Intelligent Sparse Representations for Speech*

Aditya Acharya† Karthik A Sankararaman‡ Manasij Venkatesh§

Abstract

We design a dictionary in which speech signals have a sparse representation. We utilize the property that speech is comprised of a fixed number of phonemes. The dictionary is a concatenation of the principal components of all these phonemes, and hence information about each phoneme is present in a block. Since any speech signal is a concatenation of phonemes, it can be represented as a linear combination of the columns of this dictionary. In particular, if we consider a small window of speech (containing no more than two phonemes), such a signal would ideally have a *block sparse* representation in the dictionary. The representation is obtained by solving a variation of the LASSO or basis pursuit denoising (BPDN) problem. We show that the representation is sparse enough to achieve compression. Finally, our intuition is that such a representation could also implicitly perform denoising.

1 Introduction

Speech is comprised of a fixed number of basic speech units called phonemes. An important characteristic of phonemes is that each of them occupy a unique frequency band, which makes them easily identifiable in the frequency domain. Therefore, to represent a phoneme in this domain it is crucial to know the components that represent the corresponding frequency.

In compressed sensing literature, an important assumption made is that signals have a sparse representation in some basis [CW08]. For example, images are known to have a sparse representation in a DCT basis or a wavelet basis. For images, using redundant representations and sparsity as driving forces for denoising of signals has drawn a lot of research attention [Don95, EA06]. The Lasso problem is sometimes referred to as the basis pursuit denoising problem (BPDN) for this reason.

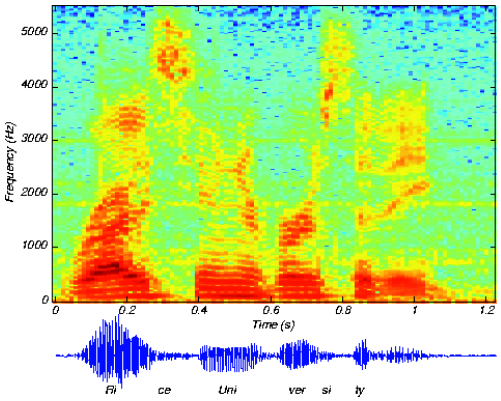

Figure 1: A typical speech spectrogram. The horizontal axis represents time, the vertical axis is frequency; a third dimension indicating the amplitude of a particular frequency at a particular time is represented by a heat map (red indicating higher values and blue lower values).

Our primary objective was to find a dictionary over which speech signals have a sparse representation. To do so, we represent human speech intelligently in a domain that allows us to exploit the inherent properties of speech. We solve a modification of the BPDN problem and test the denoising performance. Solving this problem also has the distinct advantage of identifying the phonemes present in the signal.

2 Prerequisites

A speech spectrogram is a representation of the spectrum of frequencies of a speech signal as it varies with time. Figure 1 shows a typical speech spectrogram. The horizontal axis represents time, the vertical axis is frequency; a third dimension indicating the amplitude of a particular frequency at a particular time is represented by a heat map (red indicating higher values and blue lower values).

*The codes and sample output can be found on our Github page at `http://goo.gl/f2f2f1`

†Department of Computer Science, University of Maryland. Email: acharya@cs.umd.edu

‡Department of Computer Science, University of Maryland. Email: kabinav@cs.umd.edu

§Department of Electrical and Computer Engineering, University of Maryland. Email: manasij@umd.edu

To obtain a frequency domain representation for the speech signal, we use the Discrete Cosine Transform (DCT). Formally, if x_i, $i = 1, \ldots, n$ represents a section of the speech signal, then X_k's given

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right], \ k = 0, \ldots, N-1,$$

are the DCT coefficients of that section of the speech signal. Assume, the X_k's are in a column vector. The DCT spectrogram is simply obtained by concatenating the DCT coefficient column vectors for consecutive sections(possibly overlapping) of speech.
Notations:

- We refer to principal components that correspond to the highest eigenvalues as the *top principal components*.
- Θ represents the set of all phonemes. Any phoneme is represented as $/\theta/$.

3 Dictionary design

In this section, we discuss the dataset used and the details involved in designing the dictionary.

3.1 Dataset

The TIMIT speech corpus that consists of time-aligned orthographic, phonetic and word transcriptions is used. TIMIT contains a total of 6300 sentences (5.4 hours), consisting of 10 sentences spoken by each of 630 speakers from 8 major dialect regions of the United States. All sentences are manually segmented at the phone level. The dataset has been annotated with 38 phonemes. The sampling rate is 16000Hz. The training set contains 3696 sentences from 462 speakers (3.14 hours). The test set contains 1344 sentences from 168 speakers (0.97 hours). The training and test sets do not overlap.

The duration of phonemes vary, but are no less than 50ms for the English language. We consider 20ms segments to ensure that it contains no more than two phonemes. Since one phoneme can end and another can begin in one time window, it is impossible to ensure that there is only one phoneme per window. The sampling rate is 16000Hz, we have 320 samples in a 20ms time window, that is, a *speech vector* of length 320. In this report, we refer to the DCT coefficients of this vector as y.

3.2 Principal Component Analysis

We first extract individual phonemes from a database. Principal component analysis (PCA) is performed on the DCT coefficients of *speech vectors* of these phonemes. An important intuition to have about PCA is as follows: When PCA is performed on the DCT coefficients of a particular phoneme, the top principal components contain information about the frequency band that the particular phoneme occupies. Since the frequency band occupied by a phoneme is small, it is sufficient to pick a small number of principal components.

Let the PCA basis of a particular phoneme be denoted by E. Based on eigenvalue analysis, the top k principal components of a phoneme $/\theta/$ are selected. $E^{(\theta)}$ is the *reduced principal component matrix*, and is no longer a basis. Although this matrix does not allow for exact reconstruction, it is useful for good estimation. The dictionary D is a concatenation of the reduced principal component matrices for all the phonemes, that is, a concatenation of the $E^{(\theta)}$'s. Each $E^{(\theta)}$ is a *block*, with k columns. Thus, the *block size* is k.

4 Convex Program Formulation

Let y be the section of the signal for which we want to have a sparse representation x in the dictionary D. Ideally, the optimization problem P_0 to obtain a block sparse solution is

$$\begin{aligned} \underset{T \in \mathbb{R}^m, x \in \mathbb{R}^n}{\text{minimize}} \quad & \|T\|_0 \\ \text{subject to} \quad & Dx = y \\ & T_i = \sqrt{\sum_{k \in \text{ block } i} x_k^2}, \quad i = 1, \ldots, m \end{aligned} \tag{P_0}$$

where m is the number of phonemes, block i represents the i^{th} block or the components of x corresponding to the i^{th} phoneme, and $\|\cdot\|_0$ is the cardinality.

The above problem is nonconvex and difficult to solve. Also, note that $Dx = y$ may render P_0 infeasible since D is a concatenation of the reduced principal component matrices and may not necessarily form a basis. However, y can be closely estimated by using D, even with the block sparse constraints. $\|Dx - y\|_2$ represents the error we want to minimize. Let the estimate of y be $\hat{y} = Dx$. Now, P_0 is relaxed as the following minimization problem P_1

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & \sum_{i=1}^{m} \sqrt{\sum_{k\in \text{ block } i} x_k^2} \\ \text{subject to} \quad & \|Dx - y\|_2 < \epsilon, \end{aligned} \tag{P_1}$$

which is convex. The sparsity "$\ell_{2,0}$" norm minimization problem in P_0 is relaxed to an $\ell_{2,1}$ norm minimizaton problem. P_1 is the closest possible convex problem to P_0. If D satisfies certain properties [DE03], the above problem gives us the optimal sparse solution. Note that P_1 is closely related to the LASSO or the BPDN problem, but has block sparsity constraints.

5 Methodology

In this section we will describe the various algorithms we used for achieving our goals. In the experiments section, we will provide the results for each of these methods and give a comparative study.

5.1 Solving the Convex Programming Formulation

The method we tried was to directly solve the convex programming relaxation P_1 to the original problem. We used the CVX toolbox to solve the problem. The advantage of this method was that solving this program gives us reasonable block sparse solutions. Additionally, we can have a good control over the error rate. The family of optimal solutions obtained was close to the expected solutions. But the *biggest* drawback was that this method was not scalable. CVX was significantly slow and running this program for a single sound file took several hours of CPU time on a 4GB RAM machine.

5.2 LASSO

The next method we tried was the LASSO [Tib94], where we ignore the block sparsity constraints. The convex relaxation P_1 can be further relaxed as the following LASSO problem.

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}||Dx - y||_2 + \alpha||x||_1$$

Notice that, for an arbitrary value of α the value of the optimal solution to the original convex problem and the LASSO problem may not be the same. However, we know that there exists a value of α such that the two optimal solutions can be made same. But for this value of α, the obtained optimal solution may not have the other desired properties (such as reconstruction with minimized error). Hence, we fine tuned the parameter to obtain the best performance. In particular, define $\alpha_{\max} = ||D^T y||_\infty$. The value of α was set to $0.5 * \alpha_{\max}$.

The above LASSO problem was solved using the *Alternating Direction Method of Multipliers*(ADMM) [BPC+11]. The running times were improved by a factor of 100 as opposed to solving P_1 on CVX on the same 4GB RAM machine.

As described in the original paper of [BPC+11], the ADMM formulation of the above LASSO problem is

$$\begin{aligned} \underset{x,z\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) + g(z) \\ \text{subject to} \quad & x = z \end{aligned}$$

where $f(x) = \frac{1}{2}||Dx - y||_2$ and $g(z) = \alpha||z||_1$. The ADMM step is then

$$\begin{aligned} x^{k+1} &:= (D^T D + \rho I)^{-1}(D^T y + \rho(z^k - u^k)) \\ z^{k+1} &:= S_{\alpha/\rho}(x^{k+1} + u^k) \\ u^{k+1} &:= u^k + x^{k+1} - z^{k+1} \end{aligned}$$

Here, $S_{\alpha/\rho}$ is the elementwise soft-thresholding function $S : \mathbb{R} \to \mathbb{R}$ defined as follows

$$S_{\alpha/\rho}(a) = a\left(1 - \frac{\alpha}{a\rho}\right)_+$$

Additionally, $S_{\alpha/\rho}(0) = 0$.

5.3 Group LASSO

Notice that, in Section 5.2 even though the formulation enforces sparsity on the solution x, the obtained solution need not necessarily be *group sparse*. Recall that, the section of the signal is chosen such that at most two phonemes are present. Hence, we run the group LASSO, where the regularizer term is replaced by a ℓ_1-norm of the *groups*. More formally, with every group we first take the ℓ_2-norm of the variables in that group. Finally, the regularizer term is the ℓ_1-norm of these terms.

Mathematically, the group Lasso problem is written as the following program. Here, the size of each block is n/m.

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}||Dx - y||_2 + \alpha \sum_{i=1}^{m} \sqrt{\sum_{k\in\text{block}(i)} x_k^2}$$

The value α in the above program was fine-tuned to give the best performance. Again, defining $\alpha_{\max} = ||D^T y||_\infty$ we set $\alpha = 0.1\alpha_{\max}$ The above program was again solved using ADMM. As described in [BPC+11], the above formulation can be cast as the following ADMM problem

$$\begin{aligned} \underset{x,z\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) + g(z) \\ \text{subject to} \quad & x = z \end{aligned}$$

where $f(x) = \frac{1}{2}||Dx - y||_2$ and $g(z) = \alpha \sum_{i=1}^{m} \sqrt{\sum_{k\in\text{block}(i)} z_k^2}$.

The ADMM step is as follows

$$\begin{aligned} x^{k+1} &:= (D^T D + \rho I)^{-1}(D^T y + \rho(z^k - u^k)) \\ z^{k+1} &:= \mathbf{S}_{\alpha/\rho}(x_i^{k+1} + u^k) \\ u^{k+1} &:= u^k + x^{k+1} - z^{k+1} \end{aligned}$$

Here, the proximal operator $\mathbf{S}_{\alpha/\rho}$ is a vector generalization of the one in 5.2. In particular, it is a function $\mathbf{S} : \mathbb{R}^n \to \mathbb{R}^n$ defined as follows.

$$\mathbf{S}_{\alpha/\rho}(a) = \left(1 - \frac{\alpha}{||a||_2\rho}\right)_+ a$$

Additionally, $\mathbf{S}_{\alpha/\rho}(0) = 0$.

We transform the optimal solution x^* to a solution $\hat{x}^*$ by retaining only the values of the *two most significant blocks* and setting all other coordinates to zero. We use this as the final solution.

5.4 Orthogonal Matching Pursuit

Apart from the above three convex optimization techniques, we tried two greedy approaches. Although they were faster in terms of speed, their performance on the denoising task was understandably poor. In this sub-section we will describe one of those approaches, namely the *Orthogonal Matching Pursuit* (OMP) algorithm [TG07].

The OMP algorithm takes a matrix D of size $d \times n$ and a vector y which is d-dimensional as inputs. Additionally, it also takes an integer k which represents the sparsity of the output vector x. The algorithm outputs a n-dimensional vector x such that $Dx = \hat{y}$ is an estimate of the actual signal y. The complete algorithm is described in Algorithm 1.

Algorithm 1 OMP algorithm for sparse signal recovery

procedure OMP
  Initialiaze $r_0 = y$.
  Set index set $C_0 = \phi$
  for $t = 1, 2 \ldots, k$ **do**
    Find, breaking ties arbitrarily

$$\lambda_t = \arg \max_{j=1,2,\ldots,n} |\langle r_{t-1}, D_j \rangle|$$

    Update the used index set

$$C_t = C_{t-1} \cup \{\lambda_t\}$$

    Update chosen columns matrix

$$\mathcal{D}_t = [\mathcal{D}_{t-1} D_{\lambda_t}]$$

    Solve the following least squares problem to obtain a new current estimate

$$x_t = \arg\min_x ||y - \mathcal{D}_t x||_2$$

    Update the residual as follows

$$a_t = \mathcal{D}_t x_t$$

$$r_t = y - a_t$$

  end for
  The estimate x has non-zero entries at the indices $\lambda_1, \lambda_2, \ldots, \lambda_k$. The entry at index λ_i is the i$^{\text{th}}$ co-ordinate of x_i.
end procedure

5.5 Block Orthogonal Matching Pursuit

In this section, we will describe the other Greedy approach, known as the *Block Orthogonal Matching Pursuit* (BOMP) as introduced in [EB09]. This algorithm is a modification of OMP to account for the block sparsity constraints.

The input to BOMP is the same as in Algorithm 1. Algorithm 2 summarizes the algorithm. In the description of the algorithm, $D[i]$ refers to the columns corresponding to the block i. Additionally, $D[i]_j$ refers to the j$^{\text{th}}$ column in block i of matrix D.

Algorithm 2 BOMP algorithm for sparse signal recovery

procedure BOMP

Initialiaze $r_0 = y$.

Set index set $C_0 = \phi$

for $t = 1, 2 \ldots, k$ **do**

Find, breaking ties arbitrarily

$$\lambda_t = \arg \max_{j=1,2,\ldots,n/m} \left(\max_{i=1,2,\ldots,m} |\langle r_{t-1}, D[j]_i \rangle| \right)$$

Update the used index set

$$C_t = C_{t-1} \cup \{\lambda_t\}$$

Update chosen columns matrix

$$\mathcal{D}_t = [\mathcal{D}_{t-1} D[\lambda_t]]$$

Solve the following least squares problem to obtain a new current estimate

$$x_t = \arg\min_x ||y - \mathcal{D}_t x||_2$$

Update the residual as follows

$$a_t = \mathcal{D}_t x_t$$

$$r_t = y - a_t$$

end for

The estimate x has non-zero entries at the indices $\lambda_1, \lambda_2, \ldots, \lambda_k$. The entry at index λ_i is the i$^{\text{th}}$ co-ordinate of x_i.

end procedure

6 Experiments and results

We study two main characteristics.

1. Variation of estimation accuracy of various algorithms with block size.
2. Denoising performance of various algorithms with block size.

Recall that the *block size* refers to the number of principal components selected for each phoneme.

Given a signal y, the objective is to find its sparse representation x in the dictionary D such that the error $\|y - Dx\|_2$ is small. Let $\hat{y} = Dx$ be the estimate of y, the *estimation error* is therefore $\|y - \hat{y}\|_2$. We say the estimation accuracy is good, if the *estimation error* is low.

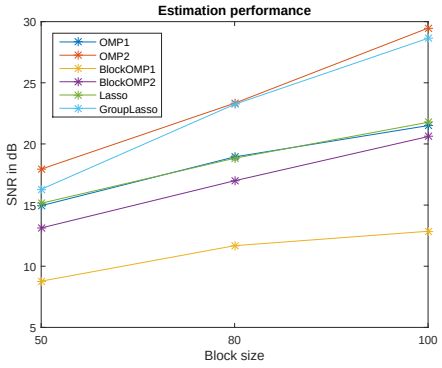

Figure 2: Variation of estimation accuracy of various algorithms with block size.

Also recall that performing PCA allows each block to estimate a particular phoneme to a high accuracy. However, if we do not pick all the principal components, we can not have a basis (we need n columns to have rank n). Hence, there will always be some error in estimation.

The metric used to compare the performance is signal-to-noise ratio (SNR), measured in dB. If $\hat{y}$ is the estimate of true signal y, SNR is calculated as

$$\text{SNR}(y, \hat{y}) = 20 * \log_{10} \frac{\|y\|_2}{\|y - \hat{y}\|_2}.$$

Note that the if the estimation error is low, SNR is high.

For the algorithms, note that OMP1 and BOMP1 have their sparsity set to the block size, and OMP2 and BOMP2 have sparsity set to twice the block size.

6.1 Estimation accuracy

Figure 2 shows the variation of the estimation accuracy with block size. As expected, if the number of principal components selected for each phoneme is increased, each block is closer to forming a basis. If each block was indeed a basis, the estimation error would be zero. Clearly, as the number of principal components are increased, the estimation error reduces, or the SNR increases.

Although each block is capable of estimating a particular phoneme, some of the principal components of different phonemes are similar (have strong correlation) since they may have overlapping frequency bands. A greedy algorithm (e.g. BOMP) which tries to find a a block that contains components with the strongest correlation, may select the wrong block be-

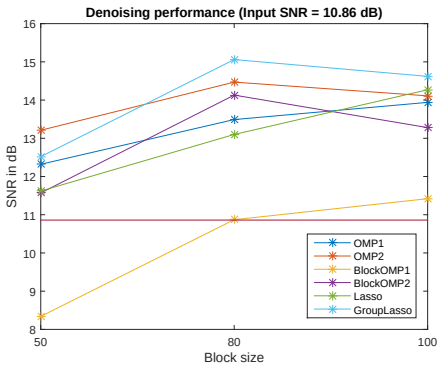

Figure 3: Denoising performance of various algorithms with block size. Input SNR = 10.86 dB.

cause of these correlations between blocks. This explains the poor estimation accuracy of the BOMP1 algorithm that picks only one block. Picking two blocks (BOMP2) improves the accuracy.

However, solving the Group LASSO problem, improves the accuracy significantly even though its average sparsity is about two blocks indicating that it is capable of finding the best two blocks. Solving this problem also provides us information about the phonemes present in the signal since the blocks that are active correspond to the phonemes that are present.

We also consider two algorithms that solves the problem without block sparsity constraints (the LASSO problem), using OMP1 and ADMM. Since we do not expect more than one block to be active, the average sparsity is set to around 1.2 times the block size for the ADMM. Thus, the performance of OMP1 and solving LASSO using ADMM are very similar.

The sparsity for OMP2 is set to twice the block size, and it is seen that it performs significantly better than OMP1. An important conclusion can be drawn from this: Sometimes, the principal components of the inactive phonemes can better estimate a part of the signal, as opposed to that of the active ones. This is further motivation to limit the size of each block.

6.2 Denoising performance

Consider the additive noise model

$$\eta = y + w,$$

where y is the clean signal, w is additive white Gaussian noise, and η is the noisy signal. The denoising problem is to estimate $\hat{y}$ from η such that $\|y - \hat{y}\|_2$ is minimized. The question we pose is, does imposing sparsity constraints denoise the signal η?

Figure 3 shows the variation of the denoising performance of various algorithms with block size. An important observation is that the denoising performance no longer improves with increasing block size. Since increasing the block size results in each block being closer to a basis, the estimate $\hat{y}$ is closer to η than to y, even when block sparsity constraints are imposed. Thus, the denoising performance becomes worse if the block size is increased from 80 to 100. Solving the Group LASSO problem results in best denoising performance among the algorithms and has a 4 dB improvement over the noisy signal. The denoising performance of the other algorithms follow the same trend as their estimation performance.

We summarize the results:

1. We achieve compression by obtaining a sparse representation.

2. We perform denoising by solving a modification of the BPDN problem.

3. Solving the Group LASSO problem results in best denoising performance, has good estimation accuracy, and also allows us to identify the phonemes present in the signal.

7 Conclusions

We intelligently designed a dictionary in which speech has a sparse representation by utilizing certain properties of speech. We obtained the sparse representation by solving a modification of the LASSO or BPDN problem. The advantages of solving this problem with block sparsity constraints has three advantages: compression, denoising, and phoneme identification. Future work involves utilizing language models for improving speed of the program. Two consecutive sections usually have the same phoneme active and hence, using this information should possibly give better results. We also aim at applying our method to phoneme identifications in multiple sources of sounds, in the future. For example, we believe that our method can be used in filtering out the spoken words from background music(An important feature which many current voice recognition softwares such as Siri, could benefit from).

References

[BPC+11] C S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers, 2011.

[CW08] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, 2008.

[DE03] David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

[Don95] David L Donoho. De-noising by soft-thresholding. *Information Theory, IEEE Transactions on*, 41(3):613–627, 1995.

[EA06] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12):3736–3745, 2006.

[EB09] Yonina C. Eldar and Helmut Bölcskei. Block-sparsity: Coherence and efficient recovery. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009 : ICASSP 2009 ; 19 - 24 April 2009, Taipei International Convention Center, Taipei, Taiwan ; proceedings*, pages 2885–2888, Piscataway, NJ, April 2009. IEEE.

[TG07] Joel A. Tropp and Anna C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE TRANS. INFORM. THEORY*, 53:4655–4666, 2007.

[Tib94] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.