Neural Networks 98 (2018) 154–177

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Towards understanding sparse filtering: A theoretical perspective

Fabio Massimo Zennaro [*], Ke Chen

School of Computer Science, The University of Manchester, Manchester, M13 9PL, UK

ARTICLE INFO

Article history:
Received 2 December 2016
Received in revised form 10 November 2017
Accepted 15 November 2017

Keywords:
Sparse filtering
Feature distribution learning
Soft clustering
Information preservation
Intrinsic structure
Cosine metric

ABSTRACT

In this paper we present a theoretical analysis to understand sparse filtering, a recent and effective algorithm for unsupervised learning. The aim of this research is not to show *whether* or *how well* sparse filtering works, but to understand *why* and *when* sparse filtering does work. We provide a thorough theoretical analysis of sparse filtering and its properties, and further offer an experimental validation of the main outcomes of our theoretical analysis. We show that sparse filtering works by explicitly maximizing the entropy of the learned representations through the maximization of the proxy of sparsity, and by implicitly preserving mutual information between original and learned representations through the constraint of preserving a structure of the data. Specifically, we show that the sparse filtering algorithm implemented using an absolute-value non-linearity determines the preservation of a data structure defined by relations of neighborhoodness under the *cosine distance*. Furthermore, we empirically validate our theoretical results with artificial and real data sets, and we apply our theoretical understanding to explain the success of sparse filtering on real-world problems. Our work provides a strong theoretical basis for understanding sparse filtering: it highlights assumptions and conditions for success behind this feature distribution learning algorithm, and provides insights for developing new feature distribution learning algorithms.

1. Introduction

Unsupervised learning deals with the *problem of modeling data*, stated as the problem of learning a transformation which maps data in a given representation onto a new representation. Contrasted with supervised learning, where we are provided labels and we learn a relationship between the data and the labels, unsupervised learning does not rely on any provided external semantics in the form of labels. In order to learn, unsupervised learning relies on the specification of assumptions and constraints that express our very understanding of the problem of modeling the data; for example, if we judge that a useful representation of the data would be provided by grouping together data instances according to a specific metric, then we may rely on distance-based clustering algorithms to generate one-hot representations of the data.

Often, the tacit aim of unsupervised learning is to generate representations of the data that may simplify the further problem of learning meaningful relationships through supervised learning. Coates, Ng, and Lee (2011) clearly showed that very simple unsupervised learning algorithms (such as k-means clustering), when properly tuned, can generate representations of the data that allow even basic classifiers, such as a linear support vector machine, to achieve state-of-the-art performances.

One common assumption hard-wired in several unsupervised learning algorithms is *sparsity* (for a review on the use of sparsity in representation learning see Bengio, Courville, & Vincent, 2013). Sparse representation learning aims at finding a mapping that produces new representations where few of the components are active while all of the others are reduced to zero. The adoption of sparsity relies both on biological analogies and on theoretical justifications (for discussion on the justification of sparsity see, for instance, Bengio et al., 2013; Földiák & Young, 1995; Ganguli & Sompolinsky, 2012; Olshausen & Field, 1997). Several state-of-the-art algorithms have been developed or have been adapted to learn sparse representations (for a recent survey of these algorithms, see Zhang, Xu, Yang, Li, & Zhang, 2015).

1.1. Sparse filtering and related work

In 2011, Ngiam, Chen, Bhaskar, Koh, and Ng (2011) proposed a novel unsupervised learning framework for generating sparse representations. Most of the successful unsupervised algorithms may be described as *data distribution learning* algorithms that try to learn new representations which better model the underlying probability distribution that generated the data. In contrast, they proposed the possibility of developing *feature distribution learning* algorithms that try to learn new representations having desirable

[*] Corresponding author.
E-mail addresses: zennarof@cs.manchester.ac.uk (F.M. Zennaro), chen@cs.manchester.ac.uk (K. Chen).

https://doi.org/10.1016/j.neunet.2017.11.010

properties, without the need of taking into account the problem of modeling the distribution of the data.

Consistently with the feature distribution learning framework, they defined an algorithm named *sparse filtering*, which ignores the problem of learning the data distribution and instead focuses only on optimizing the sparsity of the learned representations. Sparse filtering proved to be an excellent algorithm for unsupervised learning: it is extremely simple to tune since it has only a single hyper-parameter to select; it scales very well with the dimension of the input; it is easy to implement; and, more importantly, it was shown to achieve state-of-the-art performance on image recognition and phone classification (Goodfellow, Erhan, Carrier, Courville, Mirza, Hamner, Cukierski, Tang, Thaler, Lee, Zhou, Ramaiah, Feng, Li, Wang, Athanasakis, Shawe-Taylor, Milakov, Park, Ionescu, Popescu, Grozea, Bergstra, Xie, Romaszko, Xu, Chuang, & Bengio, 2013; Ngiam et al., 2011; Romaszko, 2013). Thanks to its success and to the simplicity of implementing and integrating the algorithm in already existing machine learning systems, sparse filtering was adopted in many real-world applications (see, for instance, the works of Dong, Pei, He, Liu, Dong, & Jia, 2014; Lei, Jia, Lin, Xing, & Ding, 2015; Raja, Raghavendra, Vemuri, & Busch, 2015; Ryman, Bruce, & Freund, 2016).

Some studies have also provided sparse filtering with some biological support. Bruce, Rahman, and Carrier (2016) analyzed different biologically-grounded principles for representation learning of images, using sparse filtering as a starting point for the definition of new learning algorithms. Interestingly, Kozlov and Gentner (2016) used sparse filtering to model the receptive fields of high-level auditory neurons in the European starling, providing further support to the general hypothesis that sparsity and normalization are general principles of neural computation (Carandini & Heeger, 2012).

1.2. Problem statement

So far, sparse filtering has been successfully applied to many scenarios, and its usefulness repeatedly confirmed (see, for instance, its application in Dong et al., 2014; Han, Lee, Nam, & Lee, 2016; Liu, He, Xie, Gu, Liu, & Pei, 2016; Raja et al., 2015). In general, however, a clear theoretical explanation of the algorithm is still lacking. Ngiam et al. (2011) drew connections between sparse filtering, divisive normalization, independent component analysis, and sparse coding, while Lederer and Guadarrama (2014) provided a deeper analysis of the normalization steps inside the sparse filtering algorithm. However, the reasons why and on what conditions sparse filtering works are left unexplored. In this paper, we aim at understanding from a theoretical perspective *why* and *when* sparse filtering works. It is worth clarifying that our work does not concern itself with showing *whether* or *how well* well sparse filtering works, as there have been abundant evidence in literature on its successes in different real applications.

We begin by arguing that any unsupervised learning algorithm, in order to work properly, has to deal with the problem of preserving information conveyed by the probability distribution of the data. Given that feature distribution learning ignores the problem of learning the data distribution itself, a natural question arises: *how is the information conveyed by the data distribution preserved in feature distribution learning and, specifically, in sparse filtering?*

The actual success of sparse filtering suggests that the algorithm is indeed able to preserve relevant information conveyed in the distribution of the data. However, no explanation for this behavior has been given. We suggest that information may be preserved through the preservation of the structure of the data. To understand how this may be, we study the properties of the transformations within the algorithm and pose the following question: *is there any sort of data structure that is preserved by the processing in sparse filtering?*

Through a theoretical analysis we show that sparse filtering implemented using an absolute-value non-linearity does indeed retain information through the preservation of the data structure defined by the relations of neighborhoodness under the cosine distance. Relying on this, we investigate whether our theoretical results can be used to explain the success or the failure of sparse filtering in real applications. In particular we consider the following questions: *can the success of sparse filtering be explained in terms of the type of structure preserved? Can the failure of alternative forms of sparse filtering using different non-linearities be explained counterfactually on the grounds of information preservation? Is it possible to identify scenarios in which sparse filtering is likely to be helpful and other scenarios in which it is likely not to be useful?*

1.3. Contributions

We summarize the contributions made in this study as follows:

- We provide a theoretical analysis to understand why and when sparse filtering works. We show that the standard sparse filtering algorithm implemented with an absolute-value non-linearity implicitly works under the assumption of an intrinsic radial structure of the data. This assumption naturally makes the algorithm more suitable for certain data sets.
- We empirically validate our main theoretical findings, both on artificial data and real-world data sets.
- We provide useful insights for developing new feature distribution learning algorithms based on our theoretical understanding.

1.4. Organization

The rest of this paper is organized as follows. We first review the concepts and ideas forming the foundations of our work (Section 2). Next, we provide a formal theoretical analysis of the sparse filtering algorithm based on a rigorous conceptualization of feature distribution learning (Section 3). The theoretical results inform the following experimental simulations (Section 4). We then discuss the results we collected, in relation to sparse filtering, in particular, and to feature distribution learning, in general (Section 5). Finally, we draw conclusions by summarizing our contributions and highlighting future developments (Section 6).

To facilitate our presentation, Table 1 summarizes the notation system used in this manuscript.

2. Foundations

In this section we review basic concepts underlying our study. We provide a rigorous description of unsupervised learning, we present its formalization in information-theoretic terms, we formalize the property of sparsity, and, finally, we bring all these concepts together in the definition of the sparse filtering algorithm.

2.1. Unsupervised learning

Let $\mathsf{X} = \{\mathbf{X}^{(i)} \in \mathbb{R}^O\}_{i=1}^N$ be a set of N *samples* or *data points* represented as vectors in an O-dimensional space. We will refer to the given representation of a sample $\mathbf{X}^{(i)}$ in the space $\mathbb{R}^O$ as the *original representation* of the sample $\mathbf{X}^{(i)}$ and to $\mathbb{R}^O$ as the *original space*. From an algebraic point of view, we can formalize the data set as a matrix $\mathbf{X}$ of dimensions $(O \times N)$; from a probabilistic point of view, we can model the data points $\mathbf{X}^{(i)}$ as i.i.d. samples from a multivariate random variable $X = (X_1, X_2, \ldots, X_O)$ with pdf $p(X)$.

Unsupervised learning discovers a transformation $f : \mathbb{R}^O \to \mathbb{R}^L$ mapping the set X from an O-dimensional space to the set

Table 1
Notation.

N	Number of samples.
O	Original dimensionality of the samples.
L	Learned dimensionality of the samples.
$\mathbf{X}$	Matrix of original representations with domain $\mathbb{R}^{O \times N}$.
X	Data set or collection of data.
$\mathbf{X}^{(i)}$	ith sample from $\mathbf{X}$; vector of shape $(O \times 1)$ with domain $\mathbb{R}^{O}$, $1 \leq i \leq N$.
$\mathbf{X}_j$	jth feature from $\mathbf{X}$; vector of shape $(1 \times N)$ with domain $\mathbb{R}^{N}$, $1 \leq j \leq O$.
$\mathbf{X}_j^{(i)}$	jth feature of the ith sample from $\mathbf{X}$; scalar with domain $\mathbb{R}$.
X	Multivariate random variable (random vector) modeling the original data $\mathbf{X}^{(i)}$.
$p(X)$	Probability density function of the original representations.
$p\left(\mathbf{X}^{(i)}\right)$	Probability of the outcome $\mathbf{X}^{(i)}$ when sampling from $p(X)$. $p\left(\mathbf{X}^{(i)}\right)$ is the shorthand for the more rigorous notation $p\left(X = \mathbf{X}^{(i)}\right)$.
$\mathbf{Z}$	Matrix of learned representations with domain $\mathbb{R}^{L \times N}$.
$\mathbf{F}/\tilde{\mathbf{F}}/\hat{\mathbf{F}}$	Matrix of intermediate representations with domain $\mathbb{R}^{L \times N}$.
$\mathbf{Y}$	Vector of labels associated with the data with domain $\mathbb{R}^{1 \times N}$.
$\mathbf{W}$	Matrix of weights with domain $\mathbb{R}^{L \times O}$.

$\mathsf{Z} = \{\mathbf{Z}^{(i)} \in \mathbb{R}^L\}_{i=1}^N$ in an L-dimensional space. We will refer to the transformed representation $\mathbf{Z}^{(i)}$ in the space $\mathbb{R}^L$ as the *learned representation* of the sample $\mathbf{X}^{(i)}$ and to $\mathbb{R}^L$ as the *learned space*. Again, from an algebraic point of view, we can formalize the transformed data set as a matrix $\mathbf{Z}$ of dimensions $(L \times N)$; from a probabilistic point of view, we can model the data points $\mathbf{Z}^{(i)}$ as i.i.d. samples from a multivariate random variable $Z = (Z_1, Z_2, \ldots, Z_L)$ with pdf $p(Z)$.

Unsupervised learning is often used for learning better representations for ensuing supervised tasks. Suppose that we are given a set $\mathsf{Y} = \{\mathbf{Y}^{(i)} \in \mathbb{R}\}_{i=1}^N$ of N labels, such that the ith label in Y is associated to the ith sample in X. From an algebraic point of view, we can formalize the labels as a vector $\mathbf{Y}$ of dimensions $(1 \times N)$; from a probabilistic point of view, we can model the labels $\mathbf{Y}^{(i)}$ as i.i.d. samples from a random variable Y with pdf $p(Y)$. Let us now consider the new data set $(\mathsf{X}, \mathsf{Y}) = \left\{\left(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)}\right) \in \mathbb{R}^O \times \mathbb{R}\right\}_{i=1}^N$. In this scenario, the aim of unsupervised learning is to learn from $\mathbf{X}^{(i)}$ representations $\mathbf{Z}^{(i)}$ such that modeling the relationship $g' : \mathbf{Z}^{(i)} \mapsto \mathbf{Y}^{(i)}$ or the distribution $P(Y|Z)$ is easier than modeling the relationship $g'' : \mathbf{X}^{(i)} \mapsto \mathbf{Y}^{(i)}$ or the distribution $P(Y|X)$.

Clustering. A specific form of unsupervised learning is *clustering*.

Hard clustering discovers a transformation $f : \mathbb{R}^O \to \mathbb{R}^L$ mapping the original samples $\mathbf{X}^{(i)}$ onto one-hot representations $\mathbf{Z}^{(i)}$, where the single non-null component of $\mathbf{Z}^{(i)}$ encodes the assignment of the original sample to a cluster.

Soft clustering discovers a transformation $f : \mathbb{R}^O \to \mathbb{R}^L$ mapping the original samples $\mathbf{X}^{(i)}$ onto representations $\mathbf{Z}^{(i)}$, where the value of each component of $\mathbf{Z}^{(i)}$ encodes the degree of membership of the original sample to each cluster. Soft clustering algorithms may be used for learning representations $\mathbf{Z}^{(i)}$ that simplify the problem of modeling the relationship $g' : \mathbf{Z}^{(i)} \mapsto \mathbf{Y}^{(i)}$; in this case, the soft clustering algorithm is normally grounded in the following assumptions. (i) Samples are taken to be first generated by a stochastic process with pdf $p(X^*)$; the samples are corrupted by various forms of noise; the noisy samples that we receive as original representations $\mathbf{X}^{(i)}$ follow a noisy pdf $p(X)$; the noiseless distribution underlying the data is referred to as *true* pdf $p(X^*)$. (ii) Noiseless samples generated by the true pdf $p(X^*)$ are taken to have a stronger correlation to the labels $\mathbf{Y}^{(i)}$ than the original samples $\mathbf{X}^{(i)}$. (iii) The true pdf $p(X^*)$ may be approximated through a mixture model. (iv) Relationships of neighborhoodness under a chosen metric in the original space $\mathbb{R}^O$ allows us to recover the true pdf $p(X^*)$. Based on these assumptions, soft clustering algorithms instantiate a set of C clusters (each one describing one component of the mixture model) and group into clusters nearby data points. Two data points $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ falling in the same clusters are represented by the same exemplar $\bar{\mathbf{X}}$, assuming that such an exemplar contains all the relevant information carried by $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$, and that the information contained in the difference between $\mathbf{X}^{(1)}$ or $\mathbf{X}^{(2)}$ and the exemplar $\bar{\mathbf{X}}$ amounts to noise. If the assumptions are correct, a soft clustering algorithm will learn new representations $\mathbf{Z}^{(i)}$ whose pdf $p(Z)$ is closer to the true pdf $p(X^*)$ than the original pdf $p(X)$; therefore, it will be easier to learn $g' : \mathbf{Z}^{(i)} \mapsto Y^{(i)}$ or $p(Y|Z)$ than learning $g'' : \mathbf{X}^{(i)} \mapsto \mathbf{Y}^{(i)}$ or $p(Y|X)$.

Distribution Learning. Another form of unsupervised learning is *distribution learning*.

Data distribution learning is a generic term for algorithms that aim at estimating the true pdf $p(X^*)$ from the available data. Examples of data distribution learning algorithms include (Ngiam et al., 2011): denoising auto-encoders (DAE) (Vincent, Larochelle, Bengio, & Manzagol, 2008), restricted Boltzmann machines (RBM) (Hinton, Osindero, & Teh, 2006), and independent component analysis (ICA) (Bell & Sejnowski, 1997). In the context of learning for supervised tasks, if we learn a pdf $p(Z)$ that well approximates the true pdf $p(X^*)$, we can reasonably expect that the ensuing learning of $p(Y|Z)$ will be simplified (Bengio et al., 2013).

Feature distribution learning, in contrast, denotes algorithms aimed at learning a pdf $p(Z)$ which has a set of desirable properties. It overlooks the problem of estimating the true distribution $p(X^*)$ and focuses instead on shaping the learned pdf $p(Z)$ according to chosen criteria. The most representative algorithm of this family is sparse filtering (SF) (Ngiam et al., 2011). In the context of learning for supervised tasks, learning a pdf $p(Z)$ with specific properties is meaningful if we know a priori that certain properties (such as sparsity or smoothness) will be useful for supervised learning.

2.2. *Information-theoretic aspects of unsupervised learning*

Relying on conceptual tools from information theory, Vincent, Larochelle, Lajoie, Bengio, and Manzagol (2010) argued that an unsupervised learning algorithm can generate good representations by satisfying two requirements: (i) retaining information about the input, and (ii) applying constraints that lead to the extraction of useful information from noise.

In more general terms, we may state that a good unsupervised representation may be obtained by satisfying the two following information-theoretic requirements: (i) maximizing the mutual information between input and output (*infomax principle*, Linsker, 1989), and (ii) maximizing a measure of information of the output (*informativeness principle*[1]).

As such, in order to generate good representations, an unsupervised learning algorithm has to somehow negotiate the trade-off between the infomax principle and the informativeness principle:

$$\max_{p(Z)\in\mathcal{P}} \underbrace{D\left[p(X,Z)\parallel p(X)p(Z)\right]}_{\substack{\text{infomax}\\ \text{principle}}} + \underbrace{D\left[p(Z)\parallel q(Z)\right]}_{\substack{\text{informativeness}\\ \text{principle}}} \tag{1}$$

where $D[\cdot]$ is a measure of distance or divergence between pdfs, such as the Kullback–Leibler divergence (MacKay, 2003), q is an entropy-maximizing pdf, and $\mathcal{P}$ is the space of all the pdfs defined on the space of the learned representations $\mathbf{Z}$.

Maximizing the infomax principle may be expressed as the maximization of the mutual information $I[X;Z]$, or, equivalently, as the maximization of the relative entropy between $p(X,Z)$ and $p(X)p(Z)$. Maximizing the informativeness may be expressed as the minimization of the entropy $H[Z]$ or the maximization of the relative entropy between the learned pdf $p(Z)$ and the entropy-maximizing pdf q.

Unfortunately, the objective defined in Eq. (1) is bound to remain mainly theoretical, as information-theoretic quantities are extremely hard to estimate in practice. Therefore we need to rely on approximations or heuristics to make these quantities tractable.

2.3. Sparsity

Given a generic vector $\mathbf{v}$ in an N-dimensional space, $\mathbf{v}$ is sparse if a small number of components of the vector accounts for most of the energy of the vector (Hurley & Rickard, 2009). Practically, the vector $\mathbf{v}$ is *sparse* if $n \ll N$ components of the vector $\mathbf{v}$ are active (that is, have a value different from zero) while the remaining $N-n$ components are inactive (that is, have the value zero). A vector $\mathbf{v}$ is *k-sparse* if exactly k components are active. By analogy, we may define sparsity for matrices (with reference to their components) and for random variables (with reference to their realizations).

Several measures of sparsity have been proposed in the literature (for a review of different measures of sparsity and their properties, see Hurley & Rickard, 2009). A common family of measures of sparsity is the ℓ_p-norm family: $\ell_p(\mathbf{v}) = \sqrt[p]{\sum_{i=1}^{N}|\mathbf{v}_i|^p}$. The most intuitive measure is the ℓ_0-norm which computes the number of non-zero components of a vector; however, this measure is practically inadequate, as in concrete implementations the components of a vector are rarely reduced perfectly to zero. The simplest relaxation of the ℓ_0-norm is the ℓ_1-norm, which is often referred to as *activation* in the sparse filtering literature. The negative form of the ℓ_1-norm works as an efficient proxy for measuring the ℓ_0-norm (Elad, 2010). Given a representation $\mathbf{Z}^{(i)}$, $\ell_1\left(\mathbf{Z}^{(i)}\right)$ or *activation* $\left(\mathbf{Z}^{(i)}\right)$ quantifies the sparsity of $\mathbf{Z}^{(i)}$. Minimizing the activation of the learned representation $\mathbf{Z}^{(i)}$ will maximize the ℓ_0-norm and the sparsity of $\mathbf{Z}^{(i)}$.

2.4. Sparse filtering

Sparse filtering is the most representative example of feature distribution learning algorithms (Ngiam et al., 2011). Its aim is learning a pdf $p(Z)$ which maximizes the sparsity of the learned representations $\mathbf{Z}^{(i)}$.

Enforcement of sparsity in sparse filtering. Sparse learned representations $\mathbf{Z}^{(i)}$ are achieved by enforcing three constraints on the matrix of learned representations $\mathbf{Z}$:

- *Population sparsity*: each sample $\mathbf{Z}^{(i)}$, is required to be sparse, that is, described only by a few features. The sparsity of a sample $\mathbf{Z}^{(i)}$ is computed as its activation: $\ell_1\left(\mathbf{Z}^{(i)}\right) = \sum_{j=1}^{L}\left|\mathbf{Z}_j^{(i)}\right|$.
- *Lifetime sparsity*: each feature $\mathbf{Z}_j$, is required to be sparse, that is, to describe only a few samples. Lifetime sparsity is often referred to as *selectivity* (Goh, Thome, Cord, & Lim, 2012). The sparsity of a feature $\mathbf{Z}_j$ is computed as its activation: $\ell_1\left(\mathbf{Z}_j\right) = \sum_{i=1}^{N}\left|\mathbf{Z}_j^{(i)}\right|$.
- *High dispersal*: all the features are required to have approximately the same activation. The dispersal of the features is computed as the variance of the activation across all the features: $Var\left[activation\left(\mathbf{Z}_j\right)\right] = E\left[\ell_1\left(\mathbf{Z}_j\right)^2\right] - E\left[\ell_1\left(\mathbf{Z}_j\right)\right]^2$. Lower variance corresponds to higher dispersal.

The enforcement of these three properties translates into learning non-degenerate sparse representation.

Sparse filtering algorithm. Sparse filtering is implemented as a simple algorithm in six steps (refer to Fig. 1 for an illustration of the transformations on a two-dimensional data set):

A0. Initialization of the weights: the weight matrix $\mathbf{W} \in \mathbb{R}^{L\times O}$ is initialized sampling each component from a normal distribution $\mathcal{N}(0,1)$.

A1. Linear projection of the original data: $f_{A1}(\mathbf{X}) = \mathbf{WX}$. The weight matrix $\mathbf{W}$ can be interpreted as a *dictionary* (Denil & de Freitas, 2012) or as a *filter bank* (Dong, Wu, Pei, & Jia, 2015), where each row is a codeword or a filter applied to each sample. Refer to Fig. 1(a) and 1(b) for an illustration of this transformation.

A2. Non-linear transformation: $\mathbf{F} = f_{A2}(\mathbf{WX})$, where $f_{A2}(\cdot): \mathbb{R} \rightarrow \mathbb{R}$ is an element-wise non-linear function. Although this non-linear function can, in principle, be arbitrarily chosen, all the implementations known to the authors used an element-wise absolute-value function $f(x) = |x|$. For practical reasons, this non-linearity is implemented as a soft absolute-value $f(x) = \sqrt{x^2+\epsilon}$, where ϵ is a small negligible value (for instance, $\epsilon = 10^{-8}$). Refer to Fig. 1(b) and 1(c) for an illustration of this transformation.

A3. ℓ_2-normalization along the features (or along the rows): $\tilde{\mathbf{F}} = f_{A3}(\mathbf{F}) = \left[\frac{\mathbf{F}_j^{(i)}}{\sqrt{\sum_{i=1}^{N}\left(\mathbf{F}_j^{(i)}\right)^2}}\right]$. In this step, each feature is normalized so that its squared activation is one, that is, $\sum_{i=1}^{N}\left(\tilde{\mathbf{F}}_j^{(i)}\right)^2 = 1$. Refer to Fig. 1(c) and 1(d) for an illustration of this transformation.

A4. ℓ_2-normalization along the samples (or along the columns): $\mathbf{Z} = \hat{\mathbf{F}} = f_{A4}\left(\tilde{\mathbf{F}}\right) = \left[\frac{\tilde{\mathbf{F}}_j^{(i)}}{\sqrt{\sum_{j=1}^{L}\left(\tilde{\mathbf{F}}_j^{(i)}\right)^2}}\right]$. In this step, each sample is normalized so that its squared activation is one, that is, $\sum_{j=1}^{L}\left(\hat{\mathbf{F}}_j^{(i)}\right)^2 = 1$. Refer to Fig. 1(d) and 1(e) for an illustration of this transformation.

A5. ℓ_1-minimization: $\min_{\hat{\mathbf{F}}\in\mathbb{R}^{L\times N}} \sum_{ij}\hat{\mathbf{F}}_j^{(i)}$. This minimization is the objective of sparse filtering; by minimizing the overall activation of the matrix $\hat{\mathbf{F}}$, we maximize the sparsity of the learned representations. Notice that, when learning, this objective is optimized by back-propagation with respect to the only learnable parameter $\mathbf{W}$, so that the actual objective function of sparse filtering can be defined as $\mathrm{argmin}_{\mathbf{W}\in\mathbb{R}^{L\times O}} \sum_{ij}\hat{\mathbf{F}}_j^{(i)}$.

After learning, new data $\mathbf{X}'$ is processed through step A1 to A4, such that $\mathbf{Z}' = f_{A1:A4}\left(\mathbf{X}'\right) = f_{A4}\left(f_{A3}\left(f_{A2}\left(\mathbf{WX}'\right)\right)\right)$.

[1] We named this principle *informativeness principle* for lack of a better term.

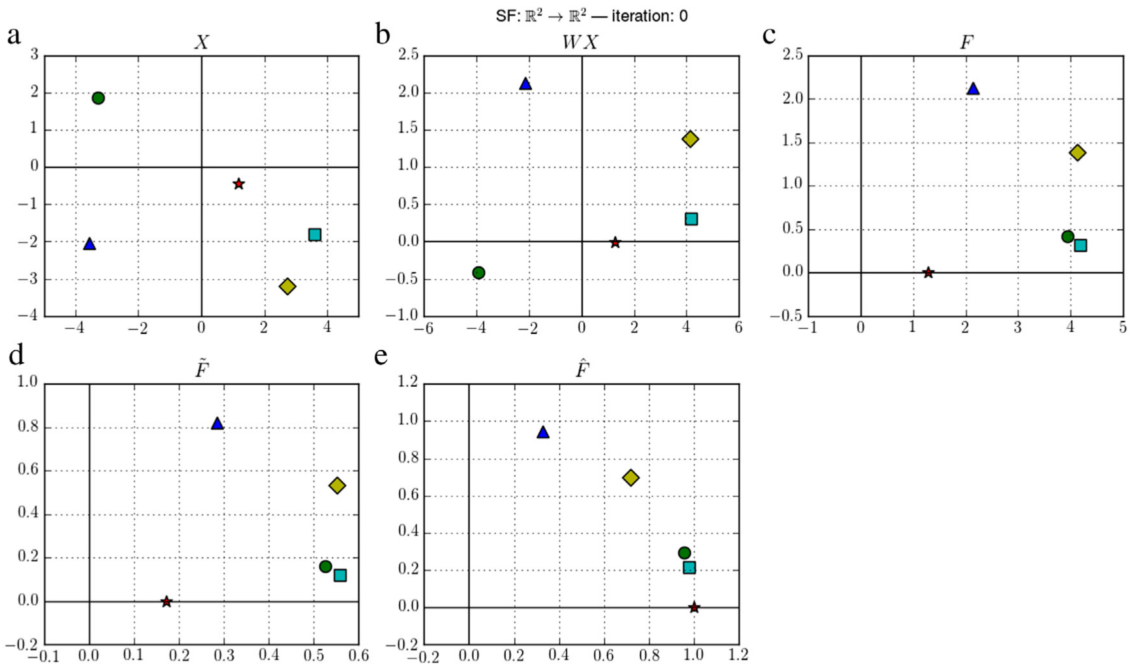

Fig. 1. Illustration of sparse filtering. Sparse filtering is applied to a random set of data X constituted by the matrix **X** containing five samples ($N = 5$) in two dimensions ($O = 2$). Each point is generated by sampling its coordinates from a uniform distribution *Unif* $(-5, 5)$. Sparse filtering is used to learn a new representation of the data in two dimensions ($L = 2$). This figure shows the transformations determined by the sparse filtering algorithm at iteration 0, after the weight matrix **W** has been randomly initialized and before any training. (a) Original representation of the data **X** in $\mathbb{R}^2$. (b) Linear projection of the data onto the intermediate representation **WX**. (c) Non-linear projection of **WX** using an absolute-value function onto the intermediate representation **F**. (d) ℓ_2-normalization of the data **F** along the features, yielding the intermediate representation $\tilde{\mathbf{F}}$. (e) ℓ_2-normalization of the data $\tilde{\mathbf{F}}$ along the samples, yielding the final learned representation $\hat{\mathbf{F}} = \mathbf{Z}$. Notice that the colors and the markers of the data points $\mathbf{X}^{(i)}$ do not have any meanings. A random color and marker have been assigned to each point in order to allow the tracking of the location of the points through the different transformations applied by sparse filtering. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

As explained by Ngiam et al. (2011), the combination of the ℓ_1-minimization with the two ℓ_2-normalizations guarantees the learning of a representation with the properties of population sparsity, lifetime sparsity, and high dispersal.

3. Theoretical analysis of sparse filtering

In order to better understand sparse filtering we first reformulate and explain this algorithm in terms of information-theoretic concepts. Relying on this improved understanding, we will then move on to a formal analysis of the sparse filtering algorithm.

In particular, Section 3.1 presents our thesis stating that sparse filtering must satisfy the informativeness and the infomax principle. Section 3.2 shows how sparse filtering satisfies the informativeness principle, while Section 3.3 introduces the hypothesis that sparse filtering satisfies the infomax principle through the preservation of structure of the data. The hypothesis on the preservation of structure is then analyzed in details in the following sections: Section 3.4 rules out the simplest hypothesis that sparse filtering preserves a structure explained by the Euclidean metric; Section 3.5 proves that sparse filtering preserve collinearity; Section 3.6 proves that collinear points are mapped onto identical representation; similarly, Section 3.7 proves that points having the same moduli are mapped onto identical representations; and, finally, Section 3.8 puts together these results to conclude that sparse filtering preserves relations of cosine neighborhoodness. Sections 3.9 and 3.10 delve deeper in the dynamics of sparse filtering providing a geometric interpretation of the algorithm in terms of basis of the learned space and filters in the original space. Sections 3.11 and 3.12 investigate the limits of the sparse filtering algorithm, by evaluating more closely the role of the absolute-value non-linearity in the preservation of structure and by deriving a probabilistic bound on the preservation of Euclidean structure. Finally, Section 3.13 draws together all the results by discussing the use of sparse filtering as a representation learning algorithm.

3.1. Information-theoretic aspects of sparse filtering

With reference to the information-theoretic description of unsupervised learning presented in Section 2.2, the aim of sparse filtering seems to be a pure optimization of the informativeness principle. Indeed, sparse filtering algorithm explicitly maximizes a property of the learned distribution (related to the informativeness principle), but it makes no reference to the problem of preserving information in the original distribution (related to the infomax principle); its loss function seems to be concerned only with the second term in Eq. (1) and to disregard the first term.

However, based on our information-theoretic understanding of unsupervised learning we argue that, actually, sparse filtering must, in some way, take into account the infomax principle. In the following, we demonstrate the following thesis:

Sparse filtering does satisfy the informativeness principle through the maximization of the proxy of sparsity and it satisfies the infomax principle through the constraint of preservation of the structure of cosine neighborhoodness of the data.

3.2. Informativeness principle in sparse filtering

Showing that sparse filtering satisfies the informativeness principle is straightforward. Since the explicit the minimization of the entropy $H[Z]$ is computationally hard, the sparse filtering algorithm adopts the standard proxy of sparsity. Increasing the sparsity of the representations $\mathbf{Z}^{(i)}$ concentrates the mass of the pdf $p(Z)$

around zero; as the pdf $p(Z)$ gets closer to a Dirac delta function, its entropy is $H[Z]$ is minimized (Hurley & Rickard, 2009; Principe, 2010). Using the formalism of Pastor, Mora-Jiménez, Jäntti, and Caamaño (2015):

$$-\ell_1(\mathbf{Z}) \uparrow \equiv H[Z] \downarrow,$$

that is, as the sparsity, measured by the negative ℓ_1-norm of the learned representations $\mathbf{Z}^{(i)}$ increases, so the entropy of the pdf $p(Z)$ decreases.

3.3. Infomax principle in sparse filtering

Showing that sparse filtering satisfies the informativeness principle is more challenging. By definition, as a feature distribution learning algorithm, sparse filtering does not address the problem of modeling the data distribution. However, by virtue of the fact that sparse filtering works and its learned representations $\mathbf{Z}^{(i)}$ allow the achievement of state-of-the-art performance when learning $p(Y|Z)$, it *must* be that the algorithm preserves information contained in the original representations $\mathbf{X}^{(i)}$. *If it were not so*, sparse filtering could simply solve its optimization problem by mapping the original data matrix $\mathbf{X}$ onto a pre-computed sparse representation matrix $\bar{\mathbf{Z}}$, containing a constant 1-sparse learned representation $\bar{\mathbf{Z}}^{(1)}$, with a minimal computational complexity of $O(1)$. The matrix $\bar{\mathbf{Z}}$ would have maximal sparsity, and the associated pdf $p(Z)$ would be a Dirac delta function centered on $\bar{\mathbf{Z}}^{(1)}$ with minimal entropy. However, if we were to use $\bar{\mathbf{Z}}$ to perform further supervised learning with respect to a vector of label $\mathbf{Y}$, the pre-computed learned representations $\mathbf{Z}^{(i)} = \bar{\mathbf{Z}}^{(1)}$ would be useless as they would provide no information about the labels because of the independence between the pre-computed representations and the given labels: $p(Y|Z) = p(Y)$.

Since sparse filtering does not try to explicitly model the distribution of the original data we hypothesize that it must implicitly preserve information about the pdf $p(X)$ through the proxy of the preservation of data structure. The geometric structure of the data in the original space $\mathbb{R}^O$ constitutes a set of realizations of the random variable X through which we can estimate the pdf $p(X)$. Preserving relationships of neighborhoodness (under a given metric) allows us to preserve information conveyed by the pdf $p(X)$: regions of high density and low density in the domain of $p(X)$ can be maintained by preserving relationships of neighborhoodness in the domain of $p(Z)$. Thus, preservation of the geometric structure under a chosen metric may act as a proxy for the maximization of mutual information $I[X; Z]$.

3.4. Non-preservation of Euclidean distance

The preservation of absolute or relative distances under the Euclidean metric is the most common way to preserve the structure of the data. However, it can be easily ruled out that sparse filtering preserves this type of structure.

Proposition 1. *Let* $\left\{\mathbf{X}^{(i)} \in \mathbb{R}^O\right\}_{i=1}^N$ *be a set of points in the original space* $\mathbb{R}^O$. *Then, the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric.*

Proof Sketch. This proposition is proved by counterexample showing that there is at least a case for which the transformations from A1 to A4 do not preserve the Euclidean distance. The full proof is available in Appendix A.1. ■

3.5. Preservation of collinearity

Having ascertained that sparse filtering cannot preserve the data structure defined by the Euclidean metric, we investigate other properties of the algorithm that may lead us to discover the preservation of alternative data structures. A first relevant observation is that sparse filtering preserves collinearity of point lying on the same line passing through the origin of the space $\mathbb{R}^O$.

Theorem 1. *Let* $\mathbf{X}^{(1)}, \mathbf{X}^{(2)} \in \mathbb{R}^O$ *be collinear points in the original space* $\mathbb{R}^O$. *Then, the outputs of transformations from A1 to A4, that is* $f_{A1:A4}\left(\mathbf{X}^{(1)}\right), f_{A1:A4}\left(\mathbf{X}^{(2)}\right) \in \mathbb{R}^L$, *are collinear.*

Before proving this theorem, we present a set of auxiliary lemmas. The proofs of these lemmas are elementary and they can be found in Appendices A.2–A.5.

Lemma 1. *Let us consider* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^O$, *two generic collinear vectors, and let* $f : \mathbb{R}^O \to \mathbb{R}^L$ *be a linear transformation defined as* $f(\mathbf{u}) = \mathbf{W}\mathbf{u}$, *where* $\mathbf{W}$ *is the matrix associated with the linear transformation. Then,* $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ *are also collinear.*

Lemma 2. *Let us consider* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, *two generic collinear vectors, and let* $f : \mathbb{R}^L \to \mathbb{R}^L$ *be the element-wise absolute-value function* $f(\mathbf{u}) = |\mathbf{u}| = \left[\left|u_j\right|\right]$. *Then* $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ *are also collinear.*

Lemma 3. *Let us consider* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, *two collinear vectors whose components are all strictly positive,*[2] *and let* $f : \mathbb{R}^L \to \mathbb{R}^L$ *be the* ℓ_2*-normalization along the features. Then* $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ *are also collinear.*

Lemma 4. *Let us consider* $\mathbf{u} \in \mathbb{R}^L$, *a vector whose components are all strictly positive,*[3] *and let* $f : \mathbb{R}^L \to \mathbb{R}^L$ *be the* ℓ_2*-normalization along the samples. Then* $f(\mathbf{u}) \in \mathbb{R}^L$ *have the same angular coordinates as* $\mathbf{u}$.

Using these lemmas, we can prove Theorem 1.

Proof of Theorem 1. To prove that the transformations from A1 to A4 preserve collinearity it is necessary to prove that all transformations preserve collinearity.

Concerning transformation A1, by Lemma 1, linear transformations preserve collinearity. Concerning transformation A2, by Lemma 2, absolute-value function preserves collinearity; indeed, it rigidly folds all the orthants on the first one. Concerning transformation A3, by Lemma 3, normalization along the features preserves collinearity; indeed, it acts simply as a rescaling of the axes. Concerning transformation A4, by Lemma 4, normalization along the samples preserves angular distances in general, and, therefore, collinearity.

Since all the transformations from A1 to A4 preserve collinearity, the overall transformation $f_{A1:A4}(\cdot)$ preserves collinearity. ■

3.6. Homo-representation of collinear points

An immediate consequence of the previous result is the following theorem which states that all the collinear points in the original representation space are mapped to an identical representation. This result is significant as it gives us a first understanding of the principle and the type of metric that sparse filtering uses to map original samples $\mathbf{X}^{(i)}$ onto their representations $\mathbf{Z}^{(i)}$.

Theorem 2. *Let* $\mathbf{X}^{(1)} \in \mathbb{R}^O$ *be a point in the original space* $\mathbb{R}^O$. *Then there is a set of infinite points* $\mathbf{X}^{(i)} \in \mathbb{R}^O$ *such that* $f_{A1:A4}\left(\mathbf{X}^{(1)}\right) = f_{A1:A4}\left(\mathbf{X}^{(i)}\right)$. *The set of the points collinear with* $\mathbf{X}^{(1)}$ *is included in this set.*

[2] Notice that we can safely make the assumption of strict positivity in sparse filtering since $\mathbf{u}$ and $\mathbf{v}$ are the output of an absolute-value function implemented as $f(x) = \sqrt{x^2 + \epsilon}$.

[3] Notice that we can safely make the assumption of strict positivity in sparse filtering since $\mathbf{u}$ is the output of the normalization along the feature which preserves the positivity.

$$\begin{array}{ll}
f_{A1}\left(\mathbf{X}^{(1)}\right)=\mathbf{W}\mathbf{X}^{(1)} & f_{A1}\left(\mathbf{X}^{(2)}\right)=k\mathbf{W}\mathbf{X}^{(1)}\\
f_{A2}\left(\mathbf{W}\mathbf{X}^{(1)}\right)=\left|\mathbf{W}\mathbf{X}^{(1)}\right| & f_{A2}\left(k\mathbf{W}\mathbf{X}^{(1)}\right)=k\left|\mathbf{W}\mathbf{X}^{(1)}\right|\\
f_{A3}\left(\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)=\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right| & f_{A3}\left(k\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)=k\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)\\
f_{A4}\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)=\dfrac{\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|}{\ell_2\left(\mathbf{X}^{(1)}\right)} & f_{A4}\left(k\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)\right)=\dfrac{k\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)}{\ell_2\left(\mathbf{X}^{(2)}\right)}
\end{array}$$

Box I.

Proof. Let us consider a point $\mathbf{X}^{(1)}$ and a generic collinear point $\mathbf{X}^{(2)}=k\mathbf{X}^{(1)}$, $k\neq 0$. Let us apply the transformation $f_{A1:A4}$ to the points $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Then we get the equations given in Box I, where $\mathbf{c}$ is a vector of normalizing constants and $\circ$ is the element-wise product (as in Lemma 3). Now, since $\ell_2\left(\mathbf{X}^{(2)}\right)=k\ell_2\left(\mathbf{X}^{(1)}\right)$, it follows:

$$\frac{k\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)}{\ell_2\left(\mathbf{X}^{(2)}\right)}=\frac{k\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right)}{k\ell_2\left(\mathbf{X}^{(1)}\right)}=f_{A4}\left(\mathbf{c}\circ\left|\mathbf{W}\mathbf{X}^{(1)}\right|\right).$$

Thus, $\mathbf{X}^{(1)}$ and any collinear point $\mathbf{X}^{(2)}$ are mapped onto the same representation $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$. ■

3.7. Homo-representation of points with same moduli

A further analysis of sparse filtering reveals that not only collinear points are mapped to the same representation, but also points in the learned representation space having the same moduli (that is, the same absolute value for their components) are mapped to identical representations. Again, this result is relevant since it sheds light on the type of structure preserved by sparse filtering.

Theorem 3. *Let $\mathbf{F}^{(1)}\in\mathbb{R}^L$ be a point in the codomain of the linear map defined by the matrix $\mathbf{W}$. It holds that for $\mathbf{F}^{(1)}$ strictly in the first orthant, there are at least 2^L points $\mathbf{F}^{(i)}\in\mathbb{R}^L$ such that $f_{A2:A4}\left(\mathbf{F}^{(1)}\right)=f_{A2:A4}\left(\mathbf{F}^{(i)}\right)$.*

Proof. By definition, $\mathbf{F}_j^{(1)}>0$, $\forall j$, $1\leq j\leq L$. It follows that $f_{A2}(\mathbf{F}^{(1)})=\mathbf{F}^{(1)}$, as the application of the absolute-value maps $\mathbf{F}^{(1)}$ to itself.

However, all the vectors $\mathbf{F}^{(i)}$ such that $\mathbf{F}_j^{(i)}=\pm\mathbf{F}_j^{(1)}$ are mapped to $\mathbf{F}^{(1)}$ by the absolute-value $f_{A2}(\cdot)$. By combinatorial analysis, there are 2^L possible ways of picking the values of $\mathbf{F}^{(i)}$, thus defining 2^L points in $\mathbb{R}^L$ that are mapped to the same value $\mathbf{F}^{(1)}$. Since all the points $\mathbf{F}^{(i)}$ are mapped to the same point $\mathbf{F}^{(1)}$ at the end of step A2, the application of the remaining deterministic functions will map them to the same representation, $f_{A2:A4}\left(\mathbf{F}^{(1)}\right)=f_{A2:A4}\left(\mathbf{F}^{(i)}\right)$. ■

3.8. Preservation of cosine neighborhoodness

In Theorem 2 we have shown that sparse filtering maps points having the same angles to the same representation. However, this property is not sufficient to preserve any complex structure. Here we further prove that sparse filtering maps points having a small cosine distance in the original space onto point having small Euclidean distance in the representation space.

Theorem 4. *Let $\mathbf{X}^{(1)},\mathbf{X}^{(2)}\in\mathbb{R}^O$ be two original data samples and let $\mathbf{Z}^{(1)},\mathbf{Z}^{(2)}\in\mathbb{R}^L$ be their representations computed by sparse filtering. If the cosine distance between the original samples is arbitrarily small $D_C\left[\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right]<\delta$, for $\delta>0$, then the Euclidean distance between the computed representations is arbitrarily small $D_E\left[\mathbf{Z}^{(1)},\mathbf{Z}^{(2)}\right]<\epsilon$, for $\epsilon>0$, and $\epsilon=L\cdot\left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}-\frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right)$, where k is a constant accounting for partial collinearity and $\ell_2\left(\tilde{\mathbf{F}}^{(i)}\right)$ is the ℓ_2-norm of the representations computed by sparse filtering after step A3. In the limit, it holds that $\lim_{\delta\to 0}\epsilon=0$.*

Proof Sketch. We provide here a synthetic sketch of the proof; the full proof is available in Appendix A.6.

We prove this theorem with the following approach: at each step of sparse filtering, (i) we consider the displacement between the representation of the two points $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$; (ii) we upper bound the displacement.

Before sparse filtering the displacement vector $\bar{\mathbf{X}}$ between $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$\bar{\mathbf{X}}=(k-1)\mathbf{X}^{(1)}+\mathbf{B} \tag{2}$$

where $k\in\mathbb{R}, k\neq 0$, is a constant accounting for partial collinearity and $\mathbf{B}$ is a bias vector. Knowing that the cosine distance between the samples is bounded by $D_C\left[\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right]<\delta$, the displacement can be upper bounded component-wise as:

$$\bar{\mathbf{X}}_j\leq\mathbf{X}_j^{(1)}\left(k-1+\sqrt{2\delta-\delta^2}\right). \tag{3}$$

After steps A1 and A2, the new displacement is:

$$\bar{\mathbf{F}}_l=(k-1)\,\mathbf{F}_l^{(1)}\pm|\mathbf{W}\mathbf{B}|_l, \tag{4}$$

whose upper bound is:

$$\bar{\mathbf{F}}_l\leq\left(k-1+\left|\sqrt{2\delta-\delta^2}\right|\right)\left|\sum_{j=1}^{O}\mathbf{W}^{(l)}\mathbf{X}_j^{(1)}\right|. \tag{5}$$

After the normalization along the rows in step A3, the displacement is reduced to:

$$\bar{\tilde{\mathbf{F}}}_l=(k-1)\,\tilde{\mathbf{F}}_l^{(1)}+\frac{|\mathbf{W}\mathbf{B}|_l}{c_l}, \tag{6}$$

where $\{c_l\}_{l=1}^L, c\in\mathbb{R}$ are constant accounting for feature-dependent sums across the N samples. Consequently the new upper bound is simply:

$$\bar{\tilde{\mathbf{F}}}_l\leq\frac{1}{c_l}\bar{\mathbf{F}}_l. \tag{7}$$

Finally, after step A4, the new displacement is:

$$\bar{\mathbf{Z}}_l=\left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}-1\right)\mathbf{Z}_l^{(1)}+\frac{|\mathbf{W}\mathbf{B}|_l}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}, \tag{8}$$

which can be upper bounded as:

$$\bar{\mathbf{Z}}_l\leq\frac{\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}\left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}-\frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right). \tag{9}$$

Therefore, the overall Euclidean distance between the representations $\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$ can be bounded by:

$$D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] \leq L \cdot \left(\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} \right). \tag{10}$$

Thus $\epsilon = L \cdot \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} \right)$. ■

Sparse filtering can then preserve cosine neighborhoodness by mapping points that have similar angular coordinates to representations that are close to each other under an Euclidean metric. However points that have large cosine distance in the original space will not necessarily be far in the representation space; this is a consequence of the fact that transformation in sparse filtering preserve collinearity and cosine neighborhoodness, but not cosine metric in general.

3.9. Basis and basis pursuit

Let us now consider the space of the learned representations $\mathbb{R}^L$. This space is spanned by the canonical set of orthonormal basis vectors $\{\mathbf{e}_i\}_{i=1}^L$, where $\mathbf{e}_1 = \begin{bmatrix}1 & 0 & \dots & 0\end{bmatrix}$, $\mathbf{e}_2 = \begin{bmatrix}0 & 1 & \dots & 0\end{bmatrix}$, ..., $\mathbf{e}_L = \begin{bmatrix}0 & 0 & \dots & 1\end{bmatrix}$.

Let Z be the set of vectors $\left\{\mathbf{Z}^{(i)}\right\}_{i=1}^N$ produced by the sparse filtering algorithm through the steps A1 to A4. If we now consider the optimization in step A5, it is easy to prove that the optimal set Z that minimizes the ℓ_1-norm is given by a multi-set[4] of the orthonormal basis vectors of $\mathbb{R}^L$.

Proposition 2. *Let $\left\{\mathbf{Z}^{(i)}\right\}_{i=1}^N$ be a set of normal vectors in $\mathbb{R}^L$. Then $\left\{\mathbf{Z}^{(i)}\right\}_{i=1}^N$ is an optimal set of vectors with respect to the ℓ_1-norm optimization problem $\min_{\mathsf{Z}\in\mathbb{R}^{L\times N}} \sum_{i=1}^N \sum_{j=1}^L \mathbf{Z}_j^{(i)}$ if and only if this set $\left\{\mathbf{Z}^{(i)}\right\}_{i=1}^N$ is a multi-set of the orthonormal basis vectors of $\mathbb{R}^L$.*

Proof Sketch. This proposition is proved geometrically, following the proof given by Bishop (2007) to show that the solutions to regularized least-squares optimization problems are sparse. The full proof is available in Appendix A.7. ■

Thus, the optimal solution for the sparse filtering algorithm is to map a set of original representations $\mathbf{X}^{(i)} \in \mathbb{R}^O$ onto the orthonormal basis vectors of $\mathbb{R}^L$, as the basis vectors $\{\mathbf{e}_i\}_{i=1}^L$ have a minimal ℓ_1-norm in $\mathbb{R}^L$ under the constraint of sparse filtering.

Ideally, through gradient descent, sparse filtering progressively pushes all the learned representations $\mathbf{Z}^{(i)} \in \mathbb{R}^L$ towards the orthonormal basis vectors of $\mathbb{R}^L$. However, in general, notice that sparse filtering is not guaranteed to find an optimal solution and it may settle into a local minimum, where the original representations $\mathbf{X}^{(i)}$ are mapped onto k-sparse ($k > 1$) representations in $\mathbb{R}^L$. The quality of the solution depends on the original data set X, on the dimensionality of the learned space L, and on the random initialization of the weight matrix $\mathbf{W}$.

Understanding the dynamics of sparse filtering in terms of basis and pursuit of basis naturally prompts a comparison with other sparse learning techniques used in signal processing and machine learning. *Basis pursuit* (Chen, Donoho, & Saunders, 2001) defines a similar ℓ_1-minimization problem, but it considers only a constraint given by a linear transformation, while sparse filtering transforms the data through non-linear transformations. *Matching pursuit* (Mallat & Zhang, 1993) aims at learning a sparse representation; differently from sparse filtering which operates on all the data in parallel, matching pursuit works by finding a linear decomposition in an iterative way by selecting one basis vector at each iteration. Dictionary learning algorithms, such as the *method of optimal directions* (Engan, Aase, & Husoy, 1999) or *k-Singular Value Decomposition* (Aharon, Elad, & Bruckstein, 2006), try to learn a dictionary and a sparse representation at the same time; however, they typically alternate between updating the dictionary and the sparse representation, while sparse filtering explicitly optimizes only the sparsity of the learned representation.

3.10. Representation filters

The idea of orthonormal basis and pursuit of this basis allows us to introduce a last conceptual tool that gives us a better insight into the dynamics of sparse filtering.

From Theorem 2 we learned that sparse filtering identifies sets of collinear points in the original space to be mapped onto basis vectors; from Theorem 3 we can deduce that there must a symmetric structure around lines of collinear points; from Theorem 4 we learned that cosine neighborhoodness is translated into Euclidean neighborhoodness. Putting together these results, we can infer that sparse filtering defines precise maps in the original representation space $\mathbb{R}^O$ in the form of *representation filters*:

Definition (*Representation Filter*)**.** A *representation filter* $R^{\mathbf{e}_j}$ is a function $R^{\mathbf{e}_j} : \mathbb{R}^O \to \mathbb{R}_{\geq 0}$ such that, for any point $\mathbf{X}^{(i)} \in \mathbb{R}^O$, $R^{\mathbf{e}_j}\left(\mathbf{X}^{(i)}\right) = D_E\left(\mathbf{e}_j, \mathbf{Z}^{(i)}\right)$, where $\mathbf{Z}^{(i)} \in \mathbb{R}^L$ is the learned representation of $\mathbf{X}^{(i)}$ and $D_E(\cdot, \cdot)$ is the Euclidean distance.

A representation filters $R^{\mathbf{e}_j}$ maps points in the original representation space $\mathbb{R}^O$ to their Euclidean distance from the basis vector $\mathbf{e}_j$. Plotting a representation filter $R^{\mathbf{e}_j}$ in $\mathbb{R}^O$ defines a region of space having a hyper-conical shape, such that all the points on the line of its height are mapped onto the basis vector $\mathbf{e}_j$, and all the points in the neighborhood defined by its volume are mapped into the neighborhood of the basis vector $\mathbf{e}_j$. Moreover, given a point $\mathbf{X}^{(i)} \in \mathbb{R}^O$, we say that the representation filter $R^{\mathbf{e}_j}_{\mathbf{X}^{(i)}}$ is centered on $\mathbf{X}^{(i)}$ if $R^{\mathbf{e}_j}_{\mathbf{X}^{(i)}}\left(\mathbf{X}^{(i)}\right) = 0$, that is, the point $\mathbf{X}^{(i)}$ lies on the line of the height of the representation cone defined by $R^{\mathbf{e}_j}_{\mathbf{X}^{(i)}}$.

The optimization process of sparse filtering can be interpreted as the *search for an optimal location of the representation filters*: hyper-conical representation filter are rotated in a continuous way in the original representation space during learning, until their placement provides an optimal solution in terms of sparsity of the learned representations. Inspecting the representation filters can provide a way to assess the progress of learning in sparse filtering.

3.11. Non-preservation of cosine neighborhoodness in alternative implementations of sparse filtering

The choice of the absolute-value non-linearity in step A2 of sparse filtering is crucial for guaranteeing the preservation of cosine neighborhoodness. Ngiam et al. (2011) suggest that this non-linearity may be substituted by other non-linear functions; for instance, standard non-linear functions from the neural networks literature, such as the sigmoid non-linearity or the rectified linear unit (ReLU), may be used. Despite this possibility, all the implementations of sparse filtering so far have relied on the absolute-value non-linearity. An unpublished technical report by Thaler[5] states that sparse filtering with alternative non-linearities (ReLU and quadratic non-linearity) does not perform as well as the absolute-value non-linearity, but does not clarify the reasons of this failure. For plain empirical reasons, the absolute-value has

[4] We now explicitly refer to Z as a multi-set because the optimal Z may contain repeated orthonormal basis vectors of $\mathbb{R}^L$ in case $N > L$.

[5] https://www.kaggle.com/c/challenges-in-representation-learning-the-black-box-learning-challenge/forums/t/4717/1st-place-entry.

always been recommended as the best non-linearity for sparse filtering.

One theoretical reason for the limited success of alternative implementations of sparse filtering is the fact that other non-linearities cannot provide strong guarantees on preservation of relevant data structures. If we replace the absolute-value non-linearity with another non-linearity, such as sigmoid or ReLU function, we lose the property of preservation of structure guaranteed by standard sparse filtering, as it is proved by the following two propositions.

Proposition 3. *Let us consider the sparse filtering algorithm implemented using a sigmoid non-linearity* $\sigma(x) = \frac{1}{1+e^{-x}}$. *Let* $\left\{\mathbf{X}^{(i)} \in \mathbb{R}^O\right\}_{i=1}^N$ *be a set of points in the original space* $\mathbb{R}^O$. *Then, the transformations A1, A2*, A3 and A4, where A2* is the sigmoid non-linearity, do not preserve the structure of the data described neither by the Euclidean metric nor by the cosine metric.*

Proof Sketch. This proposition is proved by counterexample. The full proof is available in Appendix A.8. ■

Proposition 4. *Let us consider the sparse filtering algorithm implemented using a soft ReLU non-linearity* $ReLU(x) = \max(\epsilon, x)$, *where* ϵ *is a small negligible value (for instance,* $\epsilon = 10^{-8}$*). Let* $\left\{\mathbf{X}^{(i)} \in \mathbb{R}^O\right\}_{i=1}^N$ *be a set of points in the original space* $\mathbb{R}^O$. *Then, the transformations A1, A2*, A3 and A4, where A2* is the ReLU non-linearity, do not preserve the structure of the data described neither by the Euclidean metric nor by the cosine metric.*

Proof Sketch. This proposition is proved by counterexample. The full proof is available in Appendix A.9. ■

The non-preservation of the Euclidean metric is not surprising and it is due to the fact that the normalization in step A4 does not preserve Euclidean distances. The non-preservation of cosine distances, collinearity or cosine neighborhoodness is instead due to the sigmoid and ReLU non-linearities in step A2, since these non-linearities do not determine a folding of the space in the same way of the absolute-value function.

Despite this result, it may still be possible to implement sparse filtering with alternative non-linearities in order to preserve other types of structures. It is important that the preservation properties of alternative non-linearities agree with the structure preserved by the other steps of sparse filtering (A1, A3, A4). From a theoretical perspective, the absolute-value non-linearity is a good choice for the sparse filtering algorithm, in that it preserves the common property of collinearity which is also preserved by all the other steps of the algorithm, therefore guaranteeing the preservation of the overall structure defined by cosine neighborhoodness.

3.12. Bounds on probability of preserving Euclidean neighborhoodness

Interestingly, despite the fact that sparse filtering cannot guarantee the preservation of the Euclidean metric, it is still possible to determine probabilistic bounds on the preservation of the Euclidean neighborhoodness under very simplified assumptions on the dimensionality of the original space $\mathbb{R}^O$ and the region of space within which the samples $\mathbf{X}^{(i)}$ may be drawn.

Theorem 5. *Let* $\mathbf{X}^{(1)} \in \mathbb{R}^O$ *be a point in the original space* $\mathbb{R}^O$ *and let* $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}$ *be a representation filter centered on* $\mathbf{X}^{(1)}$, *that is,* $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}\left(\mathbf{X}^{(1)}\right) = 0$. *Let us now consider a point* $\mathbf{X}^{(2)} \in \mathbb{R}^O$ *within the same representation cone, that is, a point such that* $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}\left(\mathbf{X}^{(2)}\right) \leq \epsilon$ *for an arbitrarily small* $\epsilon \in \mathbb{R}, \epsilon > 0$.

Let us assume that: (i) points $\mathbf{X}^{(i)} \in \mathbb{R}^O$ *distribute in a limited region of space bounded by a hyper-sphere of radius M; and, (ii) points* $\mathbf{X}^{(i)} \in \mathbb{R}^O$ *distribute uniformly in this limited region of space.*

Then, given that $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}\left(\mathbf{X}^{(2)}\right) \leq \epsilon$, *it follows:*

$$\frac{O\delta}{\left(\frac{M}{m}\right)^{O-1}} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)} \leq P\left(D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] \leq \delta\right) \leq \frac{O\delta}{m} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)},$$

where $\delta \in \mathbb{R}, \delta > 0$ *defines the neighborhood of* $\mathbf{X}^{(1)}$, *m is the distance of* $\mathbf{X}^{(1)}$ *from the origin, and* $\Gamma(\cdot)$ *is the gamma function.*

Proof Sketch. This proposition is proved geometrically, evaluating the limit of the ratio between the volume of a representation filter and the neighborhood of the point when the dimensionality tends to infinity. The full proof is available in Appendix A.10. ■

Notice that this proof is based on two simplified assumptions. First, the region of the original space in which a point $\mathbf{X}^{(i)}$ can fall is limited; this assumption is reasonable because, practically, the range of any feature is always bounded within a certain interval. Second, a point $\mathbf{X}^{(i)}$ has a uniform probability of falling anywhere within the area defined by the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}$; this is clearly a simplified assumption because the pdf of the data $p(X)$ may have a very irregular distribution within the area defined by the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}^{(1)}}$; however, assuming a uniform distribution, which is a distribution that maximizes our uncertainty, seems a reasonable choice. If these two assumptions are accepted, approximate bounds can be computed to evaluate the probability that sparse filtering will preserve relationship of Euclidean neighborhoodness.

3.13. Sparse filtering for representation learning

Given the above results, we may now interpret sparse filtering as a soft clustering algorithm for representation learning.

Indeed, we may state that sparse filtering implicitly makes all the assumptions made by traditional soft clustering algorithms (see Section 2.1): (i) it aims at discovering less noisy representations $\mathbf{Z}^{(i)}$ whose pdf $p(Z)$ may automatically be closer to the true stochastic generating process with pdf $p(X^*)$; (ii) it expects the true pdf $p(X^*)$ to have a stronger correlation to the labels $\mathbf{Y}^{(i)}$; (iii) it models the true pdf $p(X^*)$ with a mixture model whose components are related to the basis vectors $\{\mathbf{e}_j\}_{j=1}^L$; and, (iv) it relies on the cosine metric to evaluate relationships of neighborhoodness in the original space $\mathbb{R}^O$. From this perspective, we can interpret the dimensionality of the learned space as the number of clusters for soft clustering, the basis vectors as the cluster centroids in a space described by the cosine metric, the pursuit of the basis as the sequential process of update of the location of the centroids, and the learned representations $\mathbf{Z}^{(i)}$ as the (stochastic) degree of membership of the original data samples $\mathbf{X}^{(i)}$ to each cluster.

Given this interpretation, we can align and meaningfully compare sparse filtering with other soft clustering algorithms for representation learning that use different metrics. The choice of an appropriate metric is critical for a distance-based clustering algorithm (Xing, Ng, Jordan, & Russell, 2003), and it expresses our understanding on which spatial directions encode relevant changes (Simard, LeCun, Denker, & Victorri, 1998). It is natural then to compare sparse filtering with other standard algorithms which adopt the Euclidean metric to explain the data. Preserving the relationships of neighborhoodness under the Euclidean metric means preserving the information conveyed by the pdf $p(X)$ in the representation space defined by the Cartesian product of the random variables $X_1, X_2, \ldots, X_O$, while preserving the relationships of neighborhoodness under the cosine metric means preserving information conveyed by the pdf $p(X)$ in the representation space defined by the projection into polar (or hyper-spherical) coordinates of the random variables $X_1, X_2, \ldots, X_O$. Choosing one metric instead of the other depends on our expectation whether the structure of the data with respect to a set of labels $p(Y|X)$ is better explained by an Euclidean structure or by a radial structure.

4. Empirical validation

Based on the theoretical analysis provided in the previous section, we conduct a set of simulations aimed at verifying our theoretical results empirically. In order to make our results visualizable and easily understandable, we first conduct simple simulations in low dimensions; experiments in higher dimensions generalize our results but they do not add anything conceptually new to our conclusions. We further validate our theoretical findings with a number of benchmark data sets pertaining to real-world applications.

4.1. Properties of sparse filtering

First, we run simulations on elaborately designed toy data sets in order to validate our basic understanding of sparse filtering. These simulations aim at verifying: (i) the property of homo-representation of collinear points (see Section 3.6); (ii) the usefulness of representation filters (see Section 3.10); and, (iii) the dynamics of pursuit of basis (see Section 3.9).

We generate a random set of data X of three samples ($N = 3$) in two-dimensional space ($O = 2$). Each point is generated using spherical coordinates: the radial distance ρ is sampled from a uniform distribution $Unif\ (-5, 5)$; the angular coordinate θ is set to $\frac{\pi}{3}$ for the first two points and sampled from a uniform distribution $Unif\ (0, \pi)$ for the third point. A sparse filtering module is trained on this data set in order to learn a new representation of the data in two dimensions ($L = 2$). After training, we create a dense mesh of points $\mathbf{X}'$ in the original representation space $\mathbb{R}^O$; we project each point $\mathbf{X}'$ to its representation $\mathbf{Z}'$ in the learned representation space $\mathbb{R}^L$, and we compute the distance from each basis vector $\mathbf{e}_j$ in $\mathbb{R}^L$. The plot of each representation cone is then shown as a two-dimensional contour plot in the original space $\mathbb{R}^O$.

Fig. 2 shows the state of sparse filtering before training. From the plots 2(b) and 2(d) we can immediately verify the property of homo-representation of collinear points; indeed, in the learned space $\mathbb{R}^L$ the collinear points occupy the same location and their matrix representation is the same. From the plots 2(e) and 2(f) we can verify the existence of representation filters in the original space $\mathbb{R}^O$ and appreciate how points in the original space are mapped onto basis vectors of the learned space. Notice that, at this point, after the random initialization of the weight matrix $\mathbf{W}$, the quality of the representations generated by the untrained sparse filtering module is far from satisfactory.

Fig. 3 shows the state of sparse filtering at the end of training. From the plots 3(b) and 3(d) we can see that the trained sparse filtering module has found an optimal solution that maps all the points to basis vectors; as expected, the collinear points are mapped to the same basis vector, while the third point is mapped to the remaining basis vector. From the plots 3(e) and 3(f) we can validate our intuition about the pursuit of basis; indeed, training corresponded to a rotation of the representation filters in order to center them on the available samples.

4.2. Preservation of cosine neighborhoodness

Next, we run more simulations on other toy data sets in order to validate the properties of data structure preservation in sparse filtering. These simulations aim at verifying: (i) that sparse filtering preserves a structure defined by cosine neighborhoodness (see Section 3.8); and, (ii) that the absolute-value non-linearity is crucial in preserving structure and substituting it with other non-linearities negates this property (see Section 3.11).

We generated a random set of data $\mathbf{X}$ of nine samples ($N = 9$) in two-dimensional space ($O = 2$). Each point is generated by randomly sampling its spherical coordinates. The first three points have a radial distance $\rho \sim Unif\ (-2, 0)$ and an angular coordinate $\theta \sim Unif\left(\frac{\pi}{9} - \eta, \frac{\pi}{9} + \eta\right)$; the following three points have a radial distance $\rho \sim Unif\ (0, 3)$ and an angular coordinate $\theta \sim Unif\left(\frac{2\pi}{9} - \eta, \frac{2\pi}{9} + \eta\right)$; the last three points have a radial distance $\rho \sim Unif\ (2, 4)$ and an angular coordinate $\theta \sim Unif\left(\frac{4\pi}{9} - \eta, \frac{4\pi}{9} + \eta\right)$. The parameter η is meant to represent a form of noise and its value is set to $\eta = \frac{\pi}{45}$. In this way, we generate three clusters of points, such that the cosine distances among the points belonging to the same cluster are small, while the cosine distances among points belonging to different clusters are large. Three implementations of sparse filtering with different non-linearities (absolute-value, sigmoid, and ReLU[6]) are used to learn a new representation of the data in two dimensions ($L = 2$).

Fig. 4 shows the state of the modules of the three implementations of sparse filtering at the end of the training. From the plots 4(a)–4(c) we can immediately verify that sparse filtering with an absolute-value non-linearity preserves cosine neighborhoodness. The plots of representation filters show that points with similar angular coordinates fall within the same representation filter. The matrix plot shows that points with similar angular coordinates are projected onto very similar representations; in other words, points that originally had a small cosine distance are projected onto almost identical representations. On the other hand, from plots 4(d)-4(i) we can easily see that sparse filtering with an alternative non-linearity does not preserve cosine neighborhoodness. The plots of representation filters show that the sigmoid and the ReLU non-linearity do not induce representation cones, but, instead define large regions of the original space to be mapped onto a basis vector; several points are thus indistinctly mapped onto a basis vector. The matrix plots show that the representations computed by these alternative sparse filtering modules are not related to the original cosine distances anymore; points originally belonging to the same cluster are mapped to opposite representations, and, vice versa, points originally belonging to different clusters are mapped to identical representations.

4.3. Sparse filtering for representation learning

In the following set of simulations, we compare sparse filtering against another unsupervised algorithm in order to show under which conditions sparse filtering is a good choice for processing data. These simulations aim at verifying the following intuitive implication: if the structure of the data with respect to a specific set of labels $p(Y|X)$ is better explained by the cosine metric, then sparse filtering is likely to be a good option for unsupervised learning.

In our comparison, we measure sparse filtering against the soft k-means algorithm (MacKay, 2003). We choose this algorithm for the following reason: (i) like sparse filtering, the soft k-means algorithm is a soft clustering algorithm producing sparse representations; (ii) the algorithm is based on the Euclidean metric, thus providing a different interpretation of the data from sparse filtering; and, (iii) k-means is a well-known and easy-to-interpret algorithm (even if, analogous results may be obtained with other algorithms, such as Gaussian mixture models or sparse auto-encoders).

[6] ReLU has been implemented in a soft version, like the absolute-value: $softReLU(x) = \begin{cases} x & if\ x > 0 \\ \epsilon & otherwise \end{cases}$ for $\epsilon = 10^{-8}$.

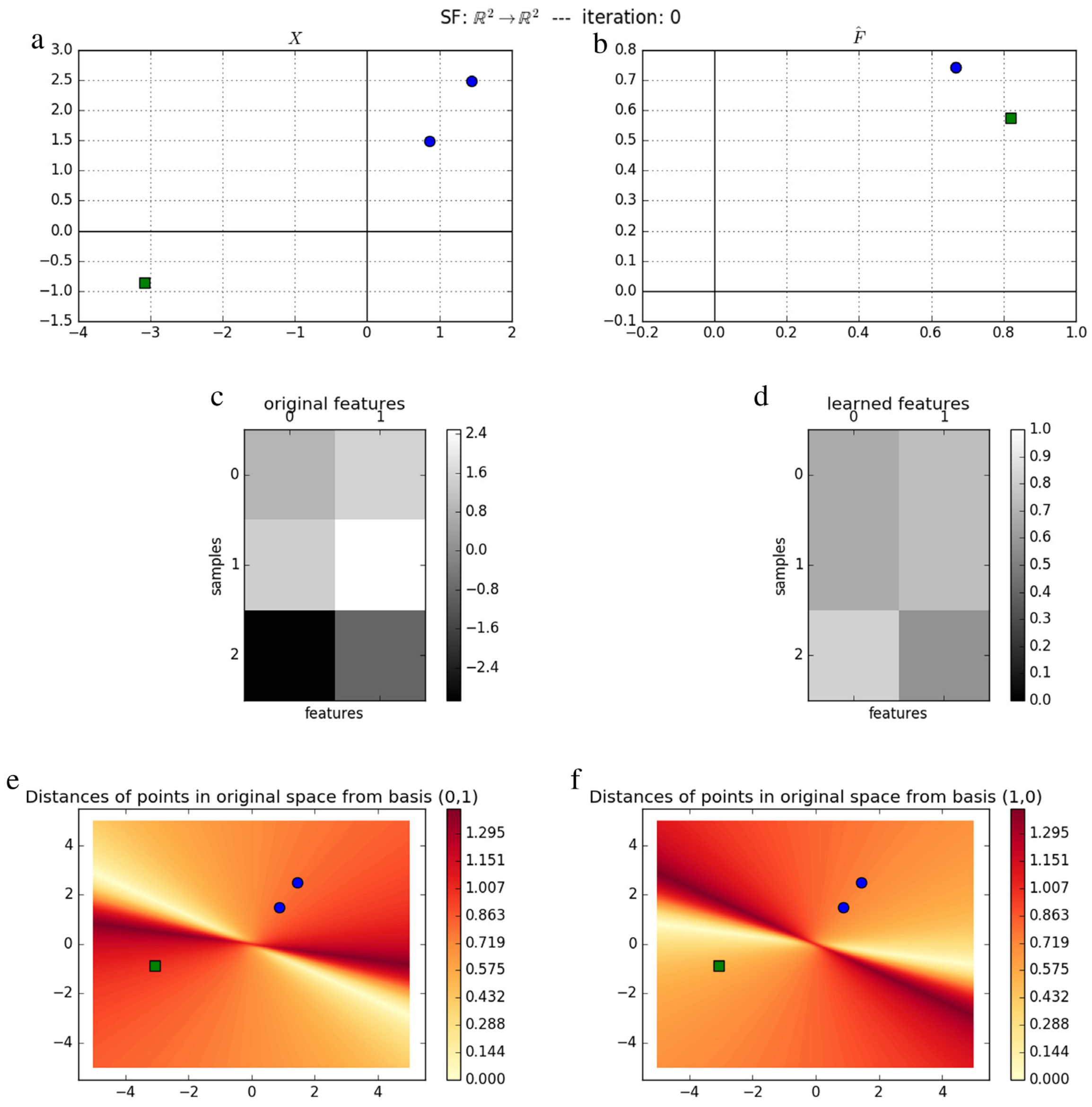

Fig. 2. Experimental validation of the properties of sparse filtering (homo-representation of collinear points, representation filters). Data is generated as explained in the text (the blue circle dots represent collinear points). (a) Data in the original representation space $\mathbb{R}^O$; (b) data in the learned representation space $\mathbb{R}^L$; (c) matrix plot of the original data $\mathbf{X}$; (d) matrix plot of the learned representations $\mathbf{Z}$; (e) plot of first representation filter showing distances from the basis vector $\mathbf{e}_1 = [0, 1]^T$; (f) plot of the second representation filter showing distances from the basis vector $\mathbf{e}_2 = [1, 0]^T$.

To validate our hypothesis, we generate two data sets, $\mathbf{X}_{Euclid}$ and $\mathbf{X}_{cosine}$. The data set $\mathbf{X}_{Euclid}$ contains data where $p(Y|X)$ is explained by the Euclidean metric. It is composed of nine samples ($N = 9$) in two dimensions ($O = 2$) sampled from three multivariate normal distribution. The first three points are sampled from $\mathcal{N}\left(\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}.05 & 0\\0 & .05\end{bmatrix}\right)$; the second three points are sampled from $\mathcal{N}\left(\begin{bmatrix}2\\-1\end{bmatrix}, \begin{bmatrix}.05 & 0\\0 & .05\end{bmatrix}\right)$; the last three points are sampled from $\mathcal{N}\left(\begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}.05 & 0\\0 & .05\end{bmatrix}\right)$. The data set $\mathbf{X}_{cosine}$ contains data where $p(Y|X)$ is explained by the cosine metric. The data is generated following the same protocol used in the simulation in Section 4.2. Sparse filtering is used to learn a new representation of the data in three dimensions ($L = 3$).

From Fig. 5, we can see that our understanding of sparse filtering is correct: if $p(Y|X)$ is better explained by the cosine metric, then sparse filtering produces a good representation; otherwise, if $p(Y|X)$ is better explained by the Euclidean metric, then it is reasonable to opt for a different unsupervised learning algorithm, such as soft k-means. In the case of the data set with Euclidean structure, plot 5(b) shows that sparse filtering is not able to preserve the identity of the generating clusters, and indeed it maps samples from the first and the third clusters onto the same representation (because of their collinearity); instead, plot 5(c) shows that soft k-means algorithm maps points from different clusters to different representations. In contrast, in the case of the data set with cosine structure, plot 5(e) shows that sparse filtering preserves the identity of the generating clusters, while plot 5(f) shows that the soft k-means algorithm is unable to map samples from different clusters onto consistent representations.

4.4. Sparse filtering on real data sets

In this last set of simulations we apply our discoveries about sparse filtering to real-world data sets to further verify our results. Once again, these experiments aim at validating the connection between the radial structure of the data and the success of sparse filtering. In the first simulation, we extend the result that we

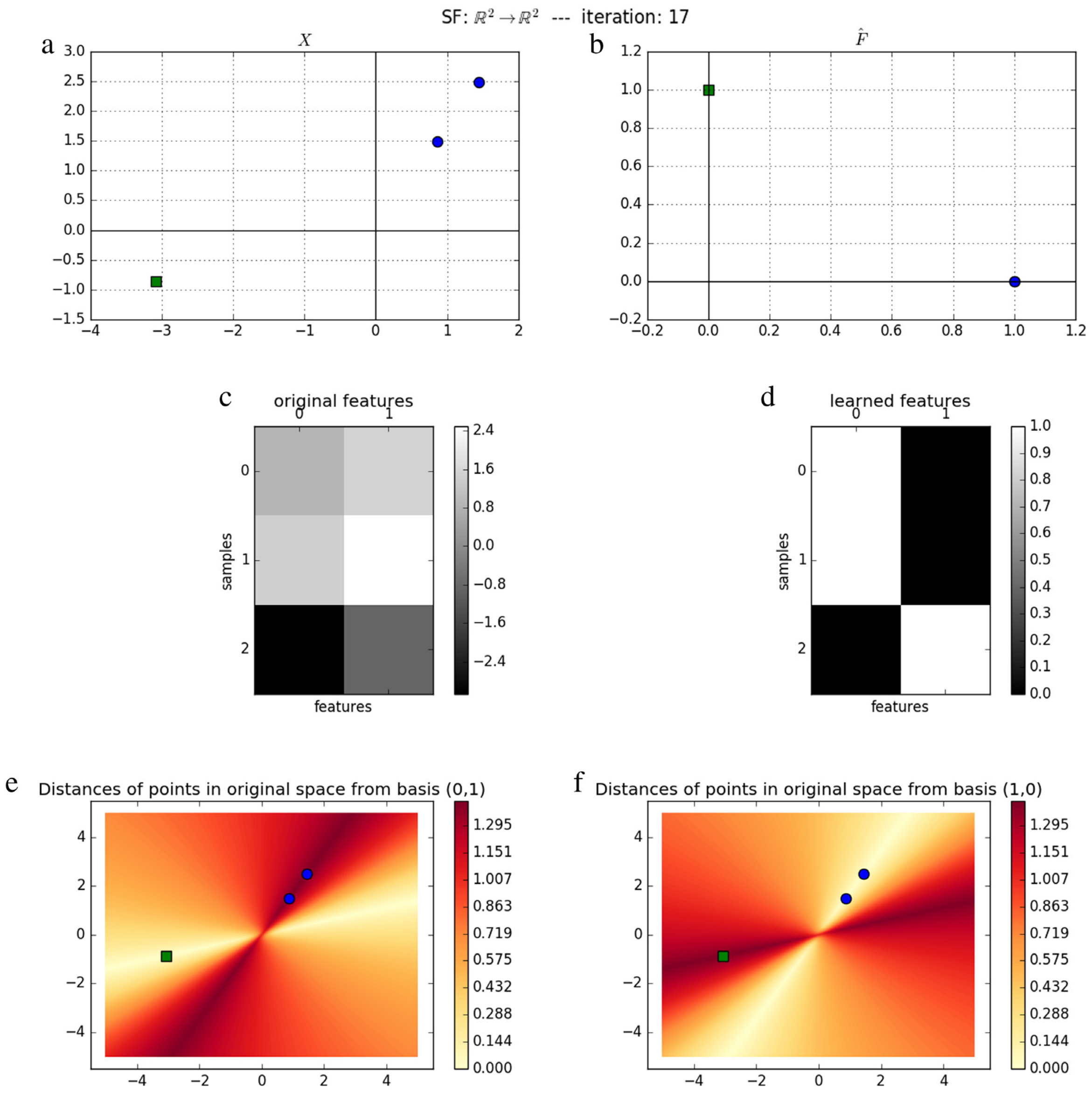

Fig. 3. Experimental validation of the properties of sparse filtering (pursuit of basis). Data is generated as explained in the text. The meaning of the subplots is the same as in Fig. 2.

proved in Section 4.3 for toy data sets to real data sets; that is, we verify the direct implication: *if* the structure of the data with respect to a specific set of labels $p(Y|X)$ is better explained by the cosine metric, *then* sparse filtering is likely to be a good option for unsupervised learning. In the second simulation, we validate, instead, the reverse implication: *if* sparse filtering happens to be a good option for unsupervised learning, *then* the structure of the data with respect to a specific set of labels $p(Y|X)$ is likely to be better explained by the cosine metric.

When dealing with real data sets, it is very challenging to assess the structure of the data. With few samples in low dimensions and with the simplified assumption that all the data belonging to a given class are generated by a single highly localized cluster, a simple visualization or a computation of distances is enough to understand which metric is underlying the data. However, when we consider real data sets, we have to deal with a large number of samples in high dimensions, and with the fact that samples belonging to the same class may be generated by different clusters spread throughout the space. In order to explore high-dimensional data in the original space before any normalization, we decided to rely on the k-nearest neighbors algorithm (KNN). We implemented two versions of KNN, one selecting k neighbors according to the Euclidean distance and one selecting k neighbors according to the cosine distance.[7] If $p(Y|X)$ is better explained by the Euclidean distance, we expect KNN with the Euclidean metric to provide better results; alternatively, if $p(Y|X)$ is better explained by the cosine distance, we expect KNN with the cosine metric to provide better results.

Berlin Emotional data set. The Berlin Emotional (EMODB) data set is a well-known audio data set in the emotion recognition community (Burkhardt, Paeschke, Rolfes, Sendlmeier, & Weiss, 2005); it contains recordings of ten German actors expressing seven different types of emotions. We opted for this data set to validate the direct implication between data structure and effectiveness of sparse filtering for the following reasons. (i) Samples in EMODB naturally lend themselves to alternative labelings; the same data

[7] The KNN using cosine distance has been implemented relying on the "trick" that the cosine distance between vectors $\mathbf{u}, \mathbf{v}$ is the same as the Euclidean metric on the ℓ_2-normalized vectors. Therefore, we perform an ℓ_2-normalization of each data sample and then we run KNN with Euclidean distance, re-using off-the-shelf KNN code optimized for the Euclidean metric.

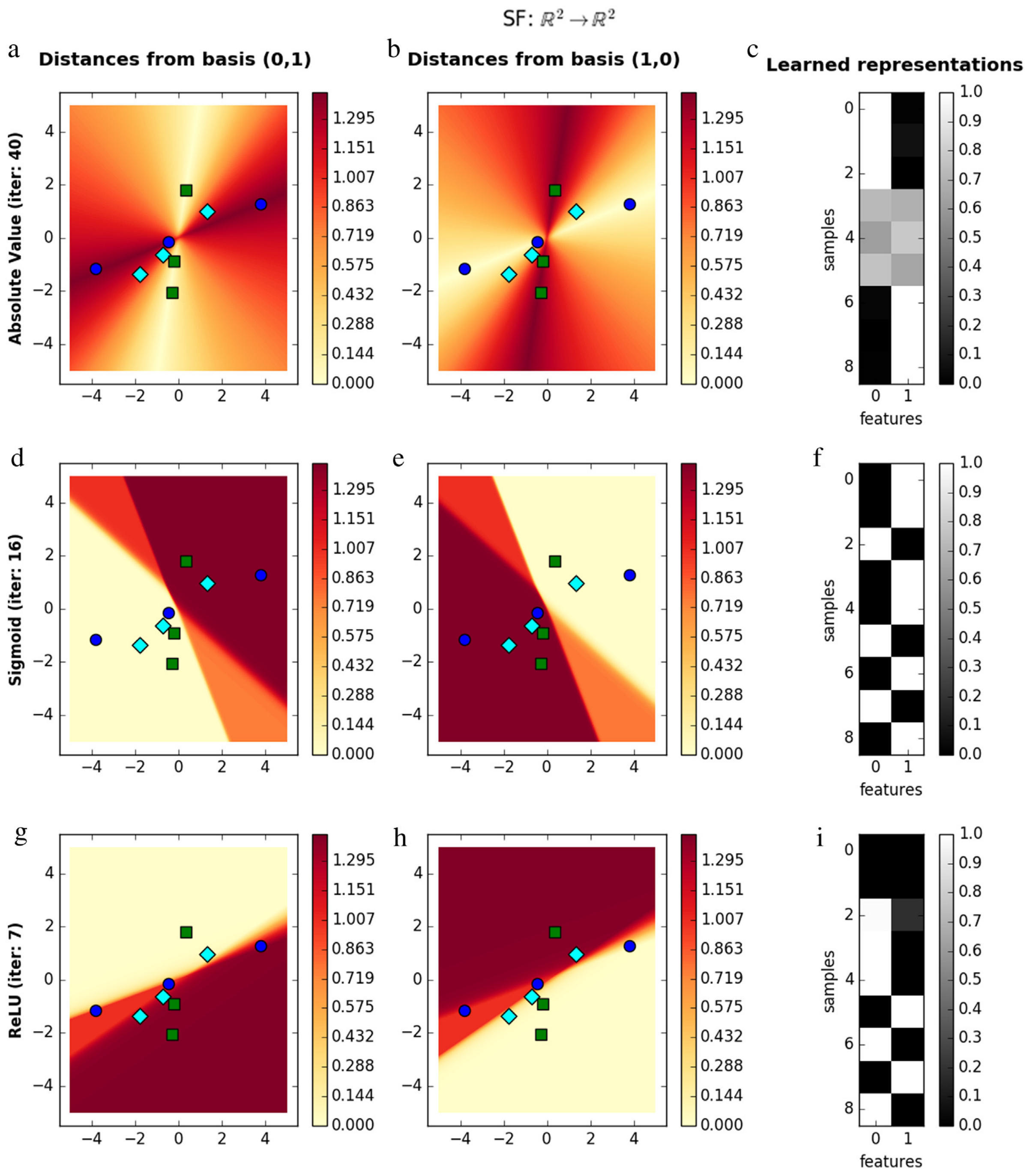

Fig. 4. Experimental validation of the preservation of cosine neighborhoodness. Data is generated as explained in the text (first set of points as blue circle dots, second set of points cyan diamond dots, third set of points as green square dots). (a, d, g) Plot of the first representation filter showing distances from the basis vector $\mathbf{e}_1 = [0, 1]^T$, respectively for the sparse filtering with absolute-value, sigmoid, and ReLU non-linearity; (b, e, h) plot of the second representation filter showing distances from the basis vector $\mathbf{e}_2 = [1, 0]^T$, respectively for the sparse filtering with absolute-value, sigmoid, and ReLU non-linearity; (c, f, i) matrix plot of the learned representations $\mathbf{Z}$, respectively for the sparse filtering with absolute-value, sigmoid, and ReLU non-linearity.

may be used both for speaker recognition (using subject labels) and for emotion recognition (using emotional labels). (ii) The same set of Mel-frequency cepstrum (Childers, Skinner, & Kemerait, 1977) coefficient (MFCC) features may reasonably be used both for speaker recognition and for emotion recognition (Schuller, Batliner, Steidl, & Seppi, 2011; Wu, Parsons, & Narayanan, 2010). Thus, the same features we can be used to explore the data under different labeling.

We first explore the structure of the data with respect to the two different labeling systems in order to evaluate whether the Euclidean distance or the cosine distance better explains the structure of the data. We run the KNN algorithm with different values of neighbors ($k = \{2, 3, 5, 7, 10, 15, 20, 25, 50, 75, 100\}$); for each configuration of KNN, fifty simulations are executed; in each simulation the data set is randomly partitioned into a training data set (900 samples) and a test data set (311 samples); KNN is then trained and tested using one of the two available metrics.

After this analysis, we use both an Euclidean-based unsupervised learning algorithm, Gaussian mixture model (Bishop, 2007), and a cosine-based unsupervised learning algorithm, sparse filtering, to project the data into an L-dimensional space. We opted for the Gaussian mixture of models (GMM) algorithm because it is

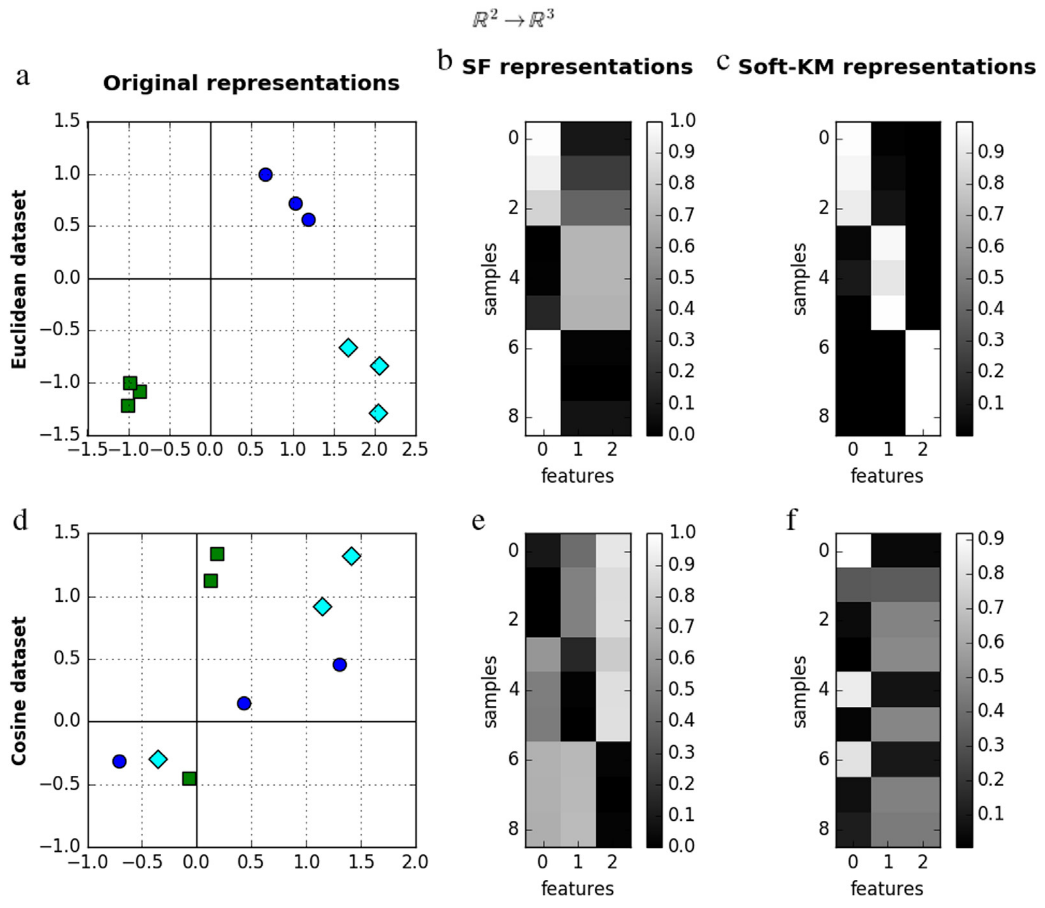

Fig. 5. Representation data with Euclidean and cosine data structure. Data is generated as explained in the text (first set of points as blue circle dots, second set of points cyan diamond dots, third set of points as green square dots). (a, d) Samples in the original space; (b, e) matrix plot of the representations learned by sparse filtering; (c, f) matrix plot of the representations learned by soft k-means.

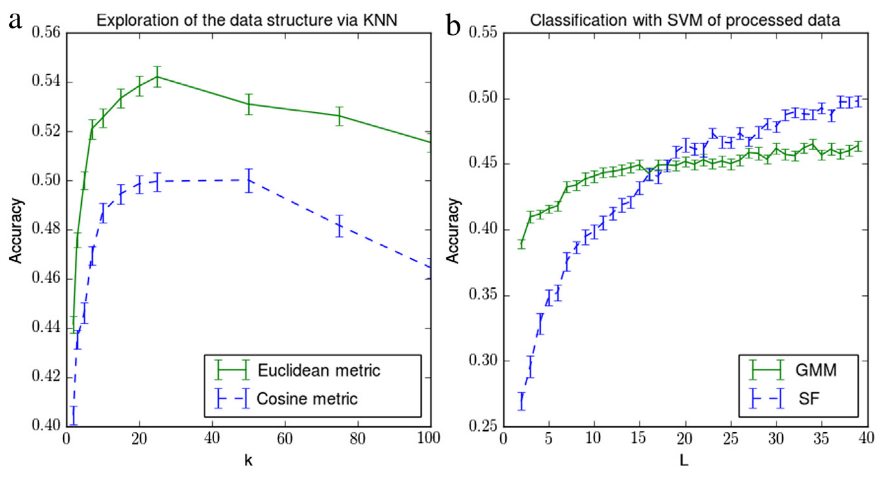

Fig. 6. Analysis of the data structure and the classification of the EMODB data set with respect to emotion labels. Classification is performed as explained in the text. (a) Exploration of the data via KNN with Euclidean metric (green continuous line) and with cosine metric (blue dashed line); (b) Classification using a linear SVM after processing with a GMM algorithm (green line) and with sparse filtering (blue line). The plot shows the average accuracy and the standard error of SVM (over fifty simulations).

based on the Euclidean metric and yields better performance than the soft k-means algorithm. After processing the data, we then run a simple linear SVM classifier on the processed data and we analyze how our observations on the structure of the data relate with the actual classification performance. We consider several values of dimensionality ($L = \{2, 3, \ldots, 40\}$); for each configuration, fifty simulations are executed; as before, in each simulation the data set is randomly partitioned into a training data set (900 samples) and in a test data set (311 samples).

Fig. 6(a) shows that the structure of EMODB data with respect to emotional labels is better explained by the Euclidean distance. This result is further confirmed by the classification with the linear SVM module in Fig. 6(b). Classification using the GMM-processed data with low learned dimensionality ($L \leq 15$) returns an accuracy that is significantly better than using sparse filtering-processed data (Wilcoxon signed-rank test, p-value $P = 5 \cdot 10^{-85}$); however, in higher dimensions the classification with sparse filtering-processed data approaches and overtakes the accuracy obtained

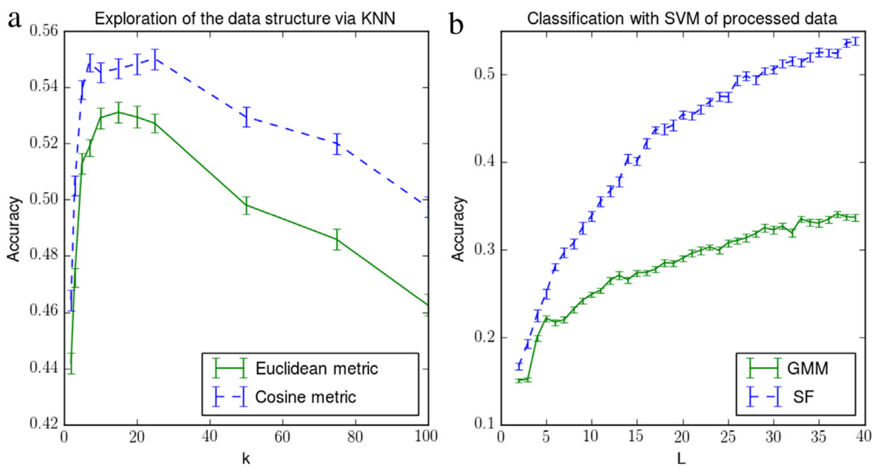

Fig. 7. Analysis of the data structure and the classification of the EMODB data set with respect to subject labels. The meaning of the subplots is the same as in Fig. 6.

using GMM-processed data. In general, in low dimensions, the Euclidean structure assumed by GMM explains the data better; in high dimensions, sparse filtering provides good results (most likely thanks to the property of sparsity), but the gap between the accuracy provided by the two representations remains limited. On the other hand, Fig. 7(a) shows that the structure of EMODB data with respect to the speaker identity labels is better explained by the cosine distance. This result is further confirmed by the classification with the linear SVM module in Fig. 7(b). Classification using the sparse filtering-processed data returns, for all learned dimensionality, an accuracy that is significantly better than GMM-processed data (Wilcoxon signed-rank test, *p*-value $P = 4 \cdot 10^{-307}$). The assumption of the cosine metric allows sparse filtering to explain the data much better, as is evident from the large gap between the accuracy provided by the two representations.

These results confirm a connection between the radial structure of the data with respect to a set of labels and the usefulness of sparse filtering.

Kaggle Black Box Learning Challenge data set. The Kaggle Black Box Learning Challenge (KBBLC) data set is a visual data set made up of obfuscated images of house numbers; the original images are taken from the well-known Street View House Numbers (SVHN) data set (Netzer, Wang, Coates, Bissacco, Wu, & Ng, 2011). Each sample in the KBBLC data set contains a single obfuscated digit and it is accompanied by a label specifying the value of the digit. We opted to validate the reverse implication between data structure and effectiveness of sparse filtering on this data set for the following reasons. (i) Sparse filtering provided state-of-the-art performance in the competitive KBBLC contest, thus showing that sparse filtering was a particularly suitable choice for this data set. (ii) The KBBLC data set is available with labels. During the challenge the authors provided obfuscated data without labels; however, after the challenge they revealed the original source of the data[8] and they released the code they used for obfuscation.[9] Thanks to this information, we were able to retrieve a large amount of data and obfuscate it, and thus recreate the original conditions of the challenge. However, differently from the challenge, we retain the labels in order to explore the structure of the data. (iii) During the challenge, the original samples from the data sets were processed without undergoing operations of windowing or convolution. Since sparse filtering was directly applied to the samples, we can analyze the structure of the samples straightforwardly. This condition is not always true. If we consider other image data sets on which sparse filtering provided good results, such as CIFAR-10 (Krizhevsky & Hinton, 2009) or STL-10 (Coates et al., 2011), sparse filtering was not applied to the original samples but to random patches extracted from the images; in this case, we should not analyze the data structure of the original samples, but the data structure of the patches. However, patches are not labeled, which hinders our ability to carry out an analysis of the data structure.

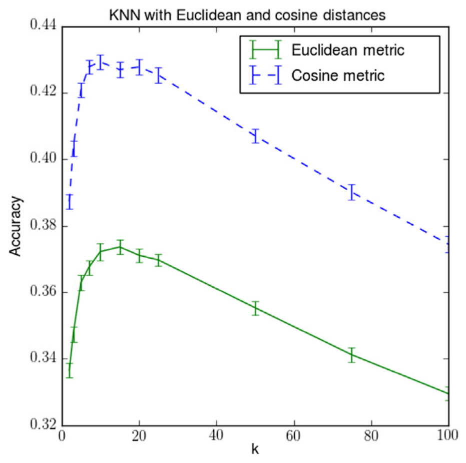

Fig. 8. Analysis of the data structure of the Kaggle Black Box Learning Challenge data set. The KNN with Euclidean metric (green continuous line) and with cosine metric (blue dashed line) has been used to explore the structure of the data. The plot shows the average accuracy and the standard error of KNN (over five simulations).

In exploring the structure of the data (with respect to the digit labels), we aim at evaluating whether the Euclidean distance or the cosine distance better explains $p(Y|X)$. We run the KNN with the same settings as in the previous experiment. In each simulation a random subset of 10 000 samples from the data set was selected and then partitioned into a training data set (9000 samples) and a test data set (1000 samples). KNN was then trained and tested using one of the two available metrics.

Fig. 8 confirms our intuition. For all the different values of *k* we considered, the cosine distance proved to be a better metric to explain the structure of the data in the Kaggle Black Box Learning Challenge. This provides an explanation why sparse filtering proved so useful with the KBBLC data, when compared to other standard unsupervised learning algorithms, especially those based

8 http://ufldl.stanford.edu/housenumbers/.

9 https://www.kaggle.com/c/challenges-in-representation-learning-the-black-box-learning-challenge/forums/t/5167/the-data.

on the Euclidean metric. This result agrees with the fact that the Euclidean metric is not a suitable metric for measuring distances among samples of digits represented in the pixel space (Simard et al., 1998).

5. Discussion

Our theoretical and empirical analysis showed that the standard sparse filtering algorithm implemented with an absolute-value non-linearity preserves the data structure explained by the cosine neighborhoodness In our experiments, we have shown that, when the relevant structure of the data has a radial structure, then sparse filtering may be expected to perform significantly better than the standard Euclidean-based alternatives. Indeed, sparse filtering may be seen as an algorithm approximately transforming cosine distances in the original space into Euclidean distances in the representation space.

It is normally assumed that the data points $\mathbf{X}^{(i)}$ are best explained as samples from a multivariate random variable $X = (X_1, X_2, \ldots, X_O)$, where each random variable X_j describes a component $\mathbf{X}_j^{(i)}$. However, given the data points $\mathbf{X}^{(i)}$, it is possible to assume that the generating process is better described by a multivariate random variable $X' = \left(X'_1, X'_2, \ldots, X'_{O-1}\right)$, where each random variable describes an angular coordinate θ_j of $\mathbf{X}_j^{(i)}$. Sparse filtering tries to preserve the information about the $O-1$ angular coordinates θ_i, discarding the information about the radial coordinate ρ. If $p(Y|X)$ is better explained in terms of radial coordinates, then sparse filtering is a very reasonable choice for unsupervised representation learning.

Our study allow us to conclude that our original thesis is correct: sparse filtering satisfies both the informativeness principle and the infomax principle. In particular, the informativeness principle is satisfied through the adoption of the proxy of sparsity, as shown in Section 3.2. The infomax principle is satisfied through the preservation of a precise structure underlying the data, that is, the radial structure of the data. Mutual information between the original representations $\mathbf{X}^{(i)}$ and the learned representations $\mathbf{Z}^{(i)}$ is retained when the structure of the data is explained by the cosine metric, that is, in an ideal case, when all the information is carried by the angular coordinates of the data, as demonstrated in Section 3.8. Indeed, the mutual information between the original and the learned representations can be formally expressed as: $I[X, Z] = H[X] - H[X|Z]$. Given that the entropy of the distribution of the data $p(X)$ is fixed, the only way to maximize the mutual information is by minimizing the conditional entropy $H[X|Z]$. Since the representation $\mathbf{Z}$ preserve all the information about the angular coordinates of the original data, the uncertainty about $\mathbf{X}$ given $\mathbf{Z}$ is minimized if the structure of the data has indeed a radial structure.

Following this reasoning, we suggested an interpretation of sparse filtering as an unsupervised soft clustering algorithm based on the cosine metric. This perspective allowed us to contrast the results of sparse filtering with other standard algorithms for clustering based on the Euclidean metric and conclude that sparse filtering does not provide a better processing of the data in absolute terms, but instead it offers an alternative interpretation of the data based on a different metric.

While in our experiments, we were aware a priori of the metric underlying synthetic data sets, in a real-world setting such knowledge may not be available and simple exploratory analysis of the data (using, for instance KNN) may be unsuitable. Sparse filtering, thanks to its scalability and its efficiency, could also be used to infer the data structure underlying the data. The usefulness of polar coordinates in several scientific fields and physical applications may suggest that interpreting data according to cosine distance could be a sensible choice.

Additionally, we proved that, in high dimensions, sparse filtering can still probabilistically preserve Euclidean distances. This is justified by the fact that, under the assumptions we made, the probability that unrelated points with high Euclidean distance will have the same angular coordinates θ_i can be bounded (Section 3.12).

Interestingly, our study of sparse filtering as an unsupervised learning algorithm shares a similar methodology with the very recent work by McNamara, Ong, and Williamson (2016). Recasting our analysis in their framework we can demonstrate that, with high probability, sparse filtering working on radial data contributes to the reduction of the risk in standard supervised learners by showing that: (i) $P(X)$ has a structure explained by cosine neighborhoodness; (ii) $P(Y|X)$ share the same structure as $P(X)$; (iii) sparse filtering relies on the cosine distance; (iv) a supervised learner, such as SVM, can exploit the new Euclidean structure in the learned representations.

6. Concluding remarks

In this paper, we have explained *why* sparse filtering works (by proving its property of preservation of cosine neighborhoodness) and *when* it should be expected to provide useful representations (by considering the data structure of the samples).

Our theoretical analysis and simulations were not designed to show that sparse filtering is able to provide state-of-the-art performance against other algorithms, but, instead, to show how the implicit assumptions and constraints of sparse filtering make it better suited for certain scenarios instead of others. In particular, for the standard sparse filtering algorithm implemented using an absolute-value non-linearity we demonstrated that its success is tied to the data having a structure explained by cosine neighborhoodness. Consistently with the no-free lunch theorem (Wolpert & Macready, 1997), we reached the conclusion that sparse filtering is not a better algorithm than other Euclidean-based clustering algorithms, but that there is a specific set of problems (in which $p(Y|X)$ is explained by the cosine metric) where the performance of sparse filtering is excellent, balanced by a set of problems (in which $p(Y|X)$ is explained by the Euclidean metric or other metrics) where its performance is less outstanding. This led us to interpret the representation of sparse filtering as a "view" of the data according to the cosine metric alternative to the more standard Euclidean view. Combining these two different views could provide representations with more discriminative power.

At the foundation of our analysis lies the understanding that sparse filtering *must* preserve some information carried by the pdf $p(X)$. Despite sparse filtering ignoring the problem of explicitly modeling the true pdf $p(X)$, we showed that the algorithm is hard-coded with an implicit constraint that guarantees the preservation of some data structure. This is clearly a specific conclusion about the particular algorithm of sparse filtering, but we can expect that this principle will be applicable to the whole class of feature distribution learning algorithms. We might expect that any feature distribution learning algorithm, in order to be successful, must take into account, through constraints or priors, the problem of preserving the mutual information between the original representations $\mathbf{X}^{(i)}$ and the learned representations $\mathbf{Z}^{(i)}$.

Being aware of this requirement can give a precious contribution in the future research and design of new feature distribution learning algorithms; for instance, it may prevent us from considering in sparse filtering alternative non-linearities that do not preserve any interesting structure (such as, the ReLU function) or it may help us to avoid solutions that, being unable to preserve any structure of the data, are bound to produce unsatisfactory representations. Ongoing research is focused on discovering which

$$f_{A1}\left(\mathbf{X}^{(1)}\right) = \mathbf{I}\mathbf{X}^{(1)} = \mathbf{X}^{(1)} \qquad f_{A1}\left(\mathbf{X}^{(2)}\right) = \mathbf{I}\mathbf{X}^{(2)} = \mathbf{X}^{(2)}$$
$$f_{A2}\left(\mathbf{X}^{(1)}\right) = \left|\mathbf{X}^{(1)}\right| = \mathbf{X}^{(1)} \qquad f_{A2}\left(\mathbf{X}^{(2)}\right) = \left|\mathbf{X}^{(2)}\right| = \mathbf{X}^{(1)}$$
$$f_{A3}\left(\mathbf{X}^{(1)}\right) = \left[\frac{\mathbf{X}_j^{(1)}\sqrt{2}}{\sqrt{N}}\right] = \left[\frac{1}{\sqrt{N}}\right] \qquad f_{A3}\left(\mathbf{X}^{(1)}\right) = \left[\frac{\mathbf{X}_j^{(1)}\sqrt{2}}{\sqrt{N}}\right] = \left[\frac{1}{\sqrt{N}}\right]$$
$$f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right) = \left[\frac{\mathbf{X}_j^{(1)}\sqrt{N}}{\sqrt{L}}\right] = \left[\frac{1}{\sqrt{L}}\right] = \mathbf{Z}^{(1)} \quad f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right) = \left[\frac{\mathbf{X}_j^{(1)}\sqrt{N}}{\sqrt{L}}\right] = \left[\frac{1}{\sqrt{L}}\right] = \mathbf{Z}^{(1)}.$$

Box II.

structures may be preserved by alternative version of sparse filtering, with a particular focus on periodic structures that may be learned using trigonometric functions.

A deeper theoretical understanding of the dynamics of sparse filtering may be developed in connection with manifold learning and information geometry (Amari, 2016). The property of preservation of structure that we uncovered in this paper may be more formally explained in the framework of differential geometry, by modeling the data samples as a points on a Riemannian manifold. Relevant data structures (that we presented in terms of Euclidean or cosine distance) may then be described in terms of Riemannian metric tensors, and preservation properties may be studied in terms of preservation of these tensors.

Another promising avenue in our research is the extension of sparse filtering to semi-supervised learning. Indeed, the paradigm of feature distribution learning seems perfectly suited for the scenario in which we are provided with few labeled samples and many unlabeled samples: we may exploit the information carried by the labeled samples to better shape the feature distribution $p(Z)$, without addressing the problem of estimating the true pdf $p(X^*)$; at the same time, the constraint of sparsity would help us to not overfit, and the constraint of structure preservation would help us to preserve the information conveyed by $p(X)$. Furthermore, assuming some regularity in the original representation space, we hypothesize that we could use the information in the labeled samples to address the problem of covariate shift (Sugiyama & Kawanabe, 2012) in a semi-supervised learning scenario.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments that improved the presentation of this manuscript. The authors are also grateful to William Woof for personal communication. F. M. Zennaro's work was supported by the Kilburn Ph.D. studentship.

Appendix. Proofs

A.1. Proposition 1

Proposition 1. *Let $\left\{\mathbf{X}^{(i)} \in \mathbb{R}^O\right\}_{i=1}^N$ be a set of points in the original space $\mathbb{R}^O$. Then, the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric.*

Proof. We prove this proposition by counterexample.

Let us consider the case in which $\mathbf{X}^{(1)}$ is a vector such that $\mathbf{X}_j^{(1)} = \frac{1}{\sqrt{2}}$, $\forall j$, $1 \leq j \leq O$, $\mathbf{X}^{(2)}$ is another vector such that $\mathbf{X}^{(2)} = -\mathbf{X}^{(1)}$, $L = O$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$D_E\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right) = \sqrt{\sum_{j=1}^{O}\left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right)^2} = \sqrt{2L}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Then we get the equations in Box II.

Thus, $f_{A1:A4}\left(\mathbf{X}^{(1)}\right) = \mathbf{Z}^{(1)}$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right) = \mathbf{Z}^{(1)}$. Now, the Euclidean distance between the vectors $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)$ is:

$$D_E\left(\mathbf{Z}^{(1)}, \mathbf{Z}^{(1)}\right) = 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. ■

A.2. Lemma 1

Lemma 1. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^O$, two generic collinear vectors, and let $f : \mathbb{R}^O \to \mathbb{R}^L$ be a linear transformation defined as $f(\mathbf{u}) = \mathbf{W}\mathbf{u}$, where $\mathbf{W}$ is the matrix associated with the linear transformation. Then, $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the linear transformation f encoded by matrix $\mathbf{W}$ and let us apply it to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = \mathbf{W}\mathbf{v} = \mathbf{W}(k\mathbf{u}) = k(\mathbf{W}\mathbf{u}) = k \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

A.3. Lemma 2

Lemma 2. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, two generic collinear vectors, and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the element-wise absolute-value function $f(\mathbf{u}) = |\mathbf{u}| = \left[\left|u_j\right|\right]$. Then $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the element-wise absolute-value function f and let us apply it to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = |\mathbf{v}| = |k\mathbf{u}| = |k| \cdot |\mathbf{u}| = |k| \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

A.4. Lemma 3

Lemma 3. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, two collinear vectors whose components are all strictly positive,*[10] *and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the ℓ_2-normalization along the features. Then $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

10 Notice that we can safely make the assumption of strict positivity in sparse filtering since $\mathbf{u}$ and $\mathbf{v}$ are the output of an absolute-value function implemented as $f(x) = \sqrt{x^2 + \epsilon}$.

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the function of normalization along the features $f(\mathbf{u}) = \left[\frac{\mathbf{u}_j}{\sqrt{\sum_{\mathbf{w}=\{\mathbf{u},\mathbf{v}\ldots\}} \mathbf{w}_j^2}}\right]$. Normalizing along the features means dividing each component $\mathbf{u}_j$ by a constant c_j equal to the ℓ_2-norm of the component j across all the available vectors $\mathbf{u}, \mathbf{v}, \ldots$, that is $f(\mathbf{u}) = \left[\frac{\mathbf{u}_j}{c_j}\right] = \mathbf{c} \circ \mathbf{u}$, where $\mathbf{c}$ is the vector containing all the constants c_j and $\circ$ is the element-wise product. Let us now apply the normalization along the features to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = \mathbf{c} \circ \mathbf{v} = \mathbf{c} \circ (k\mathbf{u}) = k \cdot (\mathbf{c} \circ \mathbf{u}) = k \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

A.5. Lemma 4

Lemma 4. *Let us consider $\mathbf{u} \in \mathbb{R}^L$, a vector whose components are all strictly positive,*[11] *and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the ℓ_2-normalization along the samples. Then $f(\mathbf{u}) \in \mathbb{R}^L$ have the same angular coordinates as $\mathbf{u}$.*

Proof. Let us consider the function of normalization along the features $f(\mathbf{u}) = \left[\frac{\mathbf{u}_j}{\sqrt{\sum_j \mathbf{u}_j^2}}\right]$. Normalizing along the samples means dividing each component $\mathbf{u}_j$ by the ℓ_2-norm of the same vector $\mathbf{u}$, that is $f(\mathbf{u}) = \left[\frac{\mathbf{u}_j}{\ell_2(\mathbf{u})}\right] = \frac{1}{\ell_2(\mathbf{u})} \cdot \mathbf{u}$. Multiplying all the components of the same vector $\mathbf{u}$ by the constant $k = \frac{1}{\ell_2(\mathbf{u})}$ leaves the angular coordinates unaltered. Therefore, the angular coordinates are preserved. ■

A.6. Theorem 4

Theorem 4. *Let $\mathbf{X}^{(1)}, \mathbf{X}^{(2)} \in \mathbb{R}^O$ be two original data samples and let $\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)} \in \mathbb{R}^L$ be their representations computed by sparse filtering. If the cosine distance between the original samples is arbitrarily small $D_C\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] < \delta$, for $\delta > 0$, then the Euclidean distance between the computed representations is arbitrarily small $D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] < \epsilon$, for $\epsilon > 0$, and $\epsilon = L \cdot \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right)$, where k is a constant accounting for partial collinearity and $\ell_2\left(\tilde{\mathbf{F}}^{(i)}\right)$ is the ℓ_2-norm of the representations computed by sparse filtering after step A3. In the limit, it holds that $\lim_{\delta\to 0}\epsilon = 0$.*

Proof. In order to prove this theorem we adopt the following strategy: we compute the representations at each step of the computation (before sparse filtering, after steps A1 and A2, after step A3 and after step A4) and we upper bound the displacement accounting for the Euclidean distance between the representations.

Recall that given two generic points $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$, we can express $\mathbf{X}^{(2)}$ as a function of $\mathbf{X}^{(1)}$ plus a *displacement* vector $\bar{\mathbf{X}}$:

$$\mathbf{X}^{(2)} = \mathbf{X}^{(1)} + \bar{\mathbf{X}}, \tag{A.1}$$

so that we can easily account for the Euclidean distance between $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ just as a function of the displacement vector $\bar{\mathbf{X}}$:

$$D_E\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] = \ell_2\left(\bar{\mathbf{X}}\right).$$

[11] Notice that we can safely make the assumption of strict positivity in sparse filtering since $\mathbf{u}$ is the output of the normalization along the feature which preserves the positivity.

(Before sparse filtering.) Let us now consider two points $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ which are almost collinear with an arbitrary small cosine distance $D_C\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] < \delta$. We can then express $\mathbf{X}^{(2)}$ as a point collinear with $\mathbf{X}^{(1)}$ to which a *bias* vector $\mathbf{B}$ is added:

$$\mathbf{X}^{(2)} = k\mathbf{X}^{(1)} + \mathbf{B},$$

where $k \in \mathbb{R}$ is a constant that preserves collinearity. With no loss of generality, we will assume $k > 1$; we exclude values of k smaller than zero which would generate a reflection (reflections are not relevant for the following treatment as they induce a cosine distance far greater than δ) and we ignore values of k between zero and one (in such a case, our proof will hold once we swap $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$). The bias vector $\mathbf{B}$ accounts for a relative displacement between the perfectly collinear sample $k\mathbf{X}^{(1)}$ and the almost collinear sample $k\mathbf{X}^{(1)} + \mathbf{B}$.

With reference to Eq. (A.1), the displacement vector $\bar{\mathbf{X}}$ is:

$$\bar{\mathbf{X}} = (k-1)\mathbf{X}^{(1)} + \mathbf{B}, \tag{A.2}$$

from which follows that:

$$D_E\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] = \ell_2\left(\bar{\mathbf{X}}\right) = \ell_2\left((k-1)\mathbf{X}^{(1)} + \mathbf{B}\right).$$

(Before sparse filtering — Upper bound) To upper bound $D_E\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right]$, we can evaluate the maximum value that $\ell_2\left(\bar{\mathbf{X}}\right)$ can reach, consistent with the constraint of a bounded cosine distance $D_C\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right]$. Formally, we can set up the optimization problem:

$$\max_{\bar{\mathbf{X}}\in\mathbb{R}^O} \ell_2\left(\bar{\mathbf{X}}\right)$$

under the constraint:

$$D_C\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] < \delta.$$

The maximization can be rewritten as:

$$\begin{aligned}\max_{\bar{\mathbf{X}}\in\mathbb{R}^O} \ell_2\left(\bar{\mathbf{X}}\right) &= \max_{\bar{\mathbf{X}}_j\in\mathbb{R}} \sqrt{\sum_{j=1}^{O} \bar{\mathbf{X}}_j^2} \\ &= \max_{\mathbf{B}_j\in\mathbb{R}} \sqrt{\sum_{j=1}^{O} \left((k-1)\mathbf{X}_j^{(1)} + \mathbf{B}_j\right)^2} \\ &= \max_{\mathbf{B}_j\in\mathbb{R}} \sqrt{\sum_{j=1}^{O} \left(\mathbf{B}_j\right)^2} \\ &= \max_{\mathbf{B}_j\in\mathbb{R}} \mathbf{B}_j,\end{aligned}$$

assuming: (i) that $\mathbf{X}^{(1)}$ and k are given, and (ii) that $\mathbf{X}_j^{(1)}$ and $\mathbf{B}_j$ are both positive (as this constitute the worst case that needs to be considered in the analysis of the upper bound). An upper bound on the displacement $\bar{\mathbf{X}}$ can be then computed from the solution to the individual constrained optimization problems for each component $\mathbf{B}_j$:

$$\max_{\mathbf{B}_j\in\mathbb{R}} \mathbf{B}_j$$

under the constraint:

$$\begin{aligned}\delta &> D_C\left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right] \\ &= D_C\left[\mathbf{X}^{(1)}, k\mathbf{X}^{(1)} + \mathbf{B}\right].\end{aligned}$$

By construction, we know that $D_C\left[\mathbf{X}^{(1)}, k\mathbf{X}^{(1)}\right] = 0$. Therefore the entire cosine distance must be accounted by the bias vector $\mathbf{B}$. Trigonometrically, from the cosine distance δ we can recover the angle opposite to a cathetus corresponding to the radius of an hypersphere centered on $k\mathbf{X}^{(1)}$ and bounding the module of $\mathbf{B}$. Let θ be the underlying angle between $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$:

$$\begin{aligned}\delta &= 1-\cos\theta\\ \theta &= \arccos(1-\delta).\end{aligned}$$

The radius of the hypersphere centered on $k\mathbf{X}^{(1)}$ inducing at most a cosine distance δ is:

$$\begin{aligned}\mathbf{B}_j &\le \mathbf{X}_j^{(1)}\sin\arccos(1-\delta)\\ &= \mathbf{X}_j^{(1)}\sqrt{1-(1-\delta)^2}\\ &= \mathbf{X}_j^{(1)}\sqrt{2\delta-\delta^2}.\end{aligned}$$

Substituting in Eq. (A.2), the displacement on each component can the be upper bounded as:

$$\begin{aligned}\bar{\mathbf{X}}_j &= (k-1)\mathbf{X}_j^{(1)}+\mathbf{B}_j\\ &\le (k-1)\mathbf{X}_j^{(1)}+\mathbf{X}_j^{(1)}\sqrt{2\delta-\delta^2}\\ &= \mathbf{X}_j^{(1)}\left(k-1+\sqrt{2\delta-\delta^2}\right).\end{aligned}$$

This upper bound depends on the original cosine distance δ, but, more significantly on the module of $\mathbf{X}^{(1)}$ and the stretching constant k. Indeed, the Euclidean distance along each component is given by the stretch ($\mathbf{X}_j^{(1)}(k-1)$) plus a small distance due to the angle ($\mathbf{X}_j^{(1)}\sqrt{2\delta-\delta^2}$).

(Steps A1 and A2) Let us now apply the linear projection and the absolute-value function defined in transformation A1 and A2:

$$\begin{aligned}\mathbf{F}^{(1)} &= f_{A1:A2}\left(\mathbf{X}^{(1)}\right) = \left|\mathbf{W}\mathbf{X}^{(1)}\right|\\ \mathbf{F}^{(2)} &= f_{A1:A2}\left(\mathbf{X}^{(2)}\right) = \left|\mathbf{W}\left(k\mathbf{X}^{(1)}+\mathbf{B}\right)\right| = k\mathbf{F}^{(1)} \pm |\mathbf{W}\mathbf{B}|.\end{aligned}$$

Component-wise we have:

$$\begin{aligned}\mathbf{F}_l^{(1)} &= \left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|\\ \mathbf{F}_l^{(2)} &= k\mathbf{F}_l^{(1)}+|\mathbf{W}\mathbf{B}|_l = k\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right| \pm \left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{B}_j\right|.\end{aligned}$$

The new displacement and the new Euclidean distance are:

$$\bar{\mathbf{F}}_l = (k-1)\,\mathbf{F}_l^{(1)} \pm |\mathbf{W}\mathbf{B}|_l, \tag{A.3}$$

$$D_E\left[\mathbf{F}^{(1)},\mathbf{F}^{(2)}\right] = \ell_2\left(\bar{\mathbf{F}}\right) = \sqrt{\sum_{l=1}^{L}\left((k-1)\,\mathbf{F}_l^{(1)} \pm |\mathbf{W}\mathbf{B}|_l\right)^2}.$$

(Steps A1 and A2 — Upper bound) The upper bound of each component of the new bias vector follows immediately:

$$\begin{aligned}|\mathbf{W}\mathbf{B}|_l &= \left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{B}_j\right|\\ &\le \left(\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\sqrt{2\delta-\delta^2}\right|\right)\\ &= \left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|,\end{aligned}$$

and then the upper bound on each component of the displacement in Eq. (A.3) is:

$$\begin{aligned}\bar{\mathbf{F}}_l &\le (k-1)\,\mathbf{F}_l^{(1)} + \left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|\\ &= (k-1)\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right| + \left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|\\ &= \left(k-1+\left|\sqrt{2\delta-\delta^2}\right|\right)\left|\sum_{j=1}^{O}\mathbf{W}^{(l)}\mathbf{X}_j^{(1)}\right|.\end{aligned}$$

(Step A3) Let us now apply the normalization along the rows defined in transformation A3:

$$\begin{aligned}\tilde{\mathbf{F}}_l^{(1)} &= f_{A3}\left(\mathbf{F}_l^{(1)}\right) = \frac{\mathbf{F}_l^{(1)}}{\sqrt{\sum_i^N\left(\mathbf{F}_l^{(i)}\right)^2}}\\ \tilde{\mathbf{F}}_l^{(2)} &= f_{A3}\left(\mathbf{F}_l^{(2)}\right) = \frac{k\mathbf{F}_l^{(1)}+|\mathbf{W}\mathbf{B}|_l}{\sqrt{\sum_i^N\left(\mathbf{F}_l^{(i)}\right)^2}} = k\tilde{\mathbf{F}}_l^{(1)} + \frac{|\mathbf{W}\mathbf{B}|_l}{\sqrt{\sum_i^N\left(\mathbf{F}_l^{(i)}\right)^2}}.\end{aligned}$$

Notice that the denominator is given by a feature-dependent sum across N samples; for simplicity, we will take this value to be a constant $\{c_l\}_{l=1}^{L}$, $c\in\mathbb{R}$:

$$\begin{aligned}\tilde{\mathbf{F}}_l^{(1)} &= \frac{\mathbf{F}_l^{(1)}}{c_l}\\ \tilde{\mathbf{F}}_l^{(2)} &= k\tilde{\mathbf{F}}_l^{(1)} + \frac{|\mathbf{W}\mathbf{B}|_l}{c_l}.\end{aligned}$$

The new displacement and the new Euclidean distance are:

$$\bar{\tilde{\mathbf{F}}}_l = (k-1)\,\tilde{\mathbf{F}}_l^{(1)} + \frac{|\mathbf{W}\mathbf{B}|_l}{c_l}, \tag{A.4}$$

$$D_E\left[\tilde{\mathbf{F}}^{(1)},\tilde{\mathbf{F}}^{(2)}\right] = \ell_2\left(\bar{\tilde{\mathbf{F}}}\right) = \sqrt{\sum_{l=1}^{L}\left((k-1)\,\tilde{\mathbf{F}}_l^{(1)} + \frac{|\mathbf{W}\mathbf{B}|_l}{c_l}\right)^2}.$$

(Step A3 — Upper bound) The upper bound of each component of the new bias vector follows immediately:

$$\frac{|\mathbf{W}\mathbf{B}|_l}{c_l} \le \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}$$

and then the upper bound on each component of the displacement in Eq. (A.4):

$$\begin{aligned}\bar{\tilde{\mathbf{F}}}_l &\le (k-1)\,\tilde{\mathbf{F}}_l^{(1)} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}\\ &= (k-1)\,\frac{\mathbf{F}_l^{(1)}}{c_l} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}\\ &= (k-1)\,\frac{\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}\\ &= \frac{k-1+\left|\sqrt{2\delta-\delta^2}\right|}{c_l}\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|\\ &= \frac{1}{c_l}\bar{\mathbf{F}}_l.\end{aligned}$$

Not surprisingly, after transformation A3, the Euclidean distance $D_E\left[\tilde{\mathbf{F}}^{(1)}, \tilde{\mathbf{F}}^{(2)}\right]$ is just rescaled since each component of the displacement $\bar{\mathbf{F}}_l$ is reduced by a factor $\frac{1}{c_l} = \frac{1}{\sqrt{\sum_i^N \left(\mathbf{F}_l^{(i)}\right)^2}}$.

(Step A4) Finally, let us apply the normalization along the samples defined in transformation A4:

$$\mathbf{Z}_l^{(1)} = f_{A4}\left(\tilde{\mathbf{F}}_l^{(1)}\right) = \frac{\tilde{\mathbf{F}}_l^{(1)}}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} = \frac{\frac{\mathbf{F}_l^{(1)}}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}$$

$$\mathbf{Z}_l^{(2)} = f_{A4}\left(\tilde{\mathbf{F}}_l^{(2)}\right) = \frac{\tilde{\mathbf{F}}_l^{(2)}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} = \frac{k\tilde{\mathbf{F}}_l^{(1)} + \frac{|\mathbf{WB}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}$$

$$= \frac{k\frac{\mathbf{F}_l^{(1)}}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} + \frac{\frac{|\mathbf{WB}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}.$$

Let us now consider the first term of $\mathbf{Z}_l^{(2)}$ and let us multiply it by $\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}$:

$$\mathbf{Z}_l^{(2)} = \frac{k\frac{\mathbf{F}_l^{(1)}}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} + \frac{\frac{|\mathbf{WB}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}$$

$$= k\mathbf{Z}_l^{(1)}\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} + \frac{\frac{|\mathbf{WB}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}.$$

The new displacement and the new Euclidean distance are:

$$\bar{\mathbf{Z}}_l = \left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - 1\right)\mathbf{Z}_l^{(1)} + \frac{|\mathbf{WB}|_l}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}, \tag{A.5}$$

$$D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] = \ell_2\left(\bar{\mathbf{Z}}\right)$$

$$= \sqrt{\sum_{l=1}^{L}\left(\left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - 1\right)\mathbf{Z}^{(1)} + \frac{|\mathbf{WB}|_l}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}\right)^2}. \tag{A.6}$$

For consistency, notice that if $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ were to be collinear, then $\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right) = k\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)$, and, by construction, $\mathbf{B} = 0$; therefore, in case of collinearity, $D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right]$ computed in Eq. (A.6) would be zero, thus agreeing with Theorem 2.

(Step A4 — Upper bound) Now, the upper bound of each component of the bias vector can be immediately upper bounded:

$$\frac{|\mathbf{WB}|_l}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} \leq \frac{\left|\sqrt{2\delta - \delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)},$$

and then the upper bound on each component of the displacement:

$$\bar{\mathbf{Z}}_l \leq \left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - 1\right)\mathbf{Z}_l^{(1)} + \frac{\left|\sqrt{2\delta - \delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}$$

$$= \left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - 1\right)\frac{\mathbf{F}_l^{(1)}}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} + \frac{\left|\sqrt{2\delta - \delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}$$

$$= \left(k\frac{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - 1\right)\frac{\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)} + \frac{\left|\sqrt{2\delta - \delta^2}\right|\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)}$$

$$= \frac{\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l}\left[\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right].$$

Notice that $\frac{\left|\sum_{j=1}^{O}\mathbf{W}_l^{(j)}\mathbf{X}_j^{(1)}\right|}{c_l} < 1$ since $c_l = \sqrt{\sum_i^N\left(\mathbf{F}_l^{(i)}\right)^2}$. Thus:

$$\bar{\mathbf{Z}}_l \leq \left[\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right].$$

The overall Euclidean distance between the representations $\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$ can then be bounded by:

$$D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] = \sqrt{\sum_{l=1}^{L}\left(\bar{\mathbf{Z}}_l\right)^2}$$

$$\leq L \cdot \left(\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right).$$

Thus $\epsilon = L \cdot \left(\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{F}}^{(2)}\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{F}}^{(1)}\right)}\right)$.

(Limit case) Lastly, let us consider the behavior of the Euclidean distance $D_E\left[\mathbf{z}_1, \mathbf{z}_2\right]$ as the cosine distance $D_C\left[\mathbf{x}_1, \mathbf{x}_2\right]$ tends to zero:

$$\lim_{\delta\to 0} D_E\left[\mathbf{z}_1, \mathbf{z}_2\right] = \lim_{\delta\to 0}\epsilon$$

$$= \lim_{\delta\to 0} L \cdot \left(\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right)$$

$$= \lim_{\delta\to 0} L \cdot \left(\frac{k}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right).$$

Let us now substitute $\ell_2\left(\tilde{\mathbf{f}}_2\right)$ with its definition. As the cosine distance δ tends to zero, $\mathbf{x}_1$ and $\mathbf{x}_2$ tend to be collinear. Therefore, $\ell_2\left(\tilde{\mathbf{f}}_2\right)$ tends to $k\ell_2\left(\tilde{\mathbf{f}}_1\right)$. We can then rewrite:

$$\lim_{\delta\to 0} D_E\left[\mathbf{z}_1, \mathbf{z}_2\right] = \lim_{\delta\to 0} L \cdot \left(\frac{k}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right)$$

$$= \lim_{\delta\to 0} L \cdot \left(\frac{k}{k \cdot \ell_2\left(\tilde{\mathbf{f}}_1\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right)$$

$$= 0.$$

Thus, in the limit, it holds that $\lim_{\delta\to 0}\epsilon = 0$. ■

$$f_{A1}\left(\mathbf{X}^{(1)}\right)=\mathbf{I}\mathbf{X}^{(1)}=\mathbf{X}^{(1)} \qquad f_{A1}\left(\mathbf{X}^{(2)}\right)=\mathbf{I}\mathbf{X}^{(2)}=\mathbf{X}^{(2)}$$
$$f_{A2*}\left(\mathbf{X}^{(1)}\right)=\sigma\left(\mathbf{X}^{(1)}\right)=\mathbf{\Sigma}^{(1)} \qquad f_{A2*}\left(\mathbf{X}^{(2)}\right)=\sigma\left(\mathbf{X}^{(2)}\right)=\mathbf{\Sigma}^{(2)}$$
$$f_{A3}\left(\mathbf{\Sigma}^{(1)}\right)=\left[\frac{\mathbf{\Sigma}_j^{(1)}}{\sqrt{\sum_{i=1}^N \mathbf{\Sigma}_j^{(i)}}}\right] \qquad f_{A3}\left(\mathbf{\Sigma}^{(2)}\right)=\left[\frac{\mathbf{\Sigma}_j^{(2)}}{\sqrt{\sum_{i=1}^N \mathbf{\Sigma}_j^{(i)}}}\right]$$
$$f_{A4}\left(\left[\frac{\mathbf{\Sigma}_j^{(1)}}{\sqrt{\sum_{j=1}^L \mathbf{\Sigma}_j^{(i)}}}\right]\right)=\left[\frac{1}{\sqrt{L}}\right]=\mathbf{Z}^{(1)} \quad f_{A4}\left(\left[\frac{\mathbf{\Sigma}_j^{(2)}}{\sqrt{\sum_{j=1}^L \mathbf{\Sigma}_j^{(i)}}}\right]\right)=\left[\frac{1}{\sqrt{L}}\right]=\mathbf{Z}^{(1)}.$$

Box III.

A.7. Proposition 2

Proposition 2. *Let* $\mathrm{Z}=\left\{\mathbf{Z}^{(i)}\right\}_{i=1}^N$ *be a set of vectors such that* $\mathbf{Z}^{(i)}\in\mathbb{R}^L$ *and* $\sum_{j=1}^L\left(\mathbf{Z}_j^{(i)}\right)^2=1$. *Then an optimal set of vectors that solve the optimization problem* $\min_{\mathbf{Z}\in\mathbb{R}^{L\times N}}\sum_{i=1}^N\sum_{j=1}^L\mathbf{Z}_j^{(i)}$ *is given by a multi-set of the orthonormal basis vectors of* $\mathbb{R}^L$.

Proof. We will prove this proposition geometrically.

Let us consider the optimization problem:

$$\min_{\mathbf{Z}\in\mathbb{R}^L}\sum_{j=1}^L \mathbf{Z}_j^{(1)},$$

subject the constraint:

$$\sum_{j=1}^L\left(\mathbf{Z}_j^{(1)}\right)^2=1.$$

The constraint defines the set of points describing a unitary hyper-sphere in $\mathbb{R}^L$, while the minimization problem defines diamond-shaped level sets (Bishop, 2007). The minimal level set intersecting the unitary hyper-sphere is the diamond inscribed in the unit sphere. The intersection points constitute the solution of our minimization problem. These points are the intersection points between the unit hyper-sphere and the axes of $\mathbb{R}^L$, having a single component set to one, while all the others are set to zero. By definition, these 1-sparse vectors are the orthonormal basis vectors $\{\mathbf{e}_i\}_{i=1}^L$. ■

A.8. Proposition 3

Proposition 3. *Let us consider the sparse filtering algorithm implemented using a sigmoid non-linearity* $\sigma(x)=\frac{1}{1+e^{-x}}$. *Let* $\left\{\mathbf{X}^{(i)}\in\mathbb{R}^O\right\}_{i=1}^N$ *be a set of points in the original space* $\mathbb{R}^O$. *Then, the transformations A1, A2*, A3 and A4, where A2* is the sigmoid non-linearity, do not preserve the structure of the data described neither by the Euclidean metric nor by the cosing metric.*

Proof. We divide this proposition in two parts and we prove each one by counterexample.

Let us focus first on the non-preservation of the Euclidean metric. Let us consider the case in which $\mathbf{X}^{(1)}$ is a vector such that $\mathbf{X}_j^{(1)}=1, \forall j, 1\le j\le O$, $\mathbf{X}^{(2)}$ is another vector such that $\mathbf{X}^{(2)}=2, \forall j, 1\le j\le O$, $L=O$, and $\mathbf{W}=\mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$D_E\left(\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right)=\sqrt{\sum_{j=1}^O(1-2)^2}=\sqrt{L}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Then we get the equations in Box III.

Thus, $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)=\mathbf{Z}^{(1)}$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)=\mathbf{Z}^{(1)}$. Now, the Euclidean distance between the vectors $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)$ is:

$$D_E\left(\mathbf{Z}^{(1)},\mathbf{Z}^{(1)}\right)=0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. This proves the first part of the proposition.

Let us focus now on the non-preservation of the cosine metric. Let us consider the case in which $\mathbf{X}^{(1)}$ is a vector such that $\mathbf{X}_j^{(1)}=2^j$, $\forall j, 1\le j\le O$, $\mathbf{X}^{(2)}$ is another vector such that $\mathbf{X}^{(2)}=-\mathbf{X}^{(1)}$, $L=O=2$, and $\mathbf{W}=\mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The cosine distance between the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$D_C\left(\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right)=1-\left|\frac{\sum_{j=1}^O\mathbf{X}_j^{(1)}\mathbf{X}_j^{(2)}}{\sqrt{\sum_{j=1}^O\left(\mathbf{X}_j^{(1)}\right)^2}\sqrt{\sum_{j=1}^O\left(\mathbf{X}_j^{(2)}\right)^2}}\right|=0.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$:

$$f_{A1}\left(\mathbf{X}^{(1)}\right)=\mathbf{I}\mathbf{X}^{(1)}=\mathbf{X}^{(1)} \qquad f_{A1}\left(\mathbf{X}^{(2)}\right)=\mathbf{I}\mathbf{X}^{(2)}=\mathbf{X}^{(2)}$$
$$f_{A2*}\left(\mathbf{X}^{(1)}\right)=\sigma\left(\mathbf{X}^{(1)}\right)=\mathbf{\Sigma}^{(1)} \qquad f_{A2*}\left(\mathbf{X}^{(2)}\right)=\sigma\left(\mathbf{X}^{(2)}\right)=\mathbf{\Sigma}^{(2)}$$
$$f_{A3}\left(\mathbf{\Sigma}^{(1)}\right)=\left[\frac{\mathbf{\Sigma}_j^{(1)}}{\sqrt{\sum_{i=1}^N \mathbf{\Sigma}_j^{(i)}}}\right] \qquad f_{A3}\left(\mathbf{\Sigma}^{(2)}\right)=\left[\frac{\mathbf{\Sigma}_j^{(2)}}{\sqrt{\sum_{i=1}^N \mathbf{\Sigma}_j^{(i)}}}\right]$$
$$f_{A4}\left(\left[\frac{\mathbf{\Sigma}_j^{(1)}}{\sqrt{\sum_{j=1}^L \mathbf{\Sigma}_j^{(i)}}}\right]\right)=\mathbf{Z}^{(1)} \quad f_{A4}\left(\left[\frac{\mathbf{\Sigma}_j^{(2)}}{\sqrt{\sum_{j=1}^L \mathbf{\Sigma}_j^{(i)}}}\right]\right)=\mathbf{Z}^{(2)}.$$

Thus, $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)=\mathbf{Z}^{(1)}$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)=\mathbf{Z}^{(2)}$, where $\mathbf{Z}^{(1)}=\left[0.99\ 0.14\right]$and $\mathbf{Z}^{(2)}=\left[0.70\ 0.71\right]$. Now, the cosine distance between the vectors $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)$ is:

$$D_C\left(\mathbf{Z}^{(1)},\mathbf{Z}^{(2)}\right)\neq 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the cosine metric. This proves the second part of the proposition. ■

A.9. Proposition 4

Proposition 4. *Let us consider the sparse filtering algorithm implemented using a soft ReLU non-linearity* $ReLU(x)=\max(\epsilon,x)$, *where* ϵ *is a small negligible value (for instance,* $\epsilon=10^{-8}$*). Let* $\left\{\mathbf{X}^{(i)}\in\mathbb{R}^O\right\}_{i=1}^N$ *be a set of points in the original space* $\mathbb{R}^O$. *Then, the*

$$f_{A1}\left(\mathbf{X}^{(1)}\right)=\mathbf{I}\mathbf{X}^{(1)}=\mathbf{X}^{(1)} \qquad f_{A1}\left(\mathbf{X}^{(2)}\right)=\mathbf{I}\mathbf{X}^{(2)}=\mathbf{X}^{(2)}$$
$$f_{A2^*}\left(\mathbf{X}^{(1)}\right)=ReLU\left(\mathbf{X}^{(1)}\right)=[\epsilon] \qquad f_{A2^*}\left(\mathbf{X}^{(2)}\right)=ReLU\left(\mathbf{X}^{(2)}\right)=[\epsilon]$$
$$f_{A3}\left([\epsilon]\right)=\left[\frac{\epsilon}{\sqrt{N\epsilon}}\right]=\left[\frac{1}{\sqrt{N}}\right] \qquad f_{A3}\left([\epsilon]\right)=\left[\frac{\epsilon}{\sqrt{N\epsilon}}\right]=\left[\frac{1}{\sqrt{N}}\right]$$
$$f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right)=\left[\frac{\sqrt{N}}{\sqrt{N}\sqrt{L}}\right]=\left[\frac{1}{\sqrt{L}}\right]=\mathbf{Z}^{(1)} \quad f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right)=\left[\frac{\sqrt{N}}{\sqrt{N}\sqrt{L}}\right]=\left[\frac{1}{\sqrt{L}}\right]=\mathbf{Z}^{(1)}.$$

Box IV.

$$f_{A1}\left(\mathbf{X}^{(1)}\right)=\mathbf{I}\mathbf{X}^{(1)}=\mathbf{X}^{(1)} \qquad f_{A1}\left(\mathbf{X}^{(2)}\right)=\mathbf{I}\mathbf{X}^{(2)}=\mathbf{X}^{(2)}$$
$$f_{A2^*}\left(\mathbf{X}^{(1)}\right)=ReLU\left(\mathbf{X}^{(1)}\right)=\mathbf{X}^{(1)} \qquad f_{A2^*}\left(\mathbf{X}^{(2)}\right)=ReLU\left(\mathbf{X}^{(2)}\right)=[\epsilon]$$
$$f_{A3}\left(\mathbf{X}^{(1)}\right)=\left[\frac{\mathbf{X}_j^{(1)}2j}{\sqrt{1+2^{2j}\epsilon^2}}\right]=\left[\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}\right] \qquad f_{A3}\left([\epsilon]\right)=\left[\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}\right]$$
$$f_{A4}\left(\left[\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}\right]\right)=\left[\frac{\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}}{\sqrt{\sum_{j=1}^{L}\frac{1}{1+2^{2j}\epsilon^2}}}\right]=\mathbf{Z}^{(1)} \quad f_{A4}\left(\left[\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}\right]\right)=\left[\frac{\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}}{\sqrt{\sum_{j=1}^{L}\frac{2^{2j}\epsilon^2}{1+2^{2j}\epsilon^2}}}\right]=\mathbf{Z}^{(2)}.$$

Box V.

transformations A1, A2, A3 and A4, where A2* is the ReLU non-linearity, do not preserve the structure of the data described neither by the Euclidean metric nor by the cosing metric.*

Proof. We divide this proposition in two parts and we prove each one by counterexample.

Let us focus first on the non-preservation of the Euclidean metric. Let us consider the case in which $\mathbf{X}^{(1)}$ is a vector such that $\mathbf{X}_j^{(1)}=-\frac{3}{\sqrt{2}}$, $\forall j$, $1\leq j\leq O$, $\mathbf{X}^{(2)}$ is another vector such that $\mathbf{X}^{(2)}=-\frac{1}{\sqrt{2}}$, $\forall j$, $1\leq j\leq O$, $L=O$, and $\mathbf{W}=\mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$D_E\left(\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right)=\sqrt{\sum_{j=1}^{O}\left(-\frac{3}{\sqrt{2}}+\frac{1}{\sqrt{2}}\right)^2}=\sqrt{2L}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Then we get the equations in Box IV.

Thus, $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)=\mathbf{Z}^{(1)}$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)=\mathbf{Z}^{(1)}$. Now, the Euclidean distance between the vectors $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)$ is:

$$D_E\left(\mathbf{Z}^{(1)},\mathbf{Z}^{(1)}\right)=0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. This proves the first part of the proposition.

Let us focus now on the non-preservation of the cosine metric. Let us consider the case in which $\mathbf{X}^{(1)}$ is a vector such that $\mathbf{X}_j^{(1)}=\frac{1}{2j}$, $\forall j$, $1\leq j\leq O$, $\mathbf{X}^{(2)}$ is another vector such that $\mathbf{X}^{(2)}=-\mathbf{X}^{(1)}$, $L=O=2$, and $\mathbf{W}=\mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The cosine distance between the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ is:

$$D_C\left(\mathbf{X}^{(1)},\mathbf{X}^{(2)}\right)=1-\left|\frac{\sum_{j=1}^{O}\mathbf{X}_j^{(1)}\mathbf{X}_j^{(2)}}{\sqrt{\sum_{j=1}^{O}\left(\mathbf{X}_j^{(1)}\right)^2}\sqrt{\sum_{j=1}^{O}\left(\mathbf{X}_j^{(2)}\right)^2}}\right|=0.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. Then we get the equations in Box V.

Thus, $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)=\mathbf{Z}^{(1)}$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)=\mathbf{Z}^{(2)}$, where $\mathbf{Z}^{(1)}=\left[\frac{\sqrt{1+16\epsilon^2}}{\sqrt{2+20\epsilon^2}}\ \frac{\sqrt{1+4\epsilon^2}}{\sqrt{2+20\epsilon^2}}\right]$ and $\mathbf{Z}^{(2)}=\left[\frac{\sqrt{1+16\epsilon^2}}{\sqrt{5+32\epsilon^2}}\ \frac{2\sqrt{1+4\epsilon^2}}{\sqrt{5+32\epsilon^2}}\right]$. Now, the cosine distance between the vectors $f_{A1:A4}\left(\mathbf{X}^{(1)}\right)$ and $f_{A1:A4}\left(\mathbf{X}^{(2)}\right)$ is:

$$D_C\left(\mathbf{Z}^{(1)},\mathbf{Z}^{(2)}\right)\neq 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the cosine metric. This proves the second part of the proposition. ■

A.10. Theorem 5

Theorem 5. *Let $\mathbf{X}^{(1)}\in\mathbb{R}^O$ be a point in the original space $\mathbb{R}^O$ and let $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}$ be a representation filter centered on $\mathbf{X}^{(1)}$, that is, $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}\left(\mathbf{X}^{(1)}\right)=0$. Let us now consider a point $\mathbf{X}^{(2)}\in\mathbb{R}^O$ within the same representation cone, that is, a point such that $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}\left(\mathbf{X}^{(2)}\right)\leq\epsilon$ for an arbitrarily small $\epsilon\in\mathbb{R}$, $\epsilon>0$.*

Let us assume that: (i) points $\mathbf{X}^{(i)}\in\mathbb{R}^O$ distribute in a limited region of space bounded by a hyper-sphere of radius M; and, (ii) points $\mathbf{X}^{(i)}\in\mathbb{R}^O$ distribute uniformly in this limited region of space.

Then, given that $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}\left(\mathbf{X}^{(2)}\right)\leq\epsilon$, it follows:

$$\frac{O\delta}{\left(\frac{M}{m}\right)^{O-1}}\cdot\frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)}\leq P\left(D_E\left[\mathbf{Z}^{(1)},\mathbf{Z}^{(2)}\right]\leq\delta\right)\leq\frac{O\delta}{m}\cdot\frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)},$$

where $\delta\in\mathbb{R}$, $\delta>0$ defines the neighborhood of $\mathbf{X}^{(1)}$, m is the distance of $\mathbf{X}^{(1)}$ from the origin, and $\Gamma(\cdot)$ is the gamma function.

Proof. Let us consider $\mathbf{X}^{(1)}\in\mathbb{R}^O$ and let us consider its neighborhood as the set of points $\mathbf{X}^{(i)}$ within a hyper-sphere of radius δ, that is, $D_E\left[\mathbf{X}^{(1)},\mathbf{X}^{(i)}\right]\leq\delta$.

Let us consider now the representation filter $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}$ and let m be the distance of $\mathbf{X}^{(1)}$ from the origin. We first define the minimal representation filter $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}$ as the hyper-cone of height m and radius δ inscribing the neighborhood of $\mathbf{X}^{(1)}$. We also define a maximal representation filter $R_{\mathbf{X}^{(1)}}^{\mathbf{e}_k}$ as the hyper-cone of height M and, by

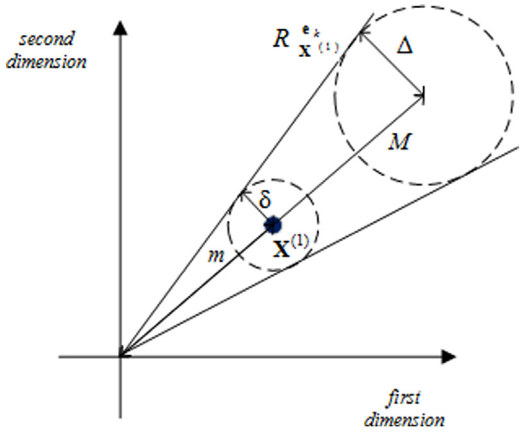

Fig. A.9. Schema of the data point $\mathbf{X}^{(1)}$, the neighborhood of $\mathbf{X}^{(1)}$, and the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}(1)}$ in two-dimensional space.

trigonometry, radius $\Delta = M \cdot \frac{\delta}{m}$. For illustration, refer to the schema in Fig. A.9, where we represented this setup in the case $O = 2$.

Let us now consider the point $\mathbf{X}^{(2)}$ sampled within the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}(1)}$. Since the sampling probability is uniform within the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}(1)}$, we can evaluate the probability of $\mathbf{X}^{(2)}$ to fall in the neighborhood of $\mathbf{X}^{(1)}$ as the volume of the neighborhood of $\mathbf{X}^{(1)}$ normalized by the total volume of the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}(1)}$.

Let us consider the neighborhood of $\mathbf{X}^{(1)}$. Its volume can be computed as:

$$V_{sphere}(O, \delta) = \mathcal{V}_O \delta^O,$$

where $\mathcal{V}_O$ is the following function:

$$\mathcal{V}_n = \frac{\pi^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}+1\right)}.$$

Let us now consider the representation filter $R^{\mathbf{e}_k}_{\mathbf{X}(1)}$. We bound this volume considering the minimal and maximal hyper-cone described above. The volume of the hyper-cone depends on the volume of the lower-dimensional hyper-sphere in the base (Ball, 1997) and it can be computed as:

$$\begin{aligned}\frac{1}{O} \cdot m \cdot V_{sphere}(O-1, \delta) &\le V_{cone}(O, \delta, l)\\ &\le \frac{1}{O} \cdot M \cdot V_{sphere}(O-1, \Delta)\\ \frac{1}{O} \cdot m \cdot \mathcal{V}_{O-1} \cdot \delta^{O-1} &\le V_{cone}(O, \delta, l)\\ &\le \frac{1}{O} \cdot M \cdot \mathcal{V}_{O-1} \cdot \left(M \cdot \frac{\delta}{m}\right)^{O-1}.\end{aligned}$$

Let us now consider the ratio of the volume of the hyper-sphere and the volume of the hyper-cone:

$$\frac{\mathcal{V}_O \delta^O}{\frac{1}{O} \cdot M \cdot \mathcal{V}_{O-1} \cdot \left(M \cdot \frac{\delta}{m}\right)^{O-1}} \le \frac{V_{sphere}(O, \delta)}{V_{cone}(O, \delta, l)} \le \frac{\mathcal{V}_O \delta^O}{\frac{1}{O} \cdot m \cdot \mathcal{V}_{O-1} \cdot \delta^{O-1}}$$

$$\frac{O\delta m^{O-1}}{M^O} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)} \le \frac{V_{sphere}(O, \delta)}{V_{cone}(O, \delta, l)} \le \frac{O\delta}{m} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)}.$$

Thus, since we assumed that $R^{\mathbf{e}_k}_{\mathbf{X}(1)}\left(\mathbf{X}^{(2)}\right) \le \epsilon$, it follows that $\frac{O\delta}{\left(\frac{M}{m}\right)^{O-1}} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)} \le P\left(D_E\left[\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}\right] \le \delta\right) \le \frac{O\delta}{m} \cdot \frac{\Gamma\left(\frac{O+1}{2}\right)}{\Gamma\left(\frac{O+2}{2}\right)}$. ■

References

Aharon, M., Elad, M., & Bruckstein, A. (2006). k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, *54*(11), 4311–4322.

Amari, S.-i. (2016). *Information geometry and its applications (Vol. 194)*. Springer.

Ball, K. (1997). An elementary introduction to modern convex geometry. *Flavors of Geometry*, *31*, 1–58.

Bell, A. J., & Sejnowski, T. J. (1997). The "independent components" of natural scenes are edge filters. *Vision Research*, *37*(23), 3327–3338.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: a review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Bishop, C. M. (2007). *Pattern recognition and machine learning*. Springer.

Bruce, N. D., Rahman, S., & Carrier, D. (2016). Sparse coding in early visual representation: from specific properties to general principles. *Neurocomputing*, *171*, 1085–1098.

Burkhardt, F., Paeschke, A., Rolfes, M., Sendlmeier, W. F., & Weiss, B. (2005). A database of German emotional speech. In *Interspeech (Vol. 5)* (pp. 1517–1520).

Carandini, M., & Heeger, D. J. (2012). Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, *13*(1), 51–62.

Chen, S. S., Donoho, D. L., & Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Review*, *43*(1), 129–159.

Childers, D. G., Skinner, D. P., & Kemerait, R. C. (1977). The cepstrum: a guide to processing. *Proceedings of the IEEE*, *65*(10), 1428–1443.

Coates, A., Ng, A. Y., & Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics* (pp. 215–223).

Denil, M., & de Freitas, N. (2012). Recklessly approximate sparse coding. ArXiv Preprint ArXiv:1208.0959.

Dong, Z., Pei, M., He, Y., Liu, T., Dong, Y., & Jia, Y. (2014). Vehicle type classification using unsupervised convolutional neural network. In *Pattern recognition, 2014 22nd international conference on* (pp. 172–177). IEEE.

Dong, Z., Wu, Y., Pei, M., & Jia, Y. (2015). Vehicle type classification using a semisupervised convolutional neural network. *Intelligent Transportation Systems, IEEE Transactions on*, *16*(4), 2247–2256.

Elad, M. (2010). *Sparse and redundant representation*. Springer.

Engan, K., Aase, S. O., & Husoy, J. H. (1999). Method of optimal directions for frame design. In *Acoustics, speech, and signal processing, 1999. Proceedings., 1999 IEEE international conference on (Vol. 5)* (pp. 2443–2446). IEEE.

Földiák, P., & Young, M. P. (1995). Sparse coding in the primate cortex. *The Handbook of Brain Theory and Neural Networks*, *1*, 1064–1068.

Ganguli, S., & Sompolinsky, H. (2012). Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual Review of Neuroscience*, *35*, 485–508.

Goh, H., Thome, N., Cord, M., & Lim, J.-H. (2012). Unsupervised and supervised visual codes with restricted Boltzmann machines. In *Computer vision–ECCV 2012* (pp. 298–311). Springer.

Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., et al. (2013). Challenges in representation learning: a report on three machine learning contests. In *Neural information processing* (pp. 117–124). Springer.

Han, Y., Lee, S., Nam, J., & Lee, K. (2016). Sparse feature learning for instrument identification: effects of sampling and pooling methods. *The Journal of the Acoustical Society of America*, *139*(5), 2290–2298.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554.

Hurley, N., & Rickard, S. (2009). Comparing measures of sparsity. *Information Theory, IEEE Transactions on*, *55*(10), 4723–4741.

Kozlov, A. S., & Gentner, T. Q. (2016). Central auditory neurons have composite receptive fields. *Proceedings of the National Academy of Sciences*, 201506903.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images*. Citeseer: University of Toronto.

Lederer, J., & Guadarrama, S. (2014). Compute less to get more: using ORC to improve sparse filtering. ArXiv Preprint ArXiv:1409.4689.

Lei, Y., Jia, F., Lin, J., Xing, S., & Ding, S. (2015). An intelligent fault diagnosis method using unsupervised feature learning towards mechanical big data. *Industrial Electronics, IEEE Transactions on*, 1.

Linsker, R. (1989). An application of the principle of maximum information preservation to linear systems. In *Advances in neural information processing systems* (pp. 186–194).

Liu, Z., He, Y., Xie, Y., Gu, H., Liu, C., & Pei, M. (2016). Pedestrian detection using deep channel features in monocular image sequences. In *International conference on neural information processing* (pp. 608–615). Springer.

MacKay, D. J. (2003). *Information theory, inference, and learning algorithms (Vol. 7)*. Cambridge University Press.

Mallat, S. G., & Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, *41*(12), 3397–3415.

McNamara, D., Ong, C. S., & Williamson, R. C. (2016). A modular theory of feature learning. ArXiv Preprint ArXiv:1611.03125.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning (Vol. 2011)* (p. 4).

Ngiam, J., Chen, Z., Bhaskar, S. A., Koh, P. W., & Ng, A. Y. (2011). Sparse filtering. In *Advances in neural information processing systems* (pp. 1125–1133).

Olshausen, B. A., Field, D. J. (1997). Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, *37*(23), 3311–3325.

Pastor, G., Mora-Jiménez, I., Jäntti, R., & Caamaño, A. J. (2015). Mathematics of sparsity and entropy: axioms, core functions and sparse recovery. ArXiv Preprint ArXiv:1501.05126.

Principe, J. C. (2010). *Information theoretic learning: Rényi's entropy and kernel perspectives*. Springer.

Raja, K. B., Raghavendra, R., Vemuri, V. K., & Busch, C. (2015). Smartphone based visible iris recognition using deep sparse filtering. *Pattern Recognition Letters*, *57*, 33–42.

Romaszko, L. (2013). A deep learning approach with an ensemble-based neural network classifier for black box ICML 2013 contest. In *Workshop on challenges in representation learning, ICML*.

Ryman, S. K., Bruce, N. D., & Freund, M. S. (2016). Temporal responses of chemically diverse sensor arrays for machine olfaction using artificial intelligence. *Sensors and Actuators B (Chemical)*.

Schuller, B., Batliner, A., Steidl, S., & Seppi, D. (2011). Recognising realistic emotions and affect in speech: state of the art and lessons learnt from the first challenge. *Speech Communication*, *53*(9), 1062–1087.

Simard, P. Y., LeCun, Y. A., Denker, J. S., & Victorri, B. (1998). Transformation invariance in pattern recognition - tangent distance and tangent propagation. In *Neural networks: Tricks of the trade* (pp. 239–274). Springer.

Sugiyama, M., & Kawanabe, M. (2012). *Machine learning in non-stationary environments: introduction to covariate shift adaptation*. MIT Press.

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on machine learning* (pp. 1096–1103). ACM.

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

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *Evolutionary Computation, IEEE Transactions on*, *1*(1), 67–82.

Wu, D., Parsons, T. D., & Narayanan, S. S. (2010). Acoustic feature analysis in speech emotion primitives estimation. In *Interspeech* (pp. 785–788).

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2003). Distance metric learning with application to clustering with side-information. *Advances in Neural Information Processing Systems*, 521–528.

Zhang, Z., Xu, Y., Yang, J., Li, X., & Zhang, D. (2015). A survey of sparse representation: algorithms and applications. *Access, IEEE*, *3*, 490–530.