Volume 7 | Issue 1 Article 12

2018

Exact Recovery of Prototypical Atoms through Dictionary Initialization

Greg Zanotti
DePaul University, gzanotti@mail.depaul.edu

Enrico Au-Yeung
DePaul University, eauyeun1@depaul.edu

Exact Recovery of Prototypical Atoms through Dictionary Initialization

Greg Zanotti[1]

Department of Mathematical Sciences

Enrico Au-Yeung, PhD; Faculty Advisor

Department of Mathematical Sciences

ABSTRACT In dictionary learning, a matrix comprised of signals Y is factorized into the product of two matrices: a matrix of prototypical "atoms" D, and a sparse matrix containing coefficients for atoms in D, called X. Dictionary learning finds applications in signal processing, image recognition, and a number of other fields. Many algorithms for solving the dictionary learning problem follow the alternating minimization paradigm; that is, by alternating solving for D and X. In 2014, Agarwal et al. proposed a dictionary initialization procedure that is used before this alternating minimization process. We show that there is a modification to this initialization algorithm and a corresponding data generating process under which full recovery of D is possible without a subsequent alternating minimization procedure. Our findings indicate that the costly step of alternating minimization can be bypassed, and that other data generating processes may enjoy the same features as the one we propose.

INTRODUCTION

Processing big data continues to be one of the most palpable challenges of the 21st century. As current computing technology reaches its physical limits, the application of advances from applied mathematics in solving this problem becomes a critical necessity. One approach that has evolved from both mathematics and computer science involves increasing the amount of apparent information in big data while simultaneously reducing its size. This technique is known as unsupervised learning or dimensionality reduction.

Dictionary learning [11] is an approach to unsupervised learning that characterizes a large collection of data (hereafter "signals") by sparse linear combinations of a small set of prototypical signals. We can think of these prototypical signals as the representatives for the larger data set. This small set of representatives is known as the dictionary. The term sparse is there to emphasize that each sample in the data can be expressed as a linear combination of a small number of elements in the dictionary. Let us begin with three motivating examples to illustrate why learning the dictionary from the data is important. These examples are given in order of

[1] gzanotti@mail.depaul.edu
Research Completed in Autumn 2016

sophistication, from the simple case in two dimension, where a scientist can visualize the data, to an application in high dimension that arises in computational neuroscience.

Example 1. Suppose we have a collection of 200 points in the plane. Each point in the plane is identified by its x-coordinate and its y-coordinate. After looking at the data, the scientist realizes that by rotating the horizontal axis of the plane, half the points will be aligned with the rotated axis, x'. By rotating the vertical axis, the other half of the points will be aligned with the second rotated axis, y'. Therefore, in this example (illustrated in Figure 1), the scientist will find it more natural to view the data in terms of the two rotated axes, rather than in terms of the original two axis. As a small step in the data analysis, she finds it helpful to think of the one hundred points that lie on the red axis as the red points, and the remaining hundred points that lie on the green axis as the green points. Visualizing the data points as either red or green are more meaningful than looking at the original coordinates of the points. Note that in this example, the ability to identify the red and green axis depends on the visualization of the data. The red and green axes are two representative vectors for this data set in the plane. In higher dimension, it is often difficult or impossible to visualize a cloud of data points. In that case, machine learning can be used to identify the representative vectors from the data, instead of from a visualization.

Example 2. A survey is conducted among 600 people. The participants of the survey evaluate their jobs on a scale from 1 to 10, where 10 means highly satisfactory and 1 indicates little job satisfaction. These ratings result in 600 numbers that are stored in a vector $\vec{y}$. The team of scientists who design the survey know from previous experience that job satisfaction can largely be explained by three factors. These three attributes are income (salary for the job), fulfillment (to what extent the worker feels she is being appreciated), and contribution (to what extent the worker feels he is making a contribution to society). Let us label these factors income, fulfillment, and contribution by the variables F_1, F_2, F_3, respectively.

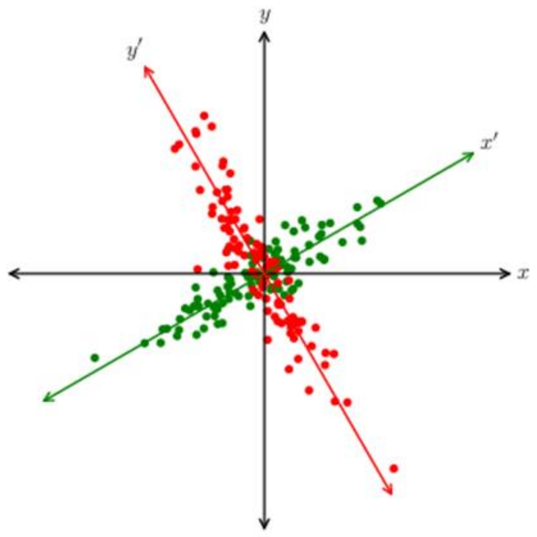

Figure 1. The axes x and y are rotated to x′ and y′ to better align with the data. These new axes act as representative vectors which describe each set of colored points.

There are three vectors $\vec{v}_1, \vec{v}_2, \vec{v}_3$ in $\mathbb{R}^{600}$ that store the corresponding values of F_1, F_2, F_3 for the 600 workers. In an ideal setting, where there is no noise to the data, and these three factors can completely explain job satisfaction (Y), we have the following ideal model:

$$Y = c_1F_1 + c_2F_2 + c_3F_3.$$

This equation expresses the relationship: the rating of a job by a worker, as an indication of job satisfaction, is determined by three explanatory variables: income, fulfillment, and contribution to society. To determine the coefficients c_1, c_2, and c_3, we can follow an approach from linear algebra: the vector $\vec{y} \in \mathbb{R}^{600}$ is projected into the subspace spanned by the vectors $\vec{v}_1, \vec{v}_2, \vec{v}_3$. To be clear, this subspace W consists of all the vectors in $\mathbb{R}^{600}$ that are linear combinations of $\vec{v}_1, \vec{v}_2, \vec{v}_3$. The task of finding the coefficients for the vector $\vec{y}$ is equivalent to the task of seeking the vector $\vec{w} \in W$ that is the best approximation to the given vector $\vec{y}$.

This approach to understand Y as a linear function of three explanatory variables F_1, F_2, F_3 relies on the previous experience of the team of scientists who design the survey. They have the prior knowledge of what are the three main determining factors of job satisfaction. The three factors are chosen before the collection of data. Since the choice of the factors is not informed by the data itself, this leads to a natural question: Is it possible that if we can learn from the data what factors are important, we might gain better insight?

Now, imagine that one purpose of the survey is to learn which jobs tend to yield high levels of job satisfaction. Instead of focusing on the job satisfaction of an individual, the team of scientists want to see what they can learn from the data. Some patterns emerge from the data. Among the participants of the survey, two hundred people are construction workers. Their data points for job satisfaction tend to lie on a plane that is spanned by two vectors $\vec{f}_1$ and $\vec{f}_2$. That is, their values for the vector $\vec{y}$ are linear combinations of $\vec{f}_1$ and $\vec{f}_2$. For those two hundred participants who work as firefighters and social workers, their data points tend to lie on a plane that is spanned by two vectors $\vec{f}_2$ and $\vec{f}_3$. For the remaining two hundred participants who work as engineers and nurses, their values of $\vec{y}$ can be expressed as linear combinations of $\vec{f}_1$ and $\vec{f}_3$. The original vector $\vec{y} \in \mathbb{R}^{600}$ can be split into three vectors $\vec{y}_1, \vec{y}_2, \vec{y}_3$ in $\mathbb{R}^{200}$. These findings can be expressed as a system of 3 equations,

$$\begin{aligned}\vec{y}_1 &= x_{11}\vec{f}_1 + x_{21}\vec{f}_2 + x_{31}\vec{f}_3\\ \vec{y}_2 &= x_{12}\vec{f}_1 + x_{22}\vec{f}_2 + x_{32}\vec{f}_3\\ \vec{y}_3 &= x_{13}\vec{f}_1 + x_{23}\vec{f}_2 + x_{33}\vec{f}_3\end{aligned}$$

and with the condition that

$$x_{31} = 0, x_{12} = 0, x_{23} = 0.$$

To summarize this finding, we can represent the situation as a matrix factorization, $Y = DX$. The matrix Y with 200 rows consists of 3 columns $\vec{y}_1, \vec{y}_2, \vec{y}_3$. The matrix D is the dictionary that consists of three columns $\vec{f}_1, \vec{f}_2, \vec{f}_3$. The first column of the matrix X contains the coefficients x_{11}, x_{21}, x_{31} for the vector $\vec{y}_1$. The matrix X with 3 rows and 3 columns is 2-sparse in each column. That means 2 entries in each column are not zero. It should be emphasized that given only the data matrix Y, we are asking a machine to learn both matrices D and X in order to express the relationship $Y = DX$.

The factors that are learned from the data are not necessarily identified as the original explanatory variables (income, fulfillment, contribution). However, from these factors, we discover an emerging pattern: job satisfaction among participants of the survey fall into three categories. Construction workers belong to one category, while firefighters and social workers belong to another. Engineers and nurses fall under a third category.

Example 3. Can a paralyzed man regain the motion of his hand? Ian Burkhart is a quadriplegic man who has become the first person to be implanted with technology that sends signals from the brain to muscles. This technological breakthrough is allowing him to regain some movement in his right arm and wrist. In 2014, scientists at Ohio State's Neurological Institute implanted a microchip into the 24-year-old quadriplegic's motor cortex. Its goal is to bypass his damaged spinal cord so that with the help of a signal decoder and electrode-packed sleeve, he can control his right arm with his thoughts.

Over a period of 15 months, researchers at the Ohio State University Wexner Medical Center and engineers from Battelle, the medical group that developed the decoder software and electrode sleeve, have helped Ian relearn fine motor skills with weekly training sessions. In a paper in *Nature* [3], the authors describe connecting a cable from the port screwed into Ian's skull (where the chip is) to a computer that translates the brain signals into instructions for the sleeve, which stimulates his muscles into moving his wrist and fingers. For example, when Ian thinks "clench fist," the implanted electrodes record the activity in his motor cortex. Those signals are decoded in real-time, jolting his arm muscles in the right places so that his fingers curl inwards.

Performing the task of signal processing with a massive amount of data presents a challenge. The human brain can generate gigabytes of brain signals in just under a minute and a half, at a sampling rate of 30,000 samples/second on 96 channels using the Neuroport neural data acquisition system [3]. The researchers need to decipher which brain signal is responsible for finger movement.

Dictionary learning. This last example in computational neuroscience illustrates the need of a powerful tool to characterize the collection of signals by sparse linear combinations of prototypical signals. Formally, consider n signals $y_1, y_2, \dots, y_n$, each in $\mathbb{R}^d$. Then the dictionary learning problem we consider is

$$\begin{aligned} \min_{D,X} \quad & \sum_{i=1}^{n} ||y_i - Dx_i||_2^2 \\ \text{s.t.} \quad & ||x_i||_0 \leq s, \qquad i = 1, \dots, n. \end{aligned}$$

where $Y \in \mathbb{R}^{d\times n}$ is a matrix of signals, $D \in \mathbb{R}^{d\times r}$ is a dictionary of r prototypical signals, and $X \in \mathbb{R}^{r\times n}$ is a sparse matrix of coefficients. Additionally, x_i means the ith column of the matrix X. The notation $||x_i||_0$ represents the zero "norm," which is the number of non-zero elements of x_i. The inequality on the zero "norm" above means that each column in X is s-sparse; that is, each column has at most s nonzero entries.

In the parlance of dictionary learning, we have r atoms $a_1, a_2, \dots, a_r$, which are vectors in $\mathbb{R}^d$. Each signal is approximately equal to a linear combination of s atoms. For example, for y_i, there exist atoms $a_{i_1}, a_{i_2}, \dots, a_{i_s}$ and coefficients $c_{i_j}, j = 1,2, \dots, s$, such that

$$y_i = c_{i_1}a_{i_1} + c_{i_2}a_{i_2} + \dots + c_{i_s}a_{i_s}.$$

Dictionary learning attempts to recover a *true dictionary* D and *sparse matrix* X which define the *signals* or *signal matrix* Y by the relationship $Y \approx DX$.

Dictionary learning allows a signal to be represented by a vector of sparse coefficients, thus massively reducing both the storage requirements and processing requirements while describing the signal in terms of atoms, which are high in information density. Dictionary learning has been applied to perform face recognition [12], image restoration and inpainting (even when the image is heavily corrupted [8] or data is limited or incomplete [9]), and modeling of data with hierarchical structure, such as images and text [7].

A number of algorithms attempt to solve the dictionary learning problem. Most algorithms can be described as alternating minimization. These algorithms begin by initializing the dictionary to a random matrix, and then alternating between solving for the dictionary D and the sparse matrix X. That means at each iteration, there are two steps. First, the matrix D is held fixed, while the best sparse matrix X is determined. Next, using the matrix X just computed, the dictionary D is updated. The method of optimal directions (MOD) [6] solves for the dictionary at each iteration, by the method of least squares, and computes the sparse matrix by a sparse coding algorithm such as Orthogonal Matching Pursuit (OMP) [5]. A more sophisticated approach is the K-SVD algorithm [2]. This widely used algorithm replaces MOD's least squares step by a more granular operation which decomposes error in the dictionary on a per-column basis.

The technique of alternating minimization involves computationally intensive operations on large matrices that can take hours or days to converge. A creative idea was introduced at the prestigious Conference on Learning Theory (COLT 2014) in Spain. Agarwal, et al. present a fast, scalable algorithm for initializing the dictionary D using a clustering procedure based on SVD to extract initial atoms [1]. This step recovers the dictionary with bounded error, and is followed by an alternating minimization procedure that iterates between LASSO [10] and least squares steps. The authors state that this is the first known exact recovery algorithm for the overcomplete $(r > d)$ dictionary case. Importantly, they also empirically verify that under a common data generating process for Y, the initialization step is not sufficient for obtaining a good approximation of the true dictionary D.

The main contribution of our work is to provide empirical evidence for a data generating process and conditions under which a modified initialization algorithm similar to that of [1] nearly recovers the atoms of the true dictionary D. This discovery is important because it can obviate the requirement of performing a subsequent alternating minimization step that ensures exact recovery. Removing this procedure can reduce the computational cost of dictionary learning significantly. We also provide in detail the calculation that provides some justification of why the algorithm works under certain assumptions on the data generating process. This calculation can be found in the Appendix to this article.

If an oracle can supply us with the dictionary D, so that the only unknown variable is the matrix X, then this can be formulated as an convex optimization problem. In that case, orthogonal matching pursuit (OMP) is an an efficient method to solve for the unknown matrix X that is sparse in each column [5]. However, in our problem, the challenge is that given the data matrix Y, both the dictionary D and the coefficients matrix X are unknown.

Standard approaches to convex optimization are well established [4]. The dictionary learning problem can be formulated as a non-convex optimization problem. There is one principal difference between non-convex optimization (NCO) method and the algorithm under consideration in this article. Note that while using the NCO method can find a dictionary D that nearly recovers the data Y, the optimization algorithm does not attempt to recover the true dictionary that generates the data. In contrast, we want an algorithm that can nearly recover all the atoms in the true dictionary.

INITIALIZATING DICTIONARIES FOR FAST OPTIMIZATION

The core insight of the initialization algorithm InitDictionaryLearn of Agarwal et al. is that the atoms extracted from the data should be limited to those that represent clusters of signals. The algorithm tests pairs of signals to see if they share an atom, then finds signals that are correlated with the pair, which forms a cluster of signals. If the cluster is "good" (a decision determined by Agarwal's UniqueIntersection algorithm), then InitDictionaryLearn extracts an atom in a process similar to PCA, using information from every entry of the signals in the cluster. We modify the algorithms InitDictionaryLearn and UniqueIntersection presented in [1], and name our modifications P1 and P2, respectively. The algorithm P1 is outlined in Algorithm 2, and P2 is outlined in Algorithm 1. Our modifications follow.

Algorithm 1 Verifying candidate atom cluster S contains a true atom. Note: p_0 and p_1 refer to the first and second entries of the tuple p.

```
1: procedure P2
2:     Input $S$, $r$, $s$, $\tau$.
3:     Initialize $P \leftarrow$ exclusive consecutive
4:       pairs in $S$, $c \leftarrow 0$, $M \leftarrow |P|$,
5:       $\epsilon_1 \leftarrow \frac{19s^3}{r}$
6:     for $p \in P$ do
7:         $i, j \leftarrow p_0, p_1$
8:         if $|\langle y_i, y_j \rangle| > \tau$ then
9:             $c \leftarrow c + 1$
10:        end if
11:    end for
12:    return $c/M \geq 1 - \epsilon_1$
13: end procedure
```

We evaluate the algorithm on the result of a data generating process wherein the original dictionary D is a square Discrete Cosine Transform matrix, and the elements of the columns of the true sparse matrix X are integers which have limitations on their magnitude and distribution. The following section provides a description of this data generating process.

We formulate a new correlation threshold τ_1 specifically for our data generating process based on the assumption that the columns of our sparse matrix take on certain values in the worst case. Our correlation threshold's calculation implies additional restrictions for the data generating process. A description is given in the Correlation Threshold section.

Algorithm 2 Clustering signals in A for extracting r true atoms, assuming each signal is represented by an s-sparse sum of the atoms. The EIGTOP function returns the top eigenvector of a matrix. "$|\langle\cdot\rangle|$" is the unsigned inner product. "$C_{:,i}$" is the ith column of C.

```
procedure P1
    Input signals $Y$, number of atoms $r$,
     sparsity $s$, correlation threshold $\tau_1$,
     minimum separation between recorered atoms in norm difference $\epsilon_A$,
     and coherence of true dictionary $\mu$.
    Initialize $A \leftarrow \emptyset$, $L \leftarrow \emptyset$,
     $\tau \leftarrow \tau_1$
    while $|A| \leq r$ do
        Randomly pick a pair of signals
          $(y_p, y_q)$ from $2^Y \setminus L$.
        $L \leftarrow L \cup \{(p,q)\}$.
        if $|\langle y_p, y_q \rangle| > \tau$ then
            $S \leftarrow$ set of all signals $z$ s.t.
             $|\langle y_p, y_z \rangle| > \tau$ and
             $|\langle y_z, y_q \rangle| > \tau$.
            if $|S|$ mod 2 is 0 then
                $z_{min} \leftarrow$ signal in $S$ with
                smallest average inner
                product with $p$ and $q$.
                $S \leftarrow S \setminus \{z_{min}\}$ .
            end if
            if $|S| \geq 2$ and
                P2($S$, $r$, $s$, $\tau$) then
                $B \leftarrow \sum_{i=1}^{|S|} z_i z_i^T$ .
                $u \leftarrow$ EIGTOP($B$).
                if $\min_{a \in A} |a - u| > \epsilon_A$ then
                    $A \leftarrow A \cup \{u\}$.
                end if
            end if
        end if
    end while
    return $A$.
end procedure
```

We also formulate a different threshold ε_1 for use in P2 for the average correlation between signals in a cluster detected by P1. This threshold is used to filter out clusters that don't contain atoms sharing the same signal, and its formulation is based on intuition given by probabilistic estimations under some assumptions described in the Probabilistic Bounds section, with details in the Appendix.

DATA GENERATING PROCESS

In [1], Agarwal et al. test a data generating process where entries of D are drawn from $\mathcal{N}(0,1)$, the support of each column vector in X is chosen uniformly and independently from subsets of size s, and the non-zero values of each X column vector are chosen uniformly and independently from $[-2,-1] \cup [1,2]$.

Our data generating process is a choice of a true dictionary D and s-sparse matrix X. The signals generated are defined by $Y = DX$. This process is inspired by problems in classical signal processing–recovery of signals created by low coherence dictionaries. Consequently, a DCT matrix is chosen as the dictionary because it has low coherence. This is our first main modification of the data generating process of [1]. Like Agarwal et al., we consider the case where the signal matrix $Y \in \mathbb{R}^{d \times n}$ has $d < n$.

Our second main modification is as follows: we choose three integers α, β, and γ, with γ positive. Like Agarwal et. al., we choose the locations of the non-zero entries uniformly and independently from the subsets of size s. We set one non-zero entry of each column to be β, and the rest drawn uniformly independently from $\{-\alpha\} \cup \{\alpha\}$. So the non-zero entries of each column are in the list $\{-\alpha, \alpha, \beta\}$. We further insist that no more than n/γ of the β-valued elements exist in the same entry of any subset of column vectors of X. This condition ensures that β-valued elements are not clustered together in dimension. Finally, this data generating process implies that any procedure clustering these vectors by the correlation function demands an additional condition; that

$$\beta^2 - 2\alpha\beta > 2\alpha^2(s-2).$$

This condition is derived in the Correlation Threshold section.

Our restriction to integer-valued elements, restriction on the relative sizes and dimensional distributions of the elements, and use of a low coherence dictionary are the major differences between our and Agarwal et al.'s data process. However, our additional free parameters allow flexibility as well.

PROBABILISTIC BOUNDS

In P1, our goal is to find clusters of signals that may share the same atom; that is, signals y_p and y_q "share an atom" if X_{sp} and X_{sq} are both non-zero. These potential clusters are constructed by selecting pairs of signals (y_p, y_q) that have large inner product (lines 10–11), and then finding other signals that have large inner product with each signal in the pair (lines 14–16). Once we identify correlated clusters of signals in P1, we extract an atom through the process in lines 25–29 [1].

Under our proposed data process and correlation threshold, lines 10–16 select clusters with signals that all share the same atom with coefficient β. However, under other data processes, there is no assurance that this will happen. Consequently, this process may select "bad" clusters which contain signals that might not (a) share a single unique atom and (b) have non-negligible contributions from other atoms (in our data process, this implies that the coefficient on these atoms is greater than α).

To gain insight into the probability that each cluster identified by $P1$ is "good", we analyze the probability that any pair of signals in a cluster shares the same unique atom, but not any other atoms. We introduce the following scenario and events to formalize this problem: pick two signals from the data, y_p and y_q, and consider two arbitrary signals y_i and y_j. Define the following events:

- $SU(y_p, y_q)$: The sums that represent y_p and y_q share exactly one unique atom.
- $SU(y_i, y_j)$: The sums that represent y_i and y_j share exactly one unique atom.
- E_1: y_i shares *exactly* one atom with y_p, and y_i shares *exactly* one atom with y_q. Also, y_j shares *exactly* one atom with y_p, and y_j shares *exactly* one atom with y_q.
- F_1: y_i shares *at least* one atom with y_p, and y_i shares *at least* one atom with y_q. Also, y_j shares *at least* one atom with y_p, and y_j shares *at least* one atom with y_q.

$SU(y_p, y_q)$ corresponds to picking the initial pair of correlated signals, as we know that if $|\langle y_p, y_q\rangle| > \tau$, then y_p and y_q share at least one atom. To make the analysis tractable, we assume that this shared atom is the only atom y_p and y_q share, even though P2 may select pairs that share more than one unique atom. Additionally, we know that if, for some signal y_z, $|\langle y_p, y_z\rangle| > \tau$ and $|\langle y_q, y_z\rangle| > \tau$, then y_z shares at least one atom with each of y_p and y_q. F_1 defines this event for some pair of signals (y_i, y_j). We are interested in the following probability: given that $SU(y_p, y_q)$ and F_1 have occurred, what is the probability that the events E_1 and $SU(y_i, y_j)$ will occur? We are interested in analyzing

$$P\big[SU(y_i, y_j) \cap E_1 \big| F_1 \cap SU(y_p, y_q)\big]. \quad (1)$$

In other words, if we have a pair (y_p, y_q) of signals which share a unique atom, and another candidate pair (y_i, y_j), each of which shares at least one atom with y_p and y_q, what is the probability that (y_i, y_j) share the same unique atom with each other (this is $SU(y_i, y_j)$) that they uniquely share with y_p and y_q (this is E_1)?

Knowing a lower bound bound on (1) allows us to select only those clusters in which enough candidate pairs of signals from the cluster are correlated with each other to, on average, share a unique atom. To establish the lower bound that ensures that the cluster shares a unique atom, we split (1) up into

$$\frac{P[SU(y_i, y_j) \cap E_1 \cap F_1 | SU(y_p, y_q)]}{P[F_1 | SU(y_p, y_q)]}, \quad (2)$$

by the definition of conditional probability. Each probability in (2) is bounded separately assuming that the signals are independent and atoms randomly distributed amongst signals (with details in the Appendix). These bounds are combined to form the lower bound on (1):

$$P[SU(y_i,y_j) \cap E_1 | F_1 \cap SU(y_p,y_q)] > 1 - \tfrac{19s^3}{r}. \quad (3)$$

In P2, we again detect correlation by checking to see if two signals have inner product with a magnitude greater than τ. We count all signals that pass this criterion, and use this count as an empirical estimator of (1). If this empirical estimation of (1) is above the lower bound on (1) required for the signals to share a unique atom, P2 returns TRUE, and we continue on to the rest of P1 (that is, lines 25–29 which extract an atom).

It is notable that this procedure does not depend on the data process we establish above, and that τ may be calculated differently for a separate data process without affecting the above calculations.

CORRELATION THRESHOLD

We would not like to extract atoms from a cluster formed by P1 if the signals do not all share an atom. We calculate a correlation threshold in order to detect and reject clusters of signals fitting this description. We derive the correlation threshold based on the worst-case inner product of two vectors which do not share the same atom. If two vectors in Y do not share the same atom, the β-valued element is not contained in the same entry. Consequently, we know that the inner product will be at most

$$\tau_1 \quad = 2\alpha\beta + \alpha^2(s-2)$$

in magnitude, as each β-valued entry may, by chance, be multiplied by a signal with α in the same entry with the same sign, leaving $s-2$ potential α-valued entries with the same sign. In these calculations, we ignore the elements of D, as each element in D is bounded in magnitude by 1, and therefore the product of any of these elements will not affect this upper bound on the inner product between two vectors in Y that do not share atoms with coefficient β.

Importantly, we must make sure that this threshold does not bar clusters comprised entirely of signals that share the same atom from being selected. In this case, without loss of generality, the worst-case result is that the β-valued entry is positive, and that the α-valued entries are all of opposite sign; therefore these entries decrease the magnitude of the inner product of two signals which share an atom.

Thus we gain the restriction that

$$\beta^2 - \alpha^2(s-2) > \tau_1 \Leftrightarrow \beta^2 - 2\alpha\beta > 2\alpha^2(s-2).$$

EXPERIMENTS

We run experiments with our data process to show that under some conditions, our modified clustering + eigenvector-based atom extraction procedure can fully recover the atoms of the original dictionary and reasonably reconstruct the original signal matrix Y without an alternating minimization step. To reconstruct the data, after the recovered dictionary D_{pred} is created by $P1$, we use OMP for sparse coding to form a recovered sparse matrix X_{pred}. We use OMP because it is a fast and an easily comparable baseline used widely in the literature. We set the following parameters of our model: $n = 2048$, $d = 256$, $s = 3$, $\beta = 10$, $\alpha = \{1,3\}$, $\gamma = 256$, and $r = 256$. To be clear, this means that the dictionary has 256 atoms, the collection of data has 2048 signals, and each signal in $\mathbb{R}^{256}$ is 3-sparse. We remind the reader that a signal is 3-sparse means that it is a linear combination of at most 3 atoms, and setting the value of β to 10 means the largest of the three coefficients is 10. The original dictionary D for the data generating process is a Discrete Cosine Transform (DCT) matrix. We use the DCT dictionary because it is a standard choice in the literature. We choose n, d, s based on similar values used in the literature [2], and we choose α, β, γ, and r to illustrate the reconstruction of signals. We implement each experiment in MATLAB on a computer with a Core i7-4650U processor and 8GB of RAM.

We use two metrics to judge the efficacy of our algorithm. The first is the *recovery rate*, $\nu = n_x/r$, where n_x is the number of atoms extracted by $P1$ that have inner product of at least 0.99 with at least one atom in the original dictionary. Our second metric is the relative error of the reconstruction of Y, defined as

$$100\frac{||D_{pred}X_{pred} - Y||_2}{||Y||_2}$$

where $||\cdot||_2$ indicates the spectral norm of a matrix (aka 2-norm, or largest singular value), and D_{pred} and X_{pred} are, respectively, the dictionary recovered by P1, and the sparse matrix recovered by OMP against D_{pred}.

For the $\alpha = 1$ case, our correlation threshold is $\tau_1 = 2(1)(10) + (1)(1) = 21$. We run our procedure P1 to construct the dictionary, and follow it with OMP to reconstruct X. We perform this experiment five times and average the metrics below. Our algorithm scans through all possible clusters of signals and stops when it has extracted 256 atoms. It therefore has the significant benefit of determining the number of atoms in the dictionary without a priori knowledge. Each of these 256 atoms has correlation of at least 0.9991 with at least one atom in the original DCT dictionary–in other words, $\nu = 1$, as we correctly recover every single atom from the original dictionary. Similarly, we reconstruct the data as well, with a relative error rate of 8.19%.

For the $\alpha = 3$ case, our correlation threshold is $\tau_1 = 2(3)(10) + (1)(9) = 69$. We use the same experimental setup as in the $\alpha = 1$ case. We perform this experiment five times. In all five runs, P1 stops after recovering all 256 atoms, again illustrating the automatic atom number determination that this approach enjoys. We also again recover all atoms, with the lowest inner product for a single recovered atom being equal to 0.9931. Thus $\nu = 1$. To be clear, the dictionary has 256 atoms, the collection of data has 2048 signals, and each signal in $\mathbb{R}^{256}$ is 3-sparse. In the $\alpha = 3$ case, our relative error rate is 21.64%. While this is larger than the $\alpha = 1$ case, we suspected that, due to the near-perfect atom recovery rate, the error must mostly be due to the sparse coding process governed by OMP. Indeed this is the case: we found that although X_{pred} recovers almost every β-valued entry, it occasionally has flipped signs. Because in this work we mainly focus on the dictionary construction method, we do not attempt to improve this error rate; however, it is possible that it may be improved through the use of a sparse coding method more sophisticated than OMP.

CONCLUSION

We proposed modifications to the dictionary initialization algorithm of Agarwal et al. and a corresponding data generating process and correlation threshold under which full atom recovery and reasonable data reconstruction is possible. We also give a probabilistic bound that can aid in the evaluation of clusters created from different data generating processes.

Our findings show that it is indeed possible to perform dictionary learning using only a clustering and atom extraction initialization algorithm paired with a sparse coding algorithm. This allows us to bypass the requirement of running an alternating minimization operation, and may indicate that other data processes enjoy this same empirical performance. Although we utilize OMP in this work, we hypothesize that more sophisticated sparse coding methods may further reduce reconstruction error as well. We leave the construction of new data generating processes and correlation thresholds and the use of other sparse coding algorithms to future research.

ACKNOWLEDGEMENTS

The authors are grateful for the financial support of the Undergraduate Research Assistant Program (URAP) grant from DePaul University's College of Science and Health.

REFERENCES

[1] A. Agarwal, A. Anandkumar, P. Jain, and P. Netrapalli, *Learning sparsely used overcomplete dictionaries*, 27th Conference on Learning Theory, (2014) 123–137.

[2] M. Aharon, M. Elad, and A. Bruckstein, *K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Transactions on Signal Processing. **54** (2006) no. 11, 4311–4322.

[3] C. E. Bouton, A. Shaikhouni, N. V. Annetta, M. A. Bockbrader, D. A. Friedenberg, D. M. Nielson, et al., *Restoring cortical control of functional movement in a human with quadriplegia*, Nature. **533** (12 May 2016), 247–250.

[4] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, (2004), 727 pages.

[5] T. T. Cai and L. Wang, *Orthogonal matching pursuit for sparse signal recovery with noise*, IEEE Transactions on Information Theory. **57** (2011), no. 7, 4680–4688.

[6] K. Engan, S.O. Aase, and J.H. Husoy, *Method of optimal directions for frame design*, Proceedings of 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. **5** (1999), 2443–2446.

[7] R. Jenatton, J. Mairal, G. Obozinski, and F. R. Bach, *Proximal methods for sparse hierarchical dictionary learning*, 27th International Conference on Machine Learning, (2010), 487–494.

[8] J. Mairal, G. Sapiro, and M. Elad, *Learning multiscale sparse representations for image and video restoration*, Multiscale Modeling & Simulation. **7** (2008), no. 1, 214–241.

[9] V. Naumova and K. Schnass, *Dictionary learning from incomplete data for efficient image restoration*, 2017 25th European Signal Processing Conference (EUSIPCO), (2017), 1425–1429.

[10] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society, Series B. **58** (1994), 267–288.

[11] I. Tosic and P. Frossard, *Dictionary Learning*, IEEE Signal Processing Magazine. **28** (2011), no. 2, 27–38.

[12] Q. Zhang and B. Li, *Discriminative K-SVD for dictionary learning in face recognition*, 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (2010), 2691–2698.

APPENDIX

In this appendix, we provide in full detail the calculations for the two-signal probabilistic bound.

PROBABILISTIC ANALYSIS OF TWO SIGNALS

Pick two signals from the data, y_p and y_q, and consider two arbitrary signals y_i and y_j. Define the following events:

- $SU(y_p, y_q)$: The sums that represent y_p and y_q share exactly one unique atom.
- $SU(y_i, y_j)$: The sums that represent y_i and y_j share exactly one unique atom.
- E_1: y_i shares *exactly* one atom with y_p, and y_i shares *exactly* one atom with y_q. Also, y_j shares *exactly* one atom with y_p, and y_j shares *exactly* one atom with y_q.
- F_1: y_i shares *at least* one atom with y_p, and y_i shares *at least* one atom with y_q. Also, y_j shares *at least* one atom with y_p, and y_j shares *at least* one atom with y_q.

Note that F_1 is just E_1, but "exactly" has been replaced with "at least." For the remainder of these calculations, we suppose that $SU(y_p, y_q)$ has occurred, and that, WLOG, say $a_{p_s} = a_{q_s}$. Denote this shared atom by a_s. Additionally, note that both E_1 and F_1 occur if y_i and y_j share just a_s with y_p and y_q. Finally, note that if E_1 has occurred, then F_1 has occurred as well.

We'd like to find a lower bound on

$$P[SU(y_i, y_j) \cap E_1 | F_1 \cap SU(y_p, y_q)].$$

By the definition of conditional probability, this probability is equal to

$$\frac{P[SU(y_i, y_j) \cap E_1 \cap F_1 | SU(y_p, y_q)]}{P[F_1 | SU(y_p, y_q)]},$$

but because E_1 satisfies the requirements for F_1 to occur, this is equal to

$$\frac{P[SU(y_i, y_j) \cap E_1 | SU(y_p, y_q)]}{P[F_1 | SU(y_p, y_q)]}.$$

So, to find a lower bound on this probability, we need to find a lower bound on $P[SU(y_i, y_j) \cap E_1 | SU(y_p, y_q)]$ and an upper bound on $P[F_1 | SU(y_p, y_q)]$.

Lower bound

We define $N(y_i)$ to be the set of atoms that construct the signal y_i. If each of y_i and y_j choose a_s, the atom that y_p and y_q share, and if they then don't choose any more atoms from $N(y_p) \cup N(y_q)$, we see that this is one way that $SU(y_i, y_j) \cap E_1$ occurs. Therefore, the probability of this constitutes a lower bound on $P[SU(y_i, y_j) \cap E_1 | SU(y_p, y_q)]$, and we see that

$$\begin{aligned} &P[SU(y_i, y_j) \cap E_1 | SU(y_p, y_q)] \\ &\geq \frac{\binom{1}{1}\binom{r-2s+1}{s-1}\binom{1}{1}\binom{r-3s+2}{s-1}}{\binom{r}{s}^2} \\ &= \frac{s^2}{r^2}[\frac{(r-s)!}{(r-1)!}]^2 \frac{(r-2s+1)!}{(r-4s+3)!}, \end{aligned}$$

where the usage of the combinatorial definition of probability requires us to assume that the choices of y_i are independent of the choice of y_j, and that the probabilities of choosing any of the $s-1$ atoms in either of the above binomial coefficients are uniform.

We need to approximate the ratio of factorials to produce a useful lower bound. We'll start by noting that if $r = 4s - 3 + k$ where k is a nonnegative integer, then

$$\begin{aligned}&\frac{(r-2s+1)!}{(r-4s+3)!} = \frac{(4s-3+k-2s+1)!}{(4s-3+k-4s+3)!}\\&= \frac{(2s-2+k)!}{k!}\\&= (2s-2+k)(2s-3+k)\cdots\\&\quad (s+k)\cdot(s+k-1)(s+k-2)\cdots(k+1).\end{aligned}$$

Note that there are always $2s-2+k-(k+1)+1 = 2s-2 = 2(s-1)$ terms in the last product. For example, if $r = 1000$ and $s = 10$, then

$$\begin{aligned}&\frac{(r-2s+1)!}{(r-4s+3)!} = \frac{981!}{963!}\\&= 964\cdot 965\cdots 972\cdot 973\cdot 974\cdots 981.\end{aligned}$$

Now consider that, with the same restrictions,

$$\begin{aligned}&\frac{(r-s)!}{(r-1)!} = \frac{1}{(r-1)\cdots(r-s+1)}\\&= \frac{1}{(4s-4+k)(4s-5+k)\cdots(3s-2+k)},\end{aligned}$$

which is always a product of

$$4s-4+k-(3s-2+k)+1 = s-1$$

terms. Then

$$\begin{aligned}&\left[\frac{(r-s)!}{(r-1)!}\right]^2 \frac{(r-2s+1)!}{(r-4s+3)!}\\&= \frac{(2s-2+k)(2s-3+k)\cdots(s+k)}{(4s-4+k)(4s-5+k)\cdots(3s-2+k)}\\&\quad\cdot\frac{(s+k-1)(s+k-2)\cdots(k+1)}{(4s-4+k)(4s-5+k)\cdots(3s-2+k)}\\&\geq \left(\frac{s+k}{3s-2+k}\right)^{s-1}\left(\frac{k+1}{3s-2+k}\right)^{s-1}\\&= \left(\frac{r-3(s-1)}{r-s+1}\right)^{s-1}\left(\frac{r-4(s-1)}{r-s+1}\right)^{s-1},\end{aligned}$$

where the last equality is reached by noting that $k = r - 4s + 3$ and rearranging.

Now applying the identity $1 - \frac{bx}{c-x} \geq \exp\left(\frac{-2bx}{c-x}\right)$, which is valid for $0 \leq b \leq 3$ and $c > 0$ for $0 \leq x \leq c/5$, we see that, with the restriction that $r \geq \max(5s, 4s-3) = 5s$,

$$\begin{aligned}
&\left(1-\frac{3(s-1)}{r-(s-1)}\right)^{s-1}\left(1-\frac{2(s-1)}{r-(s-1)}\right)^{s-1}\\
&\geq \exp\left(-\frac{6(s-1)^2}{r-(s-1)}\right)\exp\left(-\frac{4(s-1)^2}{r-(s-1)}\right)\\
&=\exp\left(-\frac{10(s-1)^2}{r-(s-1)}\right)\\
&\geq 1-\frac{10(s-1)^2}{r-(s-1)} \geq 1-\frac{11(s-1)^2}{r}
\end{aligned}$$

where the second to last inequality is by truncating the Taylor expansion of e^{-x}, and the last inequality holds because with $x = s-1$,

$$\begin{aligned}
&1-\frac{10x^2}{r-x} \geq 1-\frac{11x^2}{r}\\
&\Leftrightarrow 1-\frac{10x^2}{r-x}-(1-\frac{11x^2}{r}) \geq 0\\
&\Leftrightarrow \frac{11x^2}{r}-\frac{10x^2}{r-x} \geq 0
\end{aligned}$$

This inequality clearly holds for $x = 0$; we need to find out where it does not hold, so we find the positive roots of the function:

$$\begin{aligned}
f(x) &:= \frac{11x^2}{r}-\frac{10x^2}{r-x}\\
&= \frac{11x^2r-11x^3-10x^2r}{r(r-x)}\\
&= \frac{x^2(r-11x)}{r(r-x)}\\
&\Rightarrow f(x)=0 \text{ if } x=0,\frac{r}{11}.
\end{aligned}$$

This implies that the inequality holds for $0 \leq x \leq r/11$, or $1 \leq s \leq r/11+1$.

Now, we'd rather use s than $s-1$ in our inequality, and because

$$1-\frac{11(s-1)^2}{r} \geq 1-\frac{11s^2}{r},$$

we can. Therefore, putting this approximation back into the lower bound we had above, we achieve the lower bound

$$P[SU(y_i,y_j) \cap E_1 | SU(y_p,y_q)] \geq \frac{s^2}{r^2}\left[1-\frac{11s^2}{r}\right].$$

Upper bound

Now that we've computed the lower bound on the numerator, we'd like to find an upper bound on the denominator, $P[F_1|SU(y_p,y_q)]$.

We claim that for one signal (without loss of generality, we choose y_i) F_1 occurs in two ways, (a) and (b). Once we calculate the probability for one signal, we square this probability, because the choices of the other signal (let us say, y_q) are independent from those of y_i. Thus this argument rests on the assumption that y_i and y_j are independent. We again use the language of graph theory to demarcate these cases; consider the bipartite graph formed by of r nodes representing r atoms on one side, and two nodes representing y_p and y_q on the other. Edges between signal nodes and atom nodes indicate that the signal's sparse representation uses the atom. Then the neighborhoods $N(y_p)$ and $N(y_q)$ constitute the sets of atoms of y_p and y_q, respectively.

In (a), we suppose that y_i and y_j only choose from the atoms in $N(y_p) \cap N(y_q)$. Note that the only atom in this set is necessarily a_s. This gives the probability of the event that y_i and y_j choose exactly one atom. Since these choices are independent and symmetric, we can split up the choices between y_i and y_j; y_i chooses one atom from the intersection, then chooses $s-1$ atoms from the other $r-1$ atoms; after this, y_j does the same. Thus this argument rests on the assumptions that the probabilities of choosing from 1 atom in the intersection and the $r-1$ other atoms are independent, and therefore uniform, and therefore we can assert that

$$\frac{\binom{1}{1}\binom{r-1}{s-1}}{\binom{r}{s}} = \frac{s}{r}.$$

In (b), we calculate the probability that y_i and y_j choose 2 or more atoms from the $2s-1$ atoms in $N(y_p) \cup N(y_q)$. We need to calculate the probability of choosing at least least 2 atoms from this intersection. To do this, we rely on the assumption that choices of non-zero entries in a signal's sparse vector are uniform and independently chosen. This assumption is required for us to be able to use the combinatorial definition of probability. Under this assumption, we can then see that the probability of this event occurring is upper bounded by

$$\frac{(2s-1)(2s-1)\binom{r-2}{s-2}}{\binom{r}{s}}$$
$$= \frac{(2s-1)^2\binom{r-2}{s-2}}{\binom{r}{s}}$$
$$= (2s-1)^2\frac{s(s-1)}{r(r-1)} \leq \frac{s^2}{r^2}(2s-1)^2.$$

Though (a) and (b) are not mutually exclusive events, we can add their probabilities to reach an upper bound, and then square this upper bound to account for the choices of y_j:

$$P[F_1|SU(y_p, y_q)] \leq \left[\frac{s}{r} + \frac{s^2}{r^2}(2s-1)^2\right]^2$$
$$\leq \left[\frac{s}{r}(1 + \frac{s}{r}(2s-1)^2)\right]^2$$
$$\leq \frac{s^2}{r^2}\left[1 + \frac{4s^3}{r}\right]^2$$

Probability bound

We combine the lower bound on the denominator and the upper bound on the numerator to arrive at a lower bound for the probability; which is our initial goal.

We have that

$$\begin{aligned}
&P[SU(y_i, y_j) \cap E_1 | F_1 \cap SU(y_p, y_q)] \\
&= \frac{P[SU(y_i, y_j) \cap E_1 | SU(y_p, y_q)]}{P[F_1 | SU(y_p, y_q)]} \\
&\geq \frac{\frac{s^2}{r^2}\left[\frac{(r-s)!}{(r-1)!}\right]^2 \frac{(r-2s+1)!}{(r-4s+3)!}}{\left[\frac{s}{r} + \frac{s^2}{r^2}(2s-1)^2\right]^2} \\
&\geq \frac{\frac{s^2}{r^2}\exp\left(-\frac{10(s-1)^2}{r-(s-1)}\right)}{\frac{s^2}{r^2}\left[1 + \frac{4s^3}{r}\right]^2} \geq \frac{1 - \frac{11s^2}{r}}{\left[1 + \frac{4s^3}{r}\right]^2}.
\end{aligned}$$

We would like to get a total lower bound on in the form of a function $1 - Cs^3/r$ for some C. We suspect $C = 11 + 2(4) = 19$ to be relatively tight, but we need to show that for $s \in \mathbb{N}$,

$$\begin{aligned}
&\frac{1 - \frac{11s^2}{r}}{\left[1 + \frac{4s^3}{r}\right]^2} - \left[1 - \frac{19s^3}{r}\right] \\
&= \frac{s^2(11r^2 s - 11r^2 + 136rs^4 + 304s^7)}{r(r+4s^3)^2} \geq 0.
\end{aligned}$$

This is equivalent to showing that, for $s \in \mathbb{N}$, $1 \leq s \leq r/11 + 1$,

$$304s^7 + 136rs^4 + 11r^2 s \geq 11r^2$$

Because the LHS is smallest when $s = 1$, we must equivalently show that

$$304 + 136r + 11r^2 \geq 11r^2$$

which is clearly true. Therefore, for natural numbers s s.t. $1 \leq s \leq r/11 + 1$,

$$P[SU(y_i, y_j) \cap E_1 | F_1 \cap SU(y_p, y_q)] \geq 1 - \frac{19s^3}{r}.$$

This concludes the calculations for the lower bound.