valued data to accommodate more information in case of higher dimensional data. Utmost generalization is achieved through geometric algebra as a generalization of hypercomplex numbers.

#### 5.4.2.2 *Complex, hypercomplex, and Geometric algebra based approaches*

For an approach to tackle problem of orthogonality properly, it should be



**viable for ID signals to start with as mentioned before. After all, all of the information to sustain organic life is based on ID signals with only 4 distinct values, namely the DNA, marking the importance of ID signals as key components.**

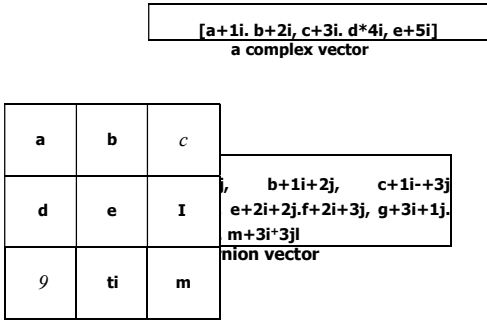


Figure 25. An encoding scheme to preserve spatio-temporal  
information for (top) ID mono audio and (bottom) 2D grayscale image  
cases.

We have sketched an original way to encode spatial/temporal information into input through complex values. In an extended application, it is possible to formulate a quaternion-valued sparse representation of color images that surpasses the conventional logic (Xu et al., 2015). Note that, quaternion algebra is the first hypercomplex number system to be devised that is similar to real and complex number systems (Moxey et al., 2003). Xu et al. (2015) state that compared to tensor-based model, quaternion-based model can achieve a more structured representation. Comparisons between QSVD and T-SVD (Tensor SVD) (Kilmer and Martin, 2011) suggest their equivalence, but superiority of QSVD arises when it is combined with the sparse representation model.

Xu et al. (2015) state that there are 4 possible models to represent color images. These is the monochromatic model, in which each channel is represented separately. There is the concatenation model, where a single vector is formed by concatenating three channels (Mairal et al., 2008b). There is also the tensor-based model, where the color image is thought of as a 3D cube of values. Finally, there is the quaternion- based model where each channel is assigned to each imaginary value (i.e. r,g,b to i,j,k respectively). Most importantly, they analytically unify all these models in their formulation.

However, there is one more possible model that is subtler. As, once can encode a mono audio as a vector of complex values where complex values indicate the timed

position, in a similar way one can encode a grayscale image as a quaternion-valued vector, where imaginary parts are used to indicate the pixel position. Then, again thinking color image as a 3D cube, there is a possible quaternion model in which imaginary units encode the position within the cube and the scalar denotes the value within that cell. Same quaternion encoding can be applied to any 3D scalar data.

There is one more issue with regard to machine learning. Xu et al. (2015) formulate a sparse representation framework that has quaternion-valued sparse codes in return. In another words, it is a quaternion to quaternion layer. Therefore, for further machine learning, a hypercomplex to real feature extraction layer is needed, as currently mainstream classification algorithms need real-valued data. Another option is to consult classification algorithms that can directly handle hypercomplex valued data. This line of logic paves way to consider complex/hypercomplex valued neural networks as viable tools (Hirose, 2012; Isokawa et al., 2003). As a future work, comparison of spatial/temporal encoded hypercomplex neural networks with conventional convolutional or recurrent neural networks may pave way to deeper understanding of deep learning. As a motivation, the fact that a single complex-valued neuron can solve the XOR problem be given (Nitta, 2003). Also, the fact that quaternions can be used to implement associative memory in neural networks is promising (Chen et al., 2017).

Another fine of generalization can deal with the case when the dataset is more than 3 dimensional, namely a volumetric animation dataset as an example. In such a case, a quaternion is not enough to designate the cell position and value. As an extension, octonion algebra can accommodate up to 7 imaginary channels (Popa, 2016; Lazendic et al., 2018); however, loses associativity property. Note that, Lazendic et al. (2018) report that all the algebras of dimension larger than 8 lose important properties, since they contain algebras of smaller dimension as subalgebras. This might be an issue related to physics of space and time, which is out of scope of this chapter. The important fact is that the study dealing with generalization of hypercomplex numbers is called Geometric algebra and is gaining attention lately (Wang et al., 2019).

### 5.5 Conclusion

This chapter aims to draw attention to orthogonal viewpoint that is taken by many machine learning methods such as fc-means or SVMs. Such viewpoint is highly ill-posed for machine learning of real world signals, which contain spatial configuration. Convolution operator can be used as a remedy for this problem, as it partially preserves the spatial information inherent in signals. However, one may need to find alternatives to convolutional approaches to further increase the understanding on this subject. Sparsity, namely spatially sparse connections in neural networks might be an alternative as described in Fig. 24. Most importantly, analytic approaches such as multilinear or geometric algebra formulations must be thoroughly investigated as alternatives, starting with ID setting to be precise.

## CHAPTER 6: CONCLUSION AND PERSPECTIVES

In this study, dictionary learning for sparse and redundant representations is modified and cast as simplicial learning that can distinguish linearly non-separable cases easily. However, this alone still does not provide a solution to the problem of orthogonality introduced in Chapter 5. Going back to our source, namely the clustering problem, one now should notice that shift invariant fc-means formulation can include rotation invariance as a more general formulation (Barth  l  my et al., 2012). Interestingly, Bar and Sapiro (2010) note that a log-polar mapping converts rotations and scalings to shifts in x and y axes respectively; therefore, invariance under general transformations is possible. In the bigger picture, convolutional logic or other frameworks that sustain invariance is related to two-stream hypothesis (i.e. where pathway and what pathway), a model of the neural processing of vision as well as hearing (Eysenck and Keane, 2005). In other words, a spatiotemporal information preserving perspective on the clustering problem brings us closer to inner workings of the brain. Also related to convolution, n-dimensional generalization of Gabor filters can be a related future work.

At the time of creation of Chapter 2, as a dominating approach in machine learning, the necessity of a deep structure was considered. In neural network research, depth comes into center attention immediately as single layer conventional

neural networks are limited in expressiveness. However, in our formulation, we are able to learn linearly nonseparable cases with a single layer. Therefore, a discussion on depth is postponed until Chapter 5, when neural networks come into play. Success of convolutional and recurrent neural networks suggest that structured depth is more expressive than multilayer perceptrons. Therefore, deep structured layers might be more important for spatiotemporal information preservation rather than being an issue of nonlin-earity. In fact, as noted before a complex valued neuron can handle nonlinearity all by itself (Nitta, 2003).

This study shows that it is possible to model generic piecewise linear constructs as a single matrix multiplication. In this regard, polytopes and more generally simpli-

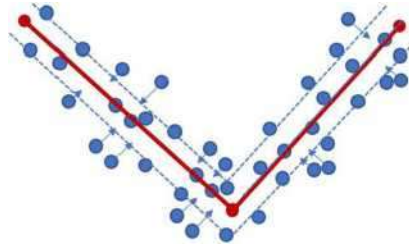


Figure 26. Noise removal as an application of simplicial learning.

cial objects come into attention. Such objects are traditionally concepts of topology and manifold learning (Lin and Zha, 2008), thus our study helps to build a bridge between. In our study, a hypergraph is kept to maintain the structure of a polytope in Chapter 3, therefore graph theory comes into play also. A future work related to hypergraphs may include an efficient algorithm for hypergraph connectivity test to designate the clusters in arbitrary dimensions. As noted before, both Chapter 3 and Chapter 4 can also be extended through kemelization and ensembling.

As the solution proposed in Chapter 4 is evolutionary, it is important to mention No free lunch theorem that states the fact that two optimization algorithms are equivalent when their performance is averaged across all possible solutions (Wolpert and Macready, 1997; Ho and Pepyne, 2002). From this perspective, the performance of Evolutionary Simplicial Learning is promising as it attains the best mean score in

the benchmark. In other words, mean score is more important in the eyes of no free lunch theorem. It is also important to mention neural gas and self-organizing map formulations as similar approaches to simplicial learning (Fritzke, 1995). The general machine learning framework introduced in this study can also be adjusted for semisupervised (Zhu and Goldberg, 2009) or self-supervised settings (Tung et al., 2017). In fact, our framework can also be applied on other signal processing tasks without much alteration. A noise removal application is given as an example in Fig. 26

Regarding Chapter 5, the spatiotemporal hypercomplex encoding scheme must be further investigated. It is also possible to merge this encoding scheme with the earlier formulations of Chapter 3 and Chapter 4, namely with additional sum-to-one and nonnegativity constraints together with structured sparsity. A final general note is the distinction between analysis versus synthesis sparse models. Throughout this thesis, synthesis model is used namely the form  $Y = AX$  where  $X$  is sparse. However, there is also the analysis model having the form  $AY = X$  where  $X$  is sparse, namely dictionary multiplied by input  $Y$  now gives the sparse codes (Shekhar et al., 2014; Gu et al., 2017). Such model is closer to neural network formulation and further investigation of analysis model might pave way to a unified perspective on sparse models that also includes neural networks.

## REFERENCES

- Aharon, M. and Elad, M. (2008). *Sparse and redundant modeling of image content using an image-signature-dictionary*. *SIAM J. Imaging Sci.*, 1(3):228-247.
- Aharon, M., Elad, M., and Bruckstein, A. (2006). *K-svd: An algorithm for designing overcomplete dictionaries for sparse representation*. *IEEE Transactions on signal processing*, 54(11):4311-4322.
- Akbari, A., Trocan, M., and Granado, B. (2016). *Image compression using adaptive sparse representations over trained dictionaries*. In *2016 IEEE 18th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1-6. IEEE.
- Arora, S., Du, S. S., Li, Z., Salakhutdinov, R., Wang, R., and Yu, D. (2019). *Harnessing the power of infinitely wide deep nets on small-data tasks*. *arXiv preprint arXiv:*

1910.01663.

- Bai, S., Kolter, J. Z., and Koltun, V. (2018). *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling*. **arXiv preprint arXiv:1803.01271**.
- Banerjee, A., Dhillon, I. S., Ghosh, J., and Sra, S. (2005). *Clustering on the unit hypersphere using von mises-fisher distributions*. **J. Machine Learn. Res.**, 6:1345—1382.
- Bar, L. and Sapiro, G. (2010). *Hierarchical dictionary learning for invariant classification*. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3578-3581. IEEE.
- Baraniuk, R. (2007). *Compressive sensing*. **Lecture Notes IEEE Signal Process. Mag.**, 24(4): 118-121.
- Baraniuk, R. G., Cevher, V., Duarte, M. F., and Hegde, C. (2010). *Model-based compressive sensing*. **IEEE Trans. Inf. Theory**, 56(4):1982-2001.
- Barazandeh, B., Bastani, K., Rafieisakhaei, M., Kim, S., Kong, Z., and Nussbaum, M. A. (2017). *Robust sparse representation-based classification using online sensor data for monitoring manual material handling tasks*. **IEEE Trans. Automation Sci. Eng.**, pp. 1-12.
- Barbarossa, S., Sardellitti, S., and Ceci, E. (2018). *Learning from signals defined over simplicial complexes*. In *IEEE Data Sci. W.*, pp. 51-55.
- Barnes, C. A., McNaughton, B. L., Mizumori, S. J., Leonard, B. W., and Lin, L. H. (1990). *Comparison of spatial and temporal characteristics of neuronal activity in sequential stages of hippocampal processing*. **Prog. Brain Res.**, 83:287-300.
- Barthélemy, Q., Larue, A., Mayoue, A., Mercier, D., and Mars, J. I. (2012). *Shift & 2d rotation invariant sparse coding for multivariate signals*. **IEEE Transactions on Signal Processing**, 60(4):1597-1611.
- Baudat, G. and Anouar, F. (2000). *Generalized discriminant analysis using a kernel approach*. **Neural Comput.**, 12(10):2385-2404.
- Belton, R. L., Fasy, B. T., Mertz, R., Micka, S., Millman, D. L., Salinas, D., Schenfisch, A., Schupbach, J., and Williams, L. (2018). *Learning simplicial complexes from*

- persistence diagrams. In *Conf. Comput. Geometry*, p. 18.
- Belyaev, Y. K. and Lumen'skii, Y. P. (1988). *Multidimensional poisson walks*. *Journal of Soviet Mathematics*, 40(2): 162-165.
- Benetos, E. and Kotropoulos, C. (2008). A tensor-based approach for automatic music genre classification. In *2008 16th European Signal Processing Conference*, pp. 1—4. TREE.
- Bengio, Y., Courville, A., and Vincent, P. (2013). *Representation learning: A review and new perspectives*. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (8): 1798-1828.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. 1st Edition. New York: Springer-Verlag.
- Blumensath, T. and Davies, M. (2006). *Sparse and shift-invariant representations of music*. *IEEE Trans. Speech Audio Process.*, 14(1):50-57.
- Blumensath, T. and Davies, M. E. (2008). *Gradient pursuits*. *IEEE Trans. Signal Process.*, 56(6):2370-2382.
- Borgefors, G. (1986). *Distance transformations in digital images*. *Comp. Vis. Graph. Image Process.*, 34(3):344—371.
- Bradley, P. S. and Mangasarian, O. L. (2000). *K-plane clustering*. *J. Global Optim.*, 16(1):23—32.
- Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pp. 93-104. ACM.
- Bryt, O. and Elad, M. (2008). *Compression of facial images using the K-SVD algorithm*. *J. Visual Commun. Image Represent.*, 19(4):270-283.
- Caiafa, C. F. and Cichocki, A. (2012). Block sparse representations of tensors using kronecker bases. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2709-2712. IEEE.
- Caiafa, C. F. and Cichocki, A. (2013). *Computing sparse representations of multidimensional signals using kronecker bases*. *Neural computation*, 25(1):186-220.
- Canas, G., Poggio, T., and Rosasco, L. (2012). Learning manifolds with k-means and k-



- flats. In *Adv. Neural Info. Process. Syst.*, pp. 2465-2473.
- Candes, E. J., Eldar, Y. C., Needell, D., and Randall, P. (2011). *Compressed sensing with coherent and redundant dictionaries*. *App. Comput. Harmonic Anal.*, 31(1):59-73.
- Candes, E. J., Romberg, J., and Tao, T. (2006). *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*. *THEE Trans. Inf. Theory*, 52(2):489-509.
- Carson, C., Belongie, S., Greenspan, H., and Malik, J. (2002). *Blobworld: Image segmentation using expectation-maximization and its application to image querying*. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(8): 1026-1038.
- Celebi, M. E., Kingravi, H. A., and Vela, P. A. (2013). *A comparative study of efficient initialization methods for the k-means clustering algorithm*. *Expert Syst. Appl.*, 40(1):200-210.
- Cevher, V., Indyk, P., Hegde, C., and Baraniuk, R. G. (2009). *Recovery of clustered sparse signals from compressive measurements*. Technical report, Rice Univ.
- Cevikalp, H., Triggs, B., Yavuz, H. S., Kucuk, Y., Kucuk, M., and Barkana, A. (2010). *Large margin classifiers based on affine hulls*. *Neurocomput.*, 73(16):3160—3168.
- Chen, J. and Huo, X. (2006). *Theoretical results on sparse representations of multiple-measurement vectors*. *IEEE Trans. Signal Process.*, 54(12):4634—4643.
- Chen, S. S., Donoho, D. L., and Saunders, M. A. (1998). *Atomic decomposition by basis pursuit*. *SIAM J. Scientific Comput.*, 20(1):33-61.
- Chen, X., Song, Q., and Li, Z. (2017). *Design and analysis of quaternion-valued neural networks for associative memories*. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(12):2305-2314.
- Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. (2015). *The ucr time series classification archive*.
- Cheng, Y. (1995). *Mean shift, mode seeking, and clustering*. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):790-799.
- Cheung, Y. M. (2003). *k\*-means: A new generalized k-means clustering algorithm*. *Pattern Recog. Lett.*, 24(15):2883-2893.

- Choy, C., Gwak, J., and Savarese, S. (2019). 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3075-3084.
- Condat, L. (2016). *Fast projection onto the simplex and the  $l_1$  ball*. *Mathematical Programming*, 158(1-2):575-585.
- Cortes, C. and Vapnik, V. (1995). *Support-vector networks*. *Mach. Learning*, 20(3):273-297.
- Cotter, S. F., Rao, B. D., Engan, K., and Kreutz-Delgado, K. (2005). *Sparse solutions to linear inverse problems with multiple measurement vectors*. *IEEE Trans. Signal Process.*, 53(7):2477-2488.
- Dalai, N. and Triggs, B. (2005). *Histograms of oriented gradients for human detection*. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pp. 886-893. IEEE.
- Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. (2007). *Image demising by sparse 3-d transform-domain collaborative filtering*. *IEEE Trans. Image Process.*, 16(8):2080-2095.
- Davenport, M. A., Duarte, M. F., Eldar, Y. C., and Kutyniok, G. (2012). *Introduction to compressed sensing*. *Compressed Sensing: Theory and Applications*, Cambridge Univ. Press.
- Davis, G., Mallat, S., and Avellaneda, M. (1997). *Adaptive greedy approximations*. *Constructive Approx.*, 13(1):57-98.
- Demiriz, A., Bennett, K. P., and Embrechts, M. J. (1999). *Semi-supervised clustering using genetic algorithms*. *Artificial Neural Netw. Eng.*, pp. 809-814.
- Dheeru, D. and Taniskidou, E. K. (2017). *UCI Machine Learning Repository*, <http://archive.ics.uci.edu/ml>.
- Dhillon, I. S., Guan, Y., and Kulis, B. (2004). *Kernel k-means: spectral clustering and normalized cuts*. In *ACM Int. Conf. Knowl. Discovery Data Mining*, pp. 551-556.
- Dong, W., Li, X., Zhang, L., and Shi, G. (2011). *Sparsity-based image denoising via dictionary learning and structural clustering*. In *Proc. IEEE Comp. Vis. Pattern*

*Recog.*, pp. 457-464.

- Donoho, D. L. (2006).** *For most large underdetermined systems of linear equations the minimal  $l_1$ -norm solution is also the sparsest solution.* **Comm. Pure Applied Math.**, 59(6):797-829.
- Du, K. L. (2010).** *Clustering: A neural network approach.* **Neural Netw.**, 23(1):89-107.
- Duan, G., Wang, H., Liu, Z., Deng, J., and Chen, Y.-W. (2012).** **K-cpd: Learning of overcomplete dictionaries for tensor sparse coding.** In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 493—496. IEEE.
- Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. (2008).** **Efficient projections onto the  $l_1$ -ball for learning in high dimensions.** In *Int. Conf. Mach. Learn.*, pp. 272-279.
- Elad, M. (2010).** *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* 1st Edition. New York: Springer-Verlag.
- Elad, M. and Aharon, M. (2006).** *Image denoising via sparse and redundant representations over learned dictionaries.* **IEEE Trans. Image Process.**, 15(12):3736-3745.
- Elad, M., Figueiredo, M. A. T., and Ma, Y. (2010).** *On the role of sparse and redundant representations in image processing.* **Proc. IEEE**, 98(6):972-982.
- Eldar, Y. C. and Bolcskei, H. (2009).** **Block-sparsity: Coherence and efficient recovery.** In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2885-2888. IEEE.
- Eldar, Y. C., Kuppinger, P., and Bolcskei, H. (2010).** *Block-sparse signals: Uncertainty relations and efficient recovery.* **IEEE Trans. Signal Process.**, 58(6):3042—3054.
- Eldar, Y. C. and Mishali, M. (2009).** *Robust recovery of signals from a structured union of subspaces.* **IEEE Trans. Inf. Theory**, 55(11):5302—5316.
- Elhamifar, E. and Vidal, R. (2009).** **Sparse subspace clustering.** In *Proc. IEEE Comp. Vis. Pattern Recog.*, pp. 2790-2797.
- Elhamifar, E. and Vidal, R. (2013).** *Sparse subspace clustering: Algorithm, theory, and applications.* **IEEE Trans. Pattern Anal. Mach. Intell.**, 35(11):2765-2781.
- Engan, K., Aase, S. O., and Husoy, J. H. (1999).** **Method of optimal directions for frame**

- design. In *Proc. IEEE Acous. Speech Signal Process.*, volume 5, pp. 2443-2446.
- Engan, K., Skretting, K., and Husoy, J. H. (2007). *Family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation*. **Digit. Signal Process.**, 17(1):32-49.
- Ester, M., Kriegel, H. P., Sander, J., and Xu, X. (1996). **A density-based algorithm for discovering clusters in large spatial databases with noise**. In *Proc. Knowledge Disc. Data Mining*, volume 96, pp. 226-231.
- Everitt, B. S. and Skrondal, A. (2002). *The Cambridge dictionary of statistics*. **2nd Edition**. New York: Cambridge University Press.
- Eysenck, M. W. and Keane, M. T. (2005). *Cognitive psychology: A student's handbook*. **5th Edition**. Psychology Press.
- Fadili, M. J., Starck, J. L., and Murtagh, F. (2007). *Inpainting and zooming using sparse representations*. **Computer J.**, 52(1):64—79.
- Fang, Y., Wu, J., and Huang, B. (2012). *2d sparse signal recovery via 2d orthogonal matching pursuit*. **Science China Information Sciences**, 55(4):889-897.
- Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). *Efficient graph-based image segmentation*. **Int. J. Comp. Vis.**, 59(2):167-181.
- Fisher, R. (1953). **Dispersion on a sphere**. In *Proc. R. Soc. Lond. A*, volume 217, pp. 295-305.
- Fred, A. L. N. and Jain, A. K. (2002). **Data clustering using evidence accumulation**. In *Object Recog. Supported User Interact. Service Robots*, pp. 276-280.
- Friedman, J., Hastie, T., and Tibshirani, R. (2010). *A note on the group lasso and a sparse group lasso*. **arXiv preprint arXiv: 1001.0736**.
- Fritzke, B. (1995). **A growing neural gas network learns topologies**. In *Advances in neural information processing systems*, pp. 625-632.
- Garcia-Cardona, C. and Wohlberg, B. (2018). *Convolutional dictionary learning: A comparative review and new algorithms*. **IEEE Transactions on Computational Imaging**, 4(3):366-381.

- Girosi, F. (1998).** *An equivalence between sparse approximation and support vector machines.* **Neural Comput.**, 10(6): 1455-1480.
- Goldstein, M. and Dengel, A. (2012).** *Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm.* **KI-2012: Poster and Demo Track**, pp. 59-63.
- Golubitsky, O., Mazalov, V., and Watt, S. M. (2012).** *An algorithm to compute the distance from a point to a simplex.* **ACM Commun. Comput. Algebra**, 46:57-57.
- Gopal, S. and Yang, Y. (2014).** *Von mises-fisher clustering models.* In *Proc. Int. Conf. Machine Learn.*, pp. 154—162.
- Gordon, A. D. (1987).** *A review of hierarchical classification.* **J. R. Stat. Soc. Series A**, pp. 119-137.
- Gowda, K. C. and Diday, E. (1991).** *Symbolic clustering using a new dissimilarity measure.* **Pattern Recog.**, 24(6):567-578.
- Gowda, K. C. and Diday, E. (1992).** *Symbolic clustering using a new similarity measure.* **IEEE Trans, on Syst. Man Cybernetics**, 22(2):368-378.
- Gribonval, R., Jenatton, R., and Bach, F. (2015).** *Sparse and spurious: Dictionary learning with noise and outliers.* **IEEE Trans. Inf. Theory**, 61(11):6298—6319.
- Gribonval, R., Jenatton, R., Bach, F., Kleinstuber, M., and Seibert, M. (2015).** *Sample complexity of dictionary learning and other matrix factorizations.* **IEEE Trans. Inf. Theory**, 61(6):3469-3486.
- Gu, S., Meng, D., Zuo, W., and Zhang, L. (2017).** *Joint convolutional analysis and synthesis sparse representation for single image layer separation.* In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1708-1716.
- Haghighat, M., Zonouz, S., and Abdel-Mottaleb, M. (2015).** *Cloudid: Trustworthy cloud-based and cross-enterprise biometric identification.* **Expert Systems with Applications**, 42(21):7905-7916.
- Hardin, J. and Rocke, D. M. (2004).** *Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator.* **Computational Statistics & Data Analysis**, 44(4):625-638.
- Haugeland, J. (1989).** *Artificial intelligence: The very idea.* **Cambridge: MIT Press.**

- Hawkins, D. M. (2004). *The problem of overfitting*. **J. Chem. Inf. Comput. Sci.**, 44(1): 1-12.
- Hazan, T., Polak, S., and Shashua, A. (2005). Sparse image coding using a 3d non-negative tensor factorization. In *Tenth IEEE International Conference on Computer Vision*, volume 1, pp. 50-57. IEEE.
- He, Z., Xu, X., and Deng, S. (2003). *Discovering cluster-based local outliers*. **Pattern Recognition Letters**, 24(9-10): 1641-1650.
- Hershey, J. R., Chen, Z., Roux, J. L., and Watanabe, S. (2016). Deep clustering: Discriminative embeddings for segmentation and separation. In *Proc. IEEE Acous. Speech Signal Process.*, pp. 31-35.
- Hinton, G. and Salakhutdinov, R. R. (2006). *Reducing the dimensionality of data with neural networks*. **Science**, 313(5786):504—507.
- Hirose, A. (2012). *Complex-valued neural networks*, volume 400. 2nd Edition. Berlin Heidelberg: Springer-Verlag.
- Ho, Y.-C. and Pepyne, D. L. (2002). Simple explanation of the no-free-lunch theorem and its implications. **Journal of optimization theory and applications**, 115(3):549-570.
- Hore, P., Hall, L. O., and Goldgof, D. B. (2009). A scalable framework for cluster ensembles. **Pattern Recog.**, 42(5):676-688.
- Huang, D. A., Kang, L. W., Wang, Y. C. F., and Lin, C. W. (2014). Self-learning based image decomposition with applications to single image demising. **IEEE Trans. Multimedia**, 16(1):83-93.
- Huang, F. and Anandkumar, A. (2015). Convolutional dictionary learning through tensor factorization. In *Feature Extraction: Modern Questions and Challenges*, pp. 116—129.
- Huang, J., Nie, F., and Huang, H. (2015). A new simplex sparse learning model to measure data similarity for clustering. In *Int. Joint Conf. Artif. Intell.*, pp. 3569-3575.
- Huang, J. and Zhang, T. (2010). The benefit of group sparsity. **The Annals of Statistics**, 38(4): 1978-2004.
- Huang, J., Zhang, T., and Metaxas, D. (2011). *Learning with structured sparsity*. **J. Mach.**

- Learning Res., 12:3371-3412.
- Hull, J. J. (1994). *A database for handwritten text recognition research*. **IEEE Trans. Patt. Anal. Mach. Intell.**, 16(5):550-554.
- Iam-on, N. and Garrett, S. (2010). *Linkclue: A matlab package for link-based cluster ensembles*. **J. Stat. Software**, 36(9): 1-36.
- Inouye, D. I., Yang, E., Allen, G. I., and Ravikumar, P. (2017). *A review of multivariate distributions for count data derived from the poisson distribution*. **Wiley Interdisciplinary Reviews: Comput. Stat.**, 9(3):e1398.
- Isokawa, T., Kusakabe, T., Matsui, N., and Peper, F. (2003). *Quaternion neural network and its application*. In *International conference on knowledge-based and intelligent information and engineering systems*, pp. 318-324. Springer.
- Jacob, L., Obozinski, G., and Vert, J.-P. (2009). *Group lasso with overlap and graph lasso*. In *Int. Conf. Mach. Learn.*, pp. 433—440.
- Jafari, M. and Molaei, H. (2014). *Spherical linear interpolation and bezier curves*. **General Scientific Researches**, 2(1): 13-17.
- Jain, A. K. (2010). *Data clustering: 50 years beyond k-means*. **Pattern recognition letters**, 31(8):651-666.
- Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. (2011). *Proximal methods for hierarchical sparse coding*. **J. Mach. Learning Res.**, 12:2297-2334.
- Jiang, Z., Lin, Z., and Davis, L. S. (2013). *Label consistent K-SVD: Learning a discriminative dictionary for recognition*. **IEEE Trans, on Patt. Anal. Mach. Intell.**, 35(11):2651-2664.
- Jolliffe, I. (2002). *Principal component analysis*. 2nd Edition. New York: Springer-Verlag.
- Jost, P., Vanderghenst, P., Lesage, S., and Gribonval, R. (2006). *MoTIF: An efficient algorithm for learning translation invariant dictionaries*. In *Proc. IEEE Acous. Speech Signal Process.*, volume 5, pp. 857-860.
- Juszczak, P., Tax, D. M. J., Pe-kalska, E., and Duin, R. P. W. (2009). *Minimum spanning tree based one-class classifier*. **Neurocomput.**, 72(7-9): 1859-1869.

- Kachuee, M., Fazeli, S., and Sarrafzadeh, M. (2018). Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 443—444. IEEE.
- Khan, S. S. and Madden, M. G. (2014). *One-class classification: Taxonomy of study and review of techniques*. *The Know. Eng. Review*, 29(3):345-374.
- Kilmer, M. E. and Martin, C. D. (2011). *Factorization strategies for third-order tensors*. *Linear Algebra and its Applications*, 435(3):641-658.
- Kiranyaz, S., Ince, T., and Gabbouj, M. (2015). *Real-time patient-specific ecg classification by 1-d convolutional neural networks*. *IEEE Transactions on Biomedical Engineering*, 63(3):664-675.
- Kiselev, V. Y., Andrews, T. S., and Hemberg, M. (2019). *Challenges in unsupervised clustering of single-cell ma-seq data*. *Nature Reviews Genetics*, 20(5):273-282.
- Kolda, T. G. and Bader, B. W. (2009). *Tensor decompositions and applications*. *SIAM review*, 51(3):455-500.
- Kong, S. and Wang, D. (2012). A dictionary learning approach for classification: separating the particularity and the commonality. In *European conference on computer vision*, pp. 186-199. Springer.
- Kriegel, H. P., Kroger, P., Sander, J., and Zimek, A. (2011). *Density-based clustering*. *Data Mining and Knowledge Discovery*, 1(3):231-240.
- Kriegel, H.-P., Schubert, M., and Zimek, A. (2008). Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 444—452. ACM.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Adv. Neural Info. Process. Sys.*, pp. 1097-1105.
- Lazarevic, A. and Kumar, Y. (2005). Feature bagging for outlier detection. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pp. 157-166. ACM.
- Lazendic, S., De Bie, H., and Pizurica, A. (2018). Octonion sparse representation for



- color and multispectral image processing. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 608-612. IEEE.
- Lazendic, S., Pizurica, A., and De Bie, H. (2018). Hypercomplex algebras for dictionary learning. In *7th Conference on Applied Geometric Algebras in Computer Science and Engineering-AGACSE 2018*, pp. 57-64. Unicamp/IMECC.
- LeCun, Y., Cortes, C., and Burges, C. J. C. (2010). *MNIST Handwritten Digit Database*.
- Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. (2017). *Deep neural networks as gaussian processes*. arXiv preprint arXiv:1711.00165.
- Lesage, S., Gribonval, R., Bimbot, F., and Benaroya, L. (2005). Learning unions of orthonormal bases with thresholded singular value decomposition. In *Proc. IEEE Acous. Speech Signal Process.*, volume 5, pp. 293-296.
- Li, H.-C., Song, M., and Chang, C.-I. (2015). Simplex volume analysis for finding endmembers in hyperspectral imagery. In *Satellite Data Comp. Commun. Process. XI*, volume 9501, p. 950107.
- Liao, H. Y. and Sapiro, G. (2008). Sparse representations for limited data tomography. In *Proc. IEEE Int. Symp. Biomed. Imag.*, pp. 1375—1378.
- Lin, T. and Zha, H. (2008). *Riemannian manifold learning*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):796-809.
- Lindemann, P. (2009). *The Gilbert-Johnson-Keerthi distance algorithm*. *Alg. Media Informatics*.
- Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413—422. IEEE.
- Lloyd, S. (1982). *Least squares quantization in PCM*. *IEEE Trans. Inf. Theory*, 28(2): 129— 137.
- Lu, H., Plataniotis, K. N., and Venetsanopoulos, A. N. (2011). *A survey of multilinear subspace learning for tensor data*. *Pattern Recognition*, 44(7):1540-1551.
- Luo, C., Ma, C., Wang, C., and Wang, Y. (2017). Learning discriminative activated simplices for action recognition. In *AAAI Conf. Artif. Intell.*, pp. 4211-4217.

- Maesschalck, R. D., Rimbaud, D. J., and Massart, D. L. (2000).** *The mahalanobis distance*. *Chemometrics Intelligent Lab. Syst.*, 50(1):1-18.
- Mairal, J., Bach, F., and Ponce, J. (2011).** *Task-driven dictionary learning*. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):791-804.
- Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2010).** *Online learning for matrix factorization and sparse coding*. *J. Mach. Learning Res.*, 11(1): 19-60.
- Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. (2008a).** **Discriminative learned dictionaries for local image analysis.** In *Proc. IEEE Comp. Vis. Pattern Recog.*, pp. 1-8.
- Mairal, J., Elad, M., and Sapiro, G. (2008b).** *Sparse representation for color image restoration*. *IEEE Trans. Image Process.*, 17(1):53-69.
- Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., and Bach, F. R. (2009).** **Supervised dictionary learning.** In *Adv. Neural Inf. Process. Syst.*, pp. 1033-1040.
- Mairal, J., Sapiro, G., and Elad, M. (2008c).** *Learning multiscale sparse representations for image and video restoration*. *SIAM Multiscale Model. Simul.*, 7(1):214—241.
- Mallat, S. and Zhang, Z. (1993).** *Matching pursuit with time-frequency dictionaries*. *IEEE Trans. Signal Process.*, 41(12):3397-3415.
- Mardia, K. V. (2014).** *Statistics of directional data*. Academic Press.
- Mardia, K. V. and Jupp, P. E. (2009).** *Directional statistics*, volume 494. Wiley.
- MathWorks (2019).** *6 functions for generating artificial datasets - File Exchange - MATLAB Central*. [Online]. Available at <https://www.mathworks.com/matlabcentral/fileexchange/41459> (Accessed 3 July 2019).
- Mayiami, M. R. and Seyfe, B. (2012).** *Nonparametric sparse representation*. arXiv preprint arXiv: 1201.2843.
- Meier, L., Geer, S. V. D., and Buhlmann, P. (2008).** *The group lasso for logistic regression*. *J. R. Stat. Soc. Series B*, 70(1):53—71.
- Michelucci, D. and Foufou, S. (2003).** *Using cayley menger determinants*. In *Proceedings of the Workshop on Geometric Constraint Solving*. Citeseer.
- Monaci, G., Jost, P., and Vandergheynst, P. (2004).** *Image compression with learnt tree-*

- structured dictionaries. In *Proc. IEEE W. Mult. Signal Process.*, pp. 35-38.**
- Moody, G. B. and Mark, R. G. (2001).** *The impact of the mit-bih arrhythmia database.* **IEEE Engineering in Medicine and Biology Magazine**, 20(3):45-50.
- Moxey, C. E., Sangwine, S. J., and Ell, T. A. (2003).** *Hypercomplex correlation techniques for vector images.* **IEEE Transactions on Signal Processing**, 51(7):1941-1953.
- Moya, M. M. and Hush, D. R. (1996).** *Network constraints and multi-objective optimization for one-class classification.* **Neural Networks**, 9(3):463—474.
- Nakashizuka, M., Nishiura, H., and Iiguni, Y. (2009).** **Sparse image representations with shift-invariant tree-structured dictionaries. In *Proc. IEEE Int. Conf. Image Process.*, pp. 2145-2148.**
- Nejati, M., Samavi, S., Derksen, H., and Najarian, K. (2016).** *Denoising by low-rank and sparse representations.* **Journal of Visual Communication and Image Representation**,
- Nesterov, Y. and Nemirovskii, A. (1994).** *Interior-point polynomial algorithms in convex programming.* **SIAM.**
- Nguyen, D. K., Than, K., and Ho, T. B. (2013).** **Simplicial nonnegative matrix factorization. In *Int. Conf. Comput. Commun. Tech.-Res. Innov. Vis. Fut.*, pp. 47—52.**
- Nitta, T. (2003).** *Solving the xor problem and the detection of symmetry using a single complex-valued neuron.* **Neural Networks**, 16(8): 1101—1105.
- Novak, P., Neumann, P., and Macas, J. (2010).** *Graph-based clustering and characterization of repetitive sequences in next-generation sequencing data.* **BMC Bioinformatics**, 11(1):378.
- Ojala, T., Pietikainen, M., and Harwood, D. (1996).** *A comparative study of texture measures with classification based on featured distributions.* **Pattern recognition**, 29(1):51— 59.
- Panagakis, Y., Kotropoulos, C., and Arce, G. R. (2009).** **Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *ISMIR*, pp. 249-254.**
- Parsons, L., Haque, E., and Liu, H. (2004).** *Subspace clustering for high dimensional*

- data: a review*. **ACM Sigkdd Explorations**, 6(1):90-105.
- Patania, A., Vaccarino, F., and Petri, G. (2017).** *Topological analysis of data*. **EPJ Data Sci.**, 6(1):7.
- Pati, Y. C., Rezaiifar, R., and Krishnaprasad, P. S. (1993).** Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proc. Asimolar Conf. Sig. Sys. Compt.*, pp. 40—44.
- Pearson, K. (1901).** *Liii. on lines and planes of closest fit to systems of points in space*. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11):559-572.
- Pelleg, D. and Moore, A. W. (2000).** X-means: Extending k-means with efficient estimation of the number of clusters. In *Int. Conf. Mach. Learn.*, pp. 727-734.
- Peng, Y., Meng, D., Xu, Z., Gao, C., Yang, Y., and Zhang, B. (2014).** Decomposable nonlocal tensor dictionary learning for multispectral image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2949— 2956.
- Peotta, L., Granai, L., and Vandergheynst, P. (2006).** Image compression using an edge adapted redundant dictionary and wavelets. **Signal Process.**, 86(3):444—456.
- Peyre, G. (2009).** *Sparse modeling of textures*. **J. Math. Imag. Vis.**, 34(1): 17-31.
- Popa, C.-A. (2016).** Octonion-valued neural networks. In *International Conference on Artificial Neural Networks*, pp. 435—443. Springer.
- Protter, M. and Elad, M. (2009).** Image sequence denoising via sparse and redundant representations. **IEEE Trans. Image Process.**, 18(1):27-35.
- Pu, Y., Yuan, W., Stevens, A., Li, C., and Carin, L. (2016).** A deep generative deconvolutional image model. In *Artificial Intelligence and Statistics*, pp. 741-750.
- Qi, N., Shi, Y., Sun, X., Wang, J., and Yin, B. (2013).** Two dimensional synthesis sparse model. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1-6. IEEE.

- Quinlan, J. R. (1986). *Induction of decision trees*. *Mach. Learning*, 1(1):81—106.
- Ramaswamy, S., Rastogi, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *ACM Sigmod Record*, volume 29, pp. 427-438. ACM.
- Ramirez, I., Sprechmann, P., and Sapiro, G. (2010). Classification and clustering via dictionary learning with structured incoherence and shared features. In *Proc. IEEE Comp. Vis. Pattern Recog.*, pp. 3501-3508.
- Rayana, S. (2016). *ODDS library*. [Online]. Available at <http://odds.cs.stonybrook.edu> (Accessed 3 July 2019).
- Reynolds, D. (2015). *Gaussian mixture models*. *Encyclopedia Biometrics*, pp. 827-832.
- Roemer, F., Del Galdo, G., and Haardt, M. (2014). Tensor-based algorithms for learning multidimensional separable dictionaries. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3963-3967. IEEE.
- Rosenblatt, F. (1958). *The perceptron: A probabilistic model for information storage and organization in the brain*. *Psychological review*, 65(6):386.
- Rosenblatt, F. (1961). Principles of neurodynamics, perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY.
- Rubinstein, R., Bruckstein, A. M., and Elad, M. (2010a). *Dictionaries for sparse representation modeling*. *Proc. IEEE*, 98(6): 1045-1057.
- Rubinstein, R., Zibulevsky, M., and Elad, M. (2010b). *Double sparsity: Learning sparse dictionaries for sparse signal approximation*. *IEEE Trans. Signal Process.*, 58(3): 1553-1564.
- Sak, H., Senior, A., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth Annual Conference of the International Speech Communication Association*.
- Sallee, P. and Olshausen, B. A. (2003). Learning sparse multiscale image representations. In *Adv. Neural Inf. Process. Syst.*, volume 15, pp. 1327-1334.
- Sandor, J. (1996). *On the arithmetical functions  $d \sim k(n)$  and  $d^{**} \sim k(n)$* . *Portugaliae*

**Mathematica**, 53:107-116.

**Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2008).** *The graph neural network model.* **IEEE Transactions on Neural Networks**, 20(1):61-80.

**Schmidhuber, J. (2015).** *Deep learning in neural networks: An overview.* **Neural Netw.**, 61:85-117.

**Scholkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001).** *Estimating the support of a high-dimensional distribution.* **Neural computation**, 13(7):1443-1471.

**Scholkopf, B., Smola, A., and Muller, K. R. (1998).** *Nonlinear component analysis as a kernel eigenvalue problem.* **Neural Comput.**, 10(5):1299-1319.

**Schuster, M. and Paliwal, K. K. (1997).** *Bidirectional recurrent neural networks.* **IEEE transactions on Signal Processing**, 45(11):2673-2681.

**Schwikowski, B., Uetz, P., and Fields, S. (2000).** *A network of protein-protein interactions in yeast.* **Nature Biotech.**, 18(12):1257.

**Sezer, O. G., Harmanci, O., and Guleryuz, O. G. (2008).** *Sparse orthonormal transforms for image compression.* In *Proc. IEEE Int. Conf. Image Process.*, pp. 149-152.

**Shao, L., Yan, R., Li, X., and Liu, Y. (2014).** *From heuristic optimization to dictionary learning: A review and comprehensive comparison of image demising algorithms.* **IEEE Trans. Cybernetics**, 44(7): 1001-1013.

**Shekhar, S., Patel, V. M., and Chellappa, R. (2014).** *Analysis sparse coding models for image-based classification.* In *2014 IEEE international conference on image processing (ICIP)*, pp. 5207-5211. **IEEE**.

**Sibson, R. (1973).** *SLINK: An optimally efficient algorithm for the single-link cluster method.* **The Comp. J.**, 16(1):30—34.

**Silva, J. and Willett, R. (2008).** *Hypergraph-based anomaly detection of high-dimensional co-occurrences.* **IEEE Trans. Patt. Anal. Mach. Intell.**, (3):563-569.

**Sivalingam, R., Boley, D., Morellas, V., and Papanikolopoulos, N. (2010).** *Tensor*

- sparse coding for region covariances. In *European conference on computer vision*, pp. 722-735. Springer.
- Skretting, K. and Engan, K. (2010). *Recursive least squares dictionary learning algorithm*. *IEEE Trans. Signal Process.*, 58(4):2121-2130.
- Song, J., Xie, X., Shi, G., and Dong, W. (2019). *Multi-layer discriminative dictionary learning with locality constraint for image classification*. *Pattern Recognition*, 91:135—146.
- Sprechmann, P. and Sapiro, G. (2010). **Dictionary learning and sparse coding for unsupervised clustering**. In *2010 IEEE international conference on acoustics, speech and signal processing*, pp. 2042-2045. IEEE.
- Steinwart, I. and Christmann, A. (2008). *Support Vector Machines*. 1st Edition. New York: Springer-Verlag.
- Stojanovic, V. and Nedic, N. (2016). *Identification of time-varying OE models in presence of non-Gaussian noise: Application to pneumatic servo drives*. *Int. J. Robust and Nonlinear Control*, 26(18):3974—3995.
- Stojanovic, V., Nedic, N., Prsic, D., and Dubonjic, L. (2016). *Optimal experiment design for identification of ARK models with constrained output in non-Gaussian noise*. *Elsevier App. Mathematical Modell.*, 40(13):6676-6689.
- Szlam, A. and Sapiro, G. (2009). **Discriminative k-metrics**. In *Int. Conf Mach. Learn.*, pp.1009-1016.
- Tasaki, H., Lenz, R., and Chao, J. (2016). **Simplex-based dimension estimation of topological manifolds**. In *Int. Conf. Patt. Recog.*, pp. 3609-3614.
- Tibshirani, R. (1996). *Regression shrinkage and selection via the lasso*. *J. R. Stat. Soc. Series B*, pp. 267-288.
- Tillmann, A. M. (2015). *On the computational intractability of exact and approximate dictionary learning*. *IEEE Signal Process. Lett.*, 22(1):45—49.
- Tseng, P. (2000). *Nearest  $q$ -flat to  $m$  points*. *J. Optimization Theory App.*, 105(1):249-252.
- Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). *Large margin*

- methods for structured and interdependent output variables. J. Mach. Learning Res.*, 6(Sep): 1453-1484.
- Tung, H.-Y., Tung, H.-W., Yumer, E., and Fragkiadaki, K. (2017). Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pp. 5236-5246.
- Tuzel, O., Porikli, F., and Meer, P. (2006). Region covariance: A fast descriptor for detection and classification. In *European conference on computer vision*, pp. 589-600. Springer.
- van der Maaten, L. and Hinton, G. (2008). *Visualizing data using t-SNE. J. Mach. Learn. Res.*, 9:2579-2605.
- Vu, T. H. and Monga, V. (2016). Learning a low-rank shared dictionary for object classification. In *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 4428—4432. IEEE.
- Vu, T. H. and Monga, V. (2017). *Fast low-rank shared dictionary learning for image classification. IEEE Transactions on Image Processing*, 26(11):5160-5175.
- Wang, C., Flynn, J., Wang, Y., and Yuille, A. (2016). Recognizing actions in 3D using action-snippets and activated simplices. In *AAAI Conf. Artif. Intell.*, pp. 3604—3610.
- Wang, J., Li, J., Han, X.-H., Lin, L., Hu, H., Xu, Y., Chen, Q., Iwamoto, Y., and Chen, Y.-W. (2020). *Tensor-based sparse representations of multi-phase medical images for classification of focal liver lesions. Pattern Recognition Letters*, 130:207-215.
- Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. (2010). Locality-constrained linear coding for image classification. In *IEEE Comp. Vis. Pattern Recog.*, pp. 3360- 3367.
- Wang, R., Wang, K., Cao, W., and Wang, X. (2019). *Geometric algebra in signal and image processing: A survey. IEEE Access*, 7:156315-156325.
- Wei, C.-P., Chao, Y.-W., Yeh, Y.-R., and Wang, Y.-C. F. (2013). *Locality-sensitive dictionary learning for sparse representation based classification. Pattern Recog.*,



46(5): 1277-1287.

- Wei, L., Qian, W., Zhou, A., Jin, W., and Jeffrey, X. Y. (2003). Hot: Hypergraph-based outlier test for categorical data. In *Pacific-Asia Conf. Know. Discov. Data Mining*, pp. 399-410.
- Weng, Y., Zhang, N., and Xia, C. (2018). Multi-agent-based unsupervised detection of energy consumption anomalies on smart campus. *IEEE Access*, 7:2169—2178.
- Wohlberg, B. (2017). Sporco: A python package for standard and convolutional sparse representations. In *Proceedings of the 15th Python in Science Conference, Austin, TX, USA*, pp. 1-8.
- Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67-82.
- Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. (2008). Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210-227.
- Xu, L., Neufeld, J., Larson, B., and Schuurmans, D. (2004). Maximum margin clustering. In *Proc. Adv. Neural Inf. Process. Sys.*, pp. 1537-1544.
- Xu, Y., Yu, L., Xu, H., Zhang, H., and Nguyen, T. (2015). Vector sparse representation of color image using quaternion matrix analysis. *IEEE Transactions on image processing*, 24(4):1315-1329.
- Yan, S., Xu, D., Zhang, B., Zhang, H., Yang, Q., and Lin, S. (2007). Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(1):40-51.
- Yang, M., Zhang, L., Feng, X., and Zhang, D. (2011). Fisher discrimination dictionary learning for sparse representation. In *2011 International Conference on Computer Vision*, pp. 543-550. IEEE.
- Yu, K., Zhang, T., and Gong, Y. (2009). Nonlinear learning using local coordinate coding. In *Adv. Neural Info. Process. Syst.*, pp. 2223-2231.
- Yu, L., Sun, H., Barbot, J. P., and Zheng, G. (2012). Bayesian compressive sensing for cluster structured sparse signals. *Elsevier Signal Process.*, 92(1):259-269.

- Yuan, M. and Lin, Y. (2006).** *Model selection and estimation in regression with grouped variables.* **J. Royal Stat. Soc. B**, 68(1):49-67.
- Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. (2010).** **Deconvolutional networks.** In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pp. 2528-2535. IEEE.
- Zepeda, J. (2010).** *Novel sparse representation methods; application to compression and indexation of images.* PhD thesis, INRIA, France.
- Zepeda, J., Guillemot, C., and Kijak, E. (2011).** *Image compression using sparse representations and the iteration-tuned and aligned dictionary.* **IEEE J. Selected Topics Signal Process.**, 5(5): 1061-1073.
- Zhang, K., Tsang, I. W., and Kwok, J. T. (2009).** *Maximum margin clustering made practical.* **IEEE Trans. Neural Netw.**, 20(4):583-596.
- Zhang, Z., Xu, Y., Yang, J., Li, X., and Zhang, D. (2015).** *A survey of sparse representation: algorithms and applications.* **IEEE Access**, 3:490-530.
- Zhao, Y., Nasrullah, Z., and Li, Z. (2019).** *Pyod: A python toolbox for scalable outlier detection.* **Journal of Machine Learning Research**, 20:1-7.
- Zhong, S. (2005).** **Efficient online spherical k-means clustering.** In *Proc. IEEE Neural Netw.*, volume 5, pp. 3180-3185.
- Zhu, K. and Vogel-Heuser, B. (2014).** *Sparse representation and its applications in micro- milling condition monitoring: noise separation and tool condition monitoring.* **Int. J. Adv. Manuf. Technol.**, 70(1): 185-199.
- Zhu, X. and Goldberg, A. B. (2009).** *Introduction to semi-supervised learning.* **Synthesis lectures on artificial intelligence and machine learning**, 3(1): 1-130.
- Zhuang, L. and Bioucas-Dias, J. M. (2018).** *Fast hyperspectral image demixing and inpainting based on low-rank and sparse representations.* **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, 11(3):730-742.

## APPENDIX A: RESIDUAL CODES

### A.1 Introduction

A usual deficiency in sparse coding step is that, algorithms listed above assume proper dictionaries at each iteration. This is indeed very problematic, especially at initial stages of the learning process. In many situations, initial dictionary will not be a good representative of the optimal one. Therefore, “optimal” coding done with such a dictionary, as targeted by both  $\ell_0$  and  $\ell_1$  norm coding schemes, will most likely result in sparse codes, which also are not good representatives of optimal state. As a result, the next dictionary will adopt this undesired property to a certain extent and convey it to successive learning steps. In this appendix, we propose a generic modification to sparse coding or the coefficient learning step, with an error correcting process by coding an intermediate error and adjusting sparse codes accordingly in a less intensive learning attempt, hence leading to a faster convergence when compared to the conventional approaches.

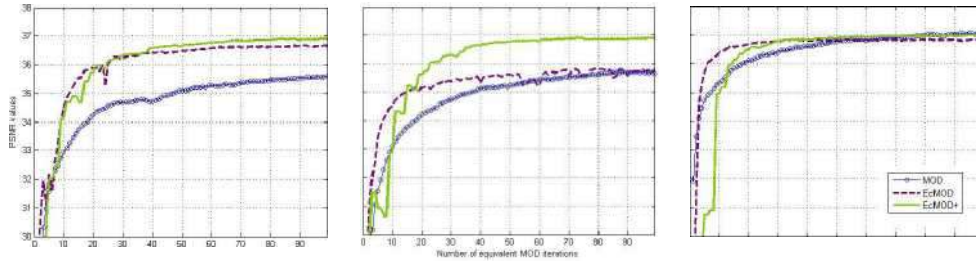
This appendix is organized as follows. Section A.2 describes the details of the proposed dictionary learning algorithm. Section A.3 demonstrates the experimental setup and the obtained results. Finally, Section A.4 concludes this appendix.

### A.2 Introducing Error Codes

We propose a formulation that incorporates an intermediate error into the learning process. In the first stage, a regular sparse coding and dictionary update procedure is performed but with a sparsity level  $m < k$  by solving Eqn. 36.

$$\underset{D, \{\mathbf{a}_i\}_{i=1}^M}{\operatorname{argmin}} \|\mathbf{y} - D\mathbf{a}\|_2 \text{ subject to } \|\mathbf{a}\|_0 \leq m \quad (36)$$

Let us now denote  $\{\mathbf{a}^*\}$  and  $D^*$  as the resulting sparse codes and the dictionary, respectively. The second stage involves sparse coding the approximation error  $\mathbf{e}^* = \mathbf{y} - D^*\mathbf{a}^*$ ,



**Figure 27. Results of dictionary learning performed on the *Barbara* image. Dictionary size  $64 \times 256$ ,  $k = 8$  for both learning and testing. Each column represents a different initial dictionary. First and middle columns are randomly initialized. Last column is of DCT initialization. Figures depict PSNR (dB) performance values versus iteration number.**

Vi, as

$$\underset{\mathbf{S}'}{\operatorname{argmin}} \|\mathbf{e}_j - \mathbf{D}^* \mathbf{b}_j\|_2^2 \text{ s.t. } \|\mathbf{b}_j\|_0 \leq m. \quad (37)$$

After acquiring  $\{\mathbf{b}^*\}$  in Eqn. 37, current-state sparse codes can further be updated as  $\mathbf{a}^* + \mathbf{b}^*$ . This step basically corresponds to some sort of feedback logic, where the first approximation is tested and then its deviation is sparse coded to be incorporated into actual codes. Note here that the original sparsity constraint still holds since  $\|\mathbf{a}^* + \mathbf{b}^*\|_0 \leq k$ . In the last stage, a final dictionary update is performed as in Eqn. 38 and an iteration is completed,

$$\underset{\mathbf{D}}{\operatorname{argmin}} \sum_{i=1}^M \|\mathbf{y}_i - \mathbf{D}(\mathbf{a}^* + \mathbf{b}^*)\|^2. \quad (38)$$

### A.3 Experimental results

Two variants of the proposed scheme have been tested experimentally, namely EcMOD in Algorithm 3 and EcMOD+ in Algorithm 4. EcMOD includes the methodology that is defined in Sect. A.2, and EcMOD-i- includes a regular least-squares dictionary update (MOD) at the end of each iteration. OMP is used for sparse coding.

Two experimental setups have been performed, corresponding to low and high

---

**Algorithm 3** EcMOD algorithm pseudocode.

---

```

1 function EcMOD( $D, Y, m, k$ )
2   while not converged do
3      $A \leftarrow \text{OMP}(D, Y, m)$ 
4      $D \leftarrow Y A^+$ 
5      $E Y \leftarrow D A$ 
6      $B \leftarrow \text{OMP}(D, E, k - \text{to})$ 
7      $D \leftarrow Y(A + B)^+$ 
  return  $D$ 

```

---

**Algorithm 4** EcMOD-i- algorithm pseudocode.

---

```

  function EcMOD+( $D, Y, m, k$ )
2   while not converged do
       $D \leftarrow \text{EcMOD}(D, Y, m, k)$ 
4    $X \leftarrow \text{OMP}(D, Y, k)$ 
       $D \leftarrow Y X^+ \text{ return } D$ 

```

---

dimensional cases respectively. In the first setup,  $8 \times 8$  distinct patches were extracted from the *Barbara* image of size  $512 \times 512$ , resulting in 4096 image patches. Dictionary size was accordingly chosen as  $64 \times 256$ . Sparsity constraint  $k$  and additional sparsity parameter  $m$  were chosen as 8 and 4 respectively, so  $m$  being equal to  $k/2$ . Results corresponding to this setup are presented in Figure 27, Figure 28 and Table 10.

In error coded schemes, as a consequence of not directly coding with sparsity  $k$ , final codes  $a^* + b$  may not necessarily be optimal for  $k$ . However, as there are two coding steps with lesser sparsity constraints and a summation, codes have a higher chance of being optimal for most of the sparsity levels less than  $k$ . As a result, converged dictionary is a better representative of such sparsity levels, as observable in the results in Table 10. Error coded schemes consistently perform better in sparser cases.

Not targeting a sparsity level  $k$  directly leads to a possibility of converged dictionary to be suboptimal for that given  $k$ . However, this drawback can be worked around by chaining a conventional step that targets an exact sparsity level  $k$ . Referred to as EcMOD+ algorithm, experiments with this further modified method show that, such architecture possesses optimality for  $k$  sparsity and also better performance for the cases where sparsity level is less than  $k$ . This phenomenon is apparent in Figure 27, where learning and testing  $k$  chosen both as 8. Performance of

EcMOD is not consistent as it performs much like MOD in second random initialization case, whereas

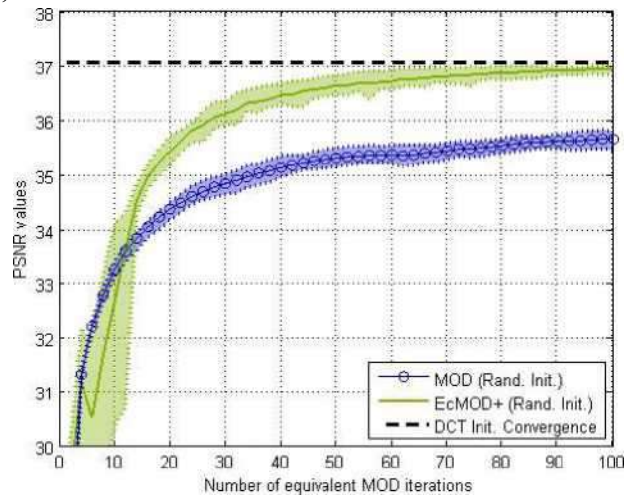


Figure 28. Results for ten cases of uniformly random initial dictionaries. Lower and upper lines of each method correspond to minimum and maximum PSNR (dB) values attained among all ten cases. Middle lines represent average values.

EcMOD+ consistently performs well.

In a more extensive manner, Figure 28 compares the performance of MOD and EcMOD+ in the case of ten different randomly initialized dictionaries. DCT convergence is supplied as a baseline. This figure represents superiority of the error coding scheme over conventional coding in the case of random initializations. “Optimal” coding with an improper random dictionary within initial stages hamper the final convergence state as observable in the case of MOD. Although not targeting optimal codes, error coded scheme EcMOD-i- converges to DCT result in all random initialization cases. This is possible because, in each step from the beginning, dictionary passes through a less intensive validation, in the expense of acquiring optimal codes. There is no total superiority in DCT initialization case as seen in Table 10. However, superiority can be achieved with more complex error coding schemes.

Finally, regarding overall sparsity levels within error codes and its evolution

during learning process, the proposed method presents interesting trends. In conventional sparse coding (targeting the sparsity  $k$ ), as approximation threshold is kept very

Table 10. Average approximation PSNR (dB) performance values of learnt dictionaries as in Figure 27 and Figure 28.  $k$  represents the sparsity used for testing.

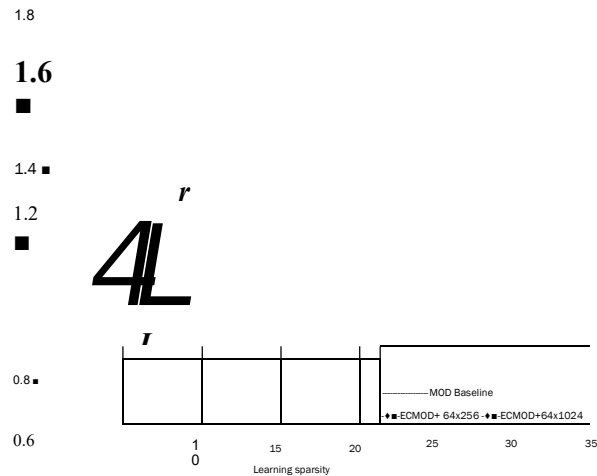
	$k = 2$	$k = 5$	$k = 10$	$k = 20$
<b>Rand. 1.</b>				
<b>MOD</b>	26.33	32.08	36.82	40.43
<b>EcMOD</b>	26.91	33.90	37.87	41.72
<b>EcMOD+</b>	26.89	33.90	38.10	41.86
<b>Rand. 2.</b>				
<b>MOD</b>	26.61	32.20	36.89	40.68
<b>EcMOD</b>	27.92	33.36	36.88	40.25
<b>EcMOD+</b>	26.73	33.87	38.08	41.83
<b>DCT Init.</b>				
<b>MOD</b>	26.22	33.02	38.40	42.82
<b>EcMOD</b>	26.95	34.05	38.06	41.98
<b>EcMOD+</b>	26.94	33.96	38.19	42.07

strict in general, sparse codes end up with using all  $k$  supports. Therefore, in methods such as MOD and K-SYD, codes consistently have  $k$  supports even starting from the initial iteration. In the proposed method, during error coding, selection of previously selected supports is frequent. This is especially observable during initial iterations. Near maximum support counts are gradually reached as the system converges, but not necessarily reaching exact maximum.

In the second set of experiments, EcMOD+ scheme has been tested with all possible 255025 image patches extracted with a full coverage of sliding window algorithm with a window size of  $8 \times 8$ . Combinations of sparsity of 4, 8, 16, and 32 against small and large dictionaries were tested. DCT was used as initial dictionary in all cases. Note that, in DCT initialization cases, there is only an advantage of faster convergence rates but not of better converged states, at least for this error coding scheme. Superior converged states with more complex schemes have been achieved, but they are omitted here because of the space limitation.

**Experimental results with the second setup are summarized in Figure 29 and Figure 30. In Figure 29, performance ratios were calculated relative to the MOD**



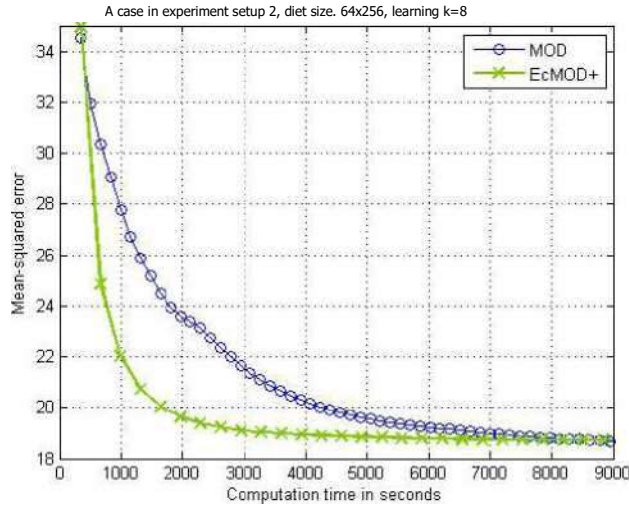


**Figure 29. Performance gain factor for different sparsity levels, in the case of a relatively small and a large dictionary.**

algorithm in terms of mean-squared error to estimate a performance gain factor for each sparsity level, approximately at fifth equivalent iteration, for an approximate convergence rate analysis. Gains for large dictionary in the case of learning with lenient sparsity levels are more striking, but stricter sparsity constraints cause substandard performance. Overall, the performance in this case is promising as it signals to scaling with input size. Smaller sized dictionary safely performs above standard. Figure 30 depicts the convergence plot for dictionary size 64 x 256 and  $\epsilon$  as 8 for both learning and testing. Significant gain in convergence rate is observable with the error coded scheme in this high dimensional setup. Note that, mean-squared error is given as measure since sliding window patches were used. Finally, Figure 31 compares atoms that lie within similar frequency domains, learnt with MOD and EcMOD+. Note here the well-defined structure of EcMOD+.

#### *A.4 Conclusion*

MOD, by itself is a greedy algorithm that targets optimality one task at a time. Tasks are considered as isolated from each other, even within the same iteration. This



**Figure 30.** The convergence performance for dictionary size 64 x 256 and A; as 8 for both learning and testing.

results in MOD being a rather short-sighted method which fails at tasks that require a broader perspective of the system.

In this paper, we presented a method in which sparse coding and dictionary update steps are intertwined through intermediate error codes. Note that there could be other ways to accomplish this. Another way could be to add sparse codes of two successive iterations and perform a dictionary update based on this accumulated code, without even introducing error codes. As an analogy, MOD can be considered as a single-step numerical method, whereas the example given would be a multi-step one. Our method can be considered as a multi-step approach that utilizes a half-step.

To summarize, our framework is generic enough to be included in many forms of learning-based approaches. In essence, our scheme includes an initial attempt of learning with less computational and spatial requirement than originally allocated. This corresponds to a single iteration of MOD performed with  $m < k$ . In this way, a feedback can be acquired that reflects the congruence of the model and the data at hand, so that current state can properly be adjusted before the final model update, which consumes the remaining resources.

This approach will be most beneficial for systems that are restricted to

random



Figure 31. A frequency subdomain of learnt dictionaries formed with same processing time. MOD on the left, EcMOD+ on the right.

initializations (as apparent in Figure 1, Figure 2 and Table 1). A random initial model is most likely to be an improper representative of a specific system. Therefore, an update based on this model, no matter how intensive it is, will result in an undesired state. In fact, an optimal update based on this improper model could be more impairing than a subopthnal one in this regard.

As a concluding remark, readers should bear in mind that this work is based on a pragmatic perspective. Although satisfactory improvement has been observed, a more rigorous theoretical approach can lead to certain variations built on top of this framework that will be far more fruitful.

APPENDIX B: K-POLYTOPES ALGORITHM DETAILS

*Algorithms in k-polytopes.* The pseudocode details of the overall algorithm as well as subdivision, pruning and merging methods are illustrated in Alg. 4, Alg. 5, Alg. 6 and Alg. 7, respectively.

Algorithm 5 An algorithm for /c-polytopes

```
1: function fc-POLYTOPES(Y, q, k)
2:   A, S, X, r <- initialize based on Y and q
3:   while not converged do
4:     X project Y onto A restricted to S
5:     A <- YX+ (dictionary update)
6:     A, S <- pruning based on A
7:     A, S <- subdivision depending on r
8:     A, <S <- merging depending on r
9:     C <- get the number of connected components in S
10:    if C ≤ k then
11:      A, T <- update parameters
12:      Go to 3:
13:    return A, X, S
```

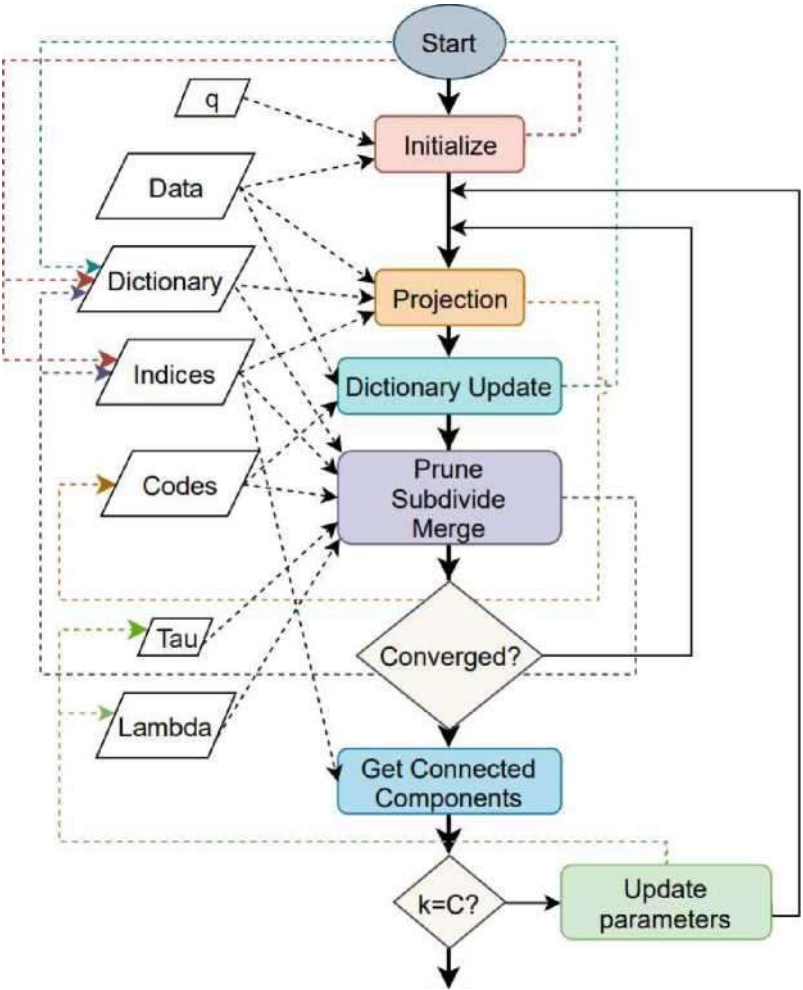
Algorithm 6 An algorithm for the subdivision process

```
1 function subdivide(A, S, t) for all h in <S do
2   K <- designate h = h_K for all pairs of nodes a and b in h_K do if r < ||a - b||_2 then
3     c = a + b
4     A_re = A ∪ c
5     h'_K <- new hyperedge over a
6     i'' <- new hyperedge over b
7     n_K <- (H_K \ K) ∪ {h'_K ∪ K} return A, S
1
```

<b>Algorithm 7</b> An algorithm for the pruning process	
1	<b>function</b> prune( $A, S, \mathbf{X}, A$ )
2	<b>for</b> all $h$ in $S$ <b>do</b>
3	$k$ 4- designate $h = h_K, \mathcal{L}'H_K$
4	$v\%$ 4- $\#$ of data points projected onto $A \setminus$
5	<b>if</b> $< A$ <b>then</b>
6	$T-L_K$ 4- $I-L_K \setminus h_K$
7	$A_k$ 4- remove vertices of $A_K$ solely used by $h_K$ <b>return</b> $A, S$
<hr/>	
<b>Algorithm 8</b> An algorithm for the merging process	
1	<b>function</b> merge( $A, S, t$ )
2	<b>for</b> each pair $h'$ and $h''$ in $S$ <b>do</b>
3	$k'$ 4- designate $h' = h_K > \mathbf{g} H_k'$
4	$k''$ 4- designate $h'' = h_K >> \mathbf{\epsilon} H_k''$
5	$AK', h >$ 4- get simplex designated by $h_K >$
6	$A_K n_h >>$ 4- get simplex designated by $h_K n$
7	<b>if</b> $A^{\wedge} j i'$ and $A_K u_{h,}$ , close enough wrt $r$ <b>then</b>
8	incident 4- “Are and $h_K >>$ incident?”
9	<b>if</b> ( $\downarrow$ incident & $k' == k''$ ) ( $\downarrow \llcorner ! = k''$ ) <b>then</b>
1	make $h_K >$ and $h_K >>$ incident
1	$A_K > . h'$ 4- stitch $A_K > . h >$ and $A_K n_h n$ <b>return</b> $A, S$
1	

*Block-diagram of  $k$ -polytopes.* The block-diagram of the proposed algorithm to solve fc-polytopes problem is illustrated in Fig. 32.

Figure 32. The block-diagram of the proposed algorithm to solve fc-polytopes problem.



## **CURRICULUM VITAE**

**Yigit Oktar was born in İzmir. In 2011, he graduated from University of Chicago with a B.S. degree. In 2013, he completed his M.S.E. from University of Pennsylvania. He worked as a Research Assistant in İzmir University of Economics from year 2017 to 2019. His publications are listed as below:**

- **Oktar, N. and Oktar, Y., 2015. Machine Learning and Neuroimaging. Journal of Neurological Sciences, 32(1).**
- **Oktar, Y. and Türkan, M., 2017, May. Dictionary learning with residual codes. Signal Process, and Comm. Appl. Conf. (SIU) pp. 1-4. IEEE.**
- **Oktar, Y. and Türkan, M., 2018. A review of sparsity-based clustering methods. Signal Processing, 148, pp.20-30.**
- **Oktar, Y. and Türkan, M., 2019. K-polytopes: a superproblem of k-means. Signal, Image and Video Processing, 13(6), pp.1207-1214.**
- **Oktar, Y. and Türkan, M., 2020. Evolutionary simplicial learning as a generative and compact sparse framework for classification. Signal Processing, 174, p. 107634.**
- **Oktar, Y., Ulucan, O., Karakaya, D., Ersoy, E. O., and Türkan, M., 2020. Binocular vision based convolutional networks. Signal Process, and Comm. Appl. Conf. (SIU) IEEE. [Accepted]**
- **Oktar, Y. and Türkan, M., 2020. Classification via simplicial learning. In 2020 IEEE Int. Conf. on image processing (ICIP) [Accepted]**
- **Oktar, Y. and Türkan, M., 2020. On the preservation of spatio-temporal information in machine learning applications. Signal Processing. [Submitted]**