1 Introduction

1.1 Objective of Blind Deconvolution

The goal of the blind deconvolution is recovering two unknown vectors, $\boldsymbol{w}$ and $\boldsymbol{x}$, of length $\mathbf{L}$ from their circular convolution[AAR14].

1.2 Significance of Blind Deconvolution

According to [SL19], blind deconvolution is an extremely important problem in multiple areas of engineering, including signal processing, communication engineering, image processing, and audio processing.

A straightforward example of blind deconvolution problem is given in [AAR14] in terms of blind channel estimation in communications:

An encoded message $\boldsymbol{x}$ passes through an unknown linear time-invariant channel $\boldsymbol{w}$, and the end user receives their convolution as a result. Under this context, by only knowing the received convolution vector $y = w * x$, we use blind deconvolution techniques to recover both the message vector $\boldsymbol{x}$ and the channel vector $\boldsymbol{w}$.

2 Methods and Techniques

2.1 Overview

Blind deconvolution is an ill-posed problem, meaning that the solution is not unique without additional assumptions or constraints.

The approach of [AAR14] assumes knowledge about the subspaces which $\boldsymbol{w}$ and $\boldsymbol{x}$ reside. It then use this knowledge to convert the non-linear convolution on $\boldsymbol{w}$ and $\boldsymbol{x}$ to a linear operation on another variable $\boldsymbol{X}$. It then introduces relaxation on the original problem to make it convex, thus $\boldsymbol{X}$ can be recovered as its global minimizer. Finally, $\boldsymbol{w}$ and $\boldsymbol{x}$ are recovered from $\boldsymbol{X}$, via very simple math operations.

2.2 Assumption

The approach is feasible only if we know the subspaces which $\boldsymbol{w}$ and $\boldsymbol{x}$ reside. With

$$w = Bh \quad h \in \Re^K, B \in \Re^{L \times K}$$

$$x = Cm \quad m \in \Re^N, C \in \Re^{L \times N}$$

Although we do not know $\boldsymbol{w}$ and $\boldsymbol{x}$ except their dimensions, we do have full knowledge about $\boldsymbol{B}$ and $\boldsymbol{C}$.

2.3 Linear Equivalent

With the knowledge about $\boldsymbol{B}$ and $\boldsymbol{C}$, we only need to find $\boldsymbol{h}$ and $\boldsymbol{m}$ to recover $\boldsymbol{w}$ and $\boldsymbol{x}$, and we can then convert the convolution to a linear operation.

$$y = w * x = \begin{bmatrix} Circ(C_1)B & Circ(C_2)B & ... & Circ(C_N)B \end{bmatrix} \begin{bmatrix} m(1)h \\ m(2)h \\ ... \\ m(n)h \end{bmatrix}$$

Where $circ(C_n)$ is the matrix equivalent of the circular convolution operation. To simplify the expression, we can get rid of the circular matrix by introducing the L-point normalized DFT matrix

F. Multiply F to the left side of the equation does not alter $\boldsymbol{h}$ and $\boldsymbol{m}$, neither does it change the magnitude of $\boldsymbol{y}$. As a result, recovering $\boldsymbol{h}$ and $\boldsymbol{m}$ from $\boldsymbol{y}$ is the same as from $\hat{\boldsymbol{y}}$.

$$\hat{y} = Fy = \begin{bmatrix} \Delta_1 \hat{B} & \Delta_1 \hat{B} & ... & \Delta_1 \hat{B} \end{bmatrix} \begin{bmatrix} m(1)h \\ m(2)h \\ ... \\ m(n)h \end{bmatrix}$$

Where

$$\Delta_n = diag(\sqrt{L}\hat{C}_n)$$
$$\hat{C} = FC$$
$$\hat{B} = FB$$

The left side of (2) is a matrix of dimension $(L, K \times N)$, and the right side is a column vector of length $(K \times N)$. Also, the column vector is the rows of the outer product of $\boldsymbol{m}$ and $\boldsymbol{h}$, $X = m \otimes h$, vertically stacked together. Therefore, the convolution could be perceived as a linear operation $\hat{y} = A(X)$, where the input is the outer product with dimension $(K \times N)$, and the output is a vector $\hat{y}$ with dimension L.

2.4 Convex Relaxation

To recover $(K \times N)$ variables from a system of linear equations, theoretically we need $(K \times N)$ linear equations, with the coefficient vector of each linear equation linearly independent with each other. In this problem, this corresponds to $L \geq K \times N$.

To recover the variables with $L << K \times N$, we convert the problem into an optimization problem over a feasible set, the minimizer of which will be attempted solution to the original problem.

First, convert the problem into a least square problem with feasibility constraint equivalent to $\hat{y} = A(X)$.

$$\begin{aligned} \min_{m,h} \quad & ||m||_2^2 + ||h||_2^2 \\ \text{subject to} \quad & \hat{y}(l) =< \hat{c}_l, m >< h, \hat{b_l} >, l = 1, ..., L \end{aligned} \tag{1}$$

Where $\hat{c}_l$ is the lth column of $\hat{C}$, $\hat{b}_l$ is the lth column of $\hat{B}$. This conversion is possible because of the transformation with L-point normalized DFT matrix F done earlier.

This problem is not convex, as its feasible set is not convex. In order to convexify it, we first compute the dual of this problem, and then compute the dual of the dual. This way, we convert it back to a convex optimization problem that is very close to the original problem. The convex relaxation of the original problem is minimizing the nuclear norm of outer product X, subject to observation result $\hat{\boldsymbol{y}}$

$$\begin{aligned} \min_{X} \quad & ||X||_* \\ \text{subject to} \quad & \hat{y} = A(X) \end{aligned} \tag{2}$$

2.5 Recovering from Outer Product

Assumes we obtain the unique solution of outer product $X = m \otimes h$, now we need recover $\boldsymbol{m}$ and $\boldsymbol{h}$ with it.

An element-wise representation of the outer product matrix is

$$X = m \otimes h = \begin{bmatrix} h_1m_1 & h_1m_2 & ... & h_1m_n \\ h_2m_1 & h_2m_2 & ... & h_2m_n \\ . & . & ... & . \\ . & . & ... & . \\ . & . & ... & . \\ h_km_1 & h_km_2 & ... & h_km_n \end{bmatrix} \tag{3}$$

We see that rows of x are product of element of h and vector m. Therefore, we can compute the ratio of elements in h from any column of X. Similarly, we can obtain the ratio of elements in m from any row of X. This way, with the additional knowledge about the norm of any one of $\boldsymbol{m}$ or $\boldsymbol{h}$, we can easily decode them from X.

3 Algorithm Result

3.1 Probabilistic Uniqueness

We have to keep in mind that original problem is different than the convex relaxation, and the fact that we do not have enough observations to guarantee exact recovery.

Consequently, the solution of the relaxed convex problem can become the unique solution of the original problem only under certain probability and constraints.

3.2 Constraints

The constraints for the probabilistic uniqueness of the result are expressed in terms of the properties of subspace matrix $\boldsymbol{B}$ and $\boldsymbol{C}$. The experiment setting has certain prerequisite properties on the matrices, and the probabilistic uniqueness guarantee puts additional constraints on the coherence and sparsity of the matrices.

3.2.1 Basic Prerequisites

The matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ has to satisfy some requirements to qualify for the experiment setting.

First, channel response vector $\boldsymbol{w}$ is time limited to Q ($K \leq Q \leq L$), meaning that the last Q out of L elements of $\boldsymbol{w}$ are zero. As a result, last Q out of L rows of $\boldsymbol{B}$ are also zero. In addition, the columns of $\boldsymbol{B}$ are assumed to be orthonormal, this does not violate generality of the result since orthonormal basis exists in every subspace.

Second, we assume codebook matrix $\boldsymbol{B}$ is randomly generated, and its columns are linearly independent so that it forms a basis of a subspace. One way to satisfy this condition is to let

$$C[l, n] \sim \mathcal{N}(0,\, L^{-1})$$

3.2.2 Coherence

In fourier domain, the coherence of matrix $\hat{\boldsymbol{B}}$ and vector $\hat{\boldsymbol{w}}$ have to satisfy certain conditions to ensure the probabilistic uniqueness.

The coherence of matrix $\hat{\boldsymbol{B}}$ is defined by

$$\mu^2_{max} = \frac{L}{K} \max_{1 \leq l \leq L} ||\hat{b}_l||^2_2$$

We can see that the coherence of matrix is how concentrated its column norm is. The coherence increases when more energy is concentrated in one specific column.

The coherence of vector $\hat{\boldsymbol{w}}$ is defined by

$$\mu_h^2 = L \max_{1\leq l\leq L} |\hat{w}(l)|^2 = L \max_{1\leq l\leq L} | < h, \hat{b}_l > |^2$$

To qualify for the probabilistic uniqueness of the solution, the coherence have to satisfy

$$max(\mu_{max}^2 K, \mu_h^2 N) \leq \frac{L}{C_\alpha'' log^3 L}$$

We can see that low coherence of $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{w}}$ makes the constraint easier to satisfy. The constraint is also easier to satisfy when L is large in terms of subspace sizes K and N, which means the reliability of experiment increases with more observation.

3.2.3 Sparsity

There is also a constraint on $\boldsymbol{B}$'s sparsity in the experiment setting, which is partly expressed in terms of $\boldsymbol{B}$'s coherence. Specifically,

$$Q \geq C_\alpha M log(L) log(M)$$

where

$$M = max(\mu_{max}^2 K, \mu_h^2 N)$$

and

$$L/Q \geq log(C_\alpha^{'} \sqrt{NlogL})/log2$$

The first constraint is more easily satisfied with smaller M, which means less coherence of $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{w}}$. The second constraint is more easily satisfied with less N, which is equivalent to a less complex codebook for generating the message.

4 Numerical Experiments

4.1 Noiseless Blind Deconvolution

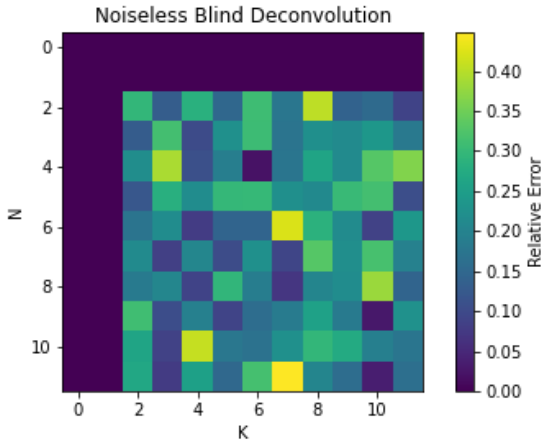

Figure 1: Phase Transition Plot of Noiseless Blind Deconvolution

Due to limited computing ability on local machine, Monte Carlo simulation with $L = 64$ and $(N, K) \in [2, 12]$ and 5 trials per setting is conducted. For each trial, a set of matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ are generated at random, as well as a set of vectors $\boldsymbol{h}$ and $\boldsymbol{m}$. The computation time for this set up is usually around 20 minutes.

Unfortunately, the result is not as good as what [AAR14] presented. With closer inspection of recovered outer product $\boldsymbol{X}^*$, I found out that most of the entries are recovered very accurately. However, for some very few entries the absolute difference could shoot up to the magnitude of 10^0. This caused the relative error to increase dramatically, usually to the level of 0.1-0.2. Since the recovery error is usually above 0.2, the relative error is plotted directly instead of the boolean value.

Also, my result does not contain a sharp boundary that differentiate the recovery quality. A potential explanation for this is that recovered signals generally contain a few entry where absolute difference is very large. Consequently, the plot becomes very noisy and thus blurred the boundary.

4.2 Noisy Blind Deconvolution

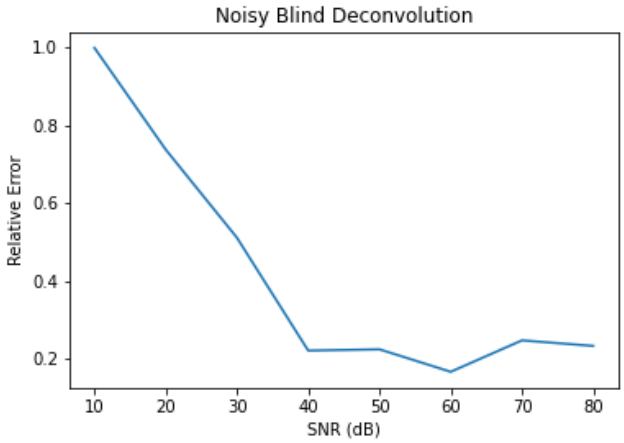

Figure 2: Error vs SNR in Blind Deconvolution

In this Monte Carlo simulation, $L = 64, N = 12, L = 12$, $SNR \in [10, 80]$ with step size of 10dB, and 10 trials are conducted for each parameter setting. When SNR is relatively small (less than 60dB), the computation time drops dramatically. This is because the constraint of the convex optimization is relaxed with the upper bound of noise level. The optimization problem in the presence of noise then becomes

$$\begin{aligned} \min_{X} \quad & ||X||_* \\ \text{subject to} \quad & ||\hat{y} - A(X)||_2 \leq \delta \end{aligned} \tag{4}$$

Where δ is linearly propertional to the noise level.

The relationship between the relative error and the noiseless matches the expectation. With noise level equal to the energy of the signal, signal recovery becomes impossible as the relative error approaches 1. From 10 to 40 dB, the error drops down linearly, as the guaranteed in [AAR14]. After the noise level becomes sufficiently insignificant, the relative error stays roughly constant. The little fluctuation from 50 to 80 dB is also likely caused by the same factor presented in the last section.

4.3 Blind Deconvolution with Non-Diffuse Subspace Matrix

In this experiment, the diffuse property of matrix $\boldsymbol{B}$ is deliberately violated. First, $\boldsymbol{B}$ is constructed the same way as the previous experiments. Then, its columns are scaled with a set of linear coefficients so that its coherence increases dramatically. The coefficient values are $2^{-1}, 2^{-2}, ..., 2^{-(k-2)}, 2^{-(k-1)}, 2^{-(k-1)}$. With this, $\boldsymbol{B}$ still has orthogonal columns and the total energy of $\hat{\boldsymbol{B}}$ does not change.

Monte Carlo simulation with $L = 64$ and $(N, K) \in [2, 12]$ and 5 trials per setting is conducted. For each trial, a set of matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ are generated at random, as well as a set of vectors $\boldsymbol{h}$ and $\boldsymbol{m}$. Also, the order of the scale coefficients for $\boldsymbol{B}$ are re-shuffled in each trial.

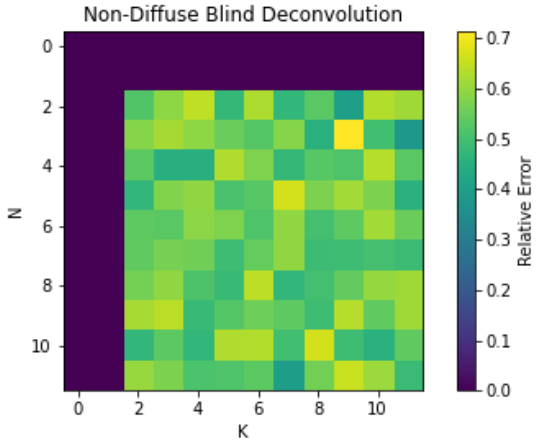

Figure 3: Phase Transition Plot When Diffuse Property is Violated

The result of this experiment matches the expectation. The performance degrades significantly compared to the diffuse case, and the noise effect on the plot that was significant in the first experiment fades away due to the large inflation of error.

4.4 Non Blind Deconvolution

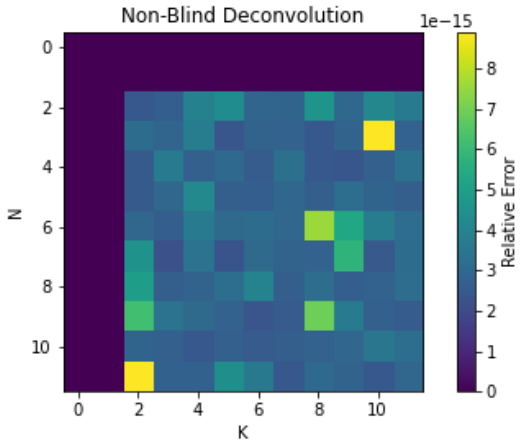

Figure 4: Phase Transition Plot of Non-Blind Deconvolution

In this experiment, we assume that $\boldsymbol{w}$ is known, therefore the only signal to be recovered is $\boldsymbol{h}$. In this scenario, circular convolution can be expressed as the product of a circulant matrix of a known vector and a unknown vector. Therefore, we can use the pseudo inverse of the circulant matrix to recover the unknown vector directly.

$$x = Circ(w)^{\dagger}\, y$$

Monte Carlo simulation with $L = 64$ and $(N, K) \in [2, 12]$ and 20 trials per setting is conducted. For each trial, a set of matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ are generated at random, as well as a set of vectors $\boldsymbol{h}$ and $\boldsymbol{m}$.

The computation time is hugely reduced, to a little more than one second. The result is also uniformly accurate, with errors on the magnitude of 10^{-15}. Since the recovery problem is reduced to computing an affine function, there is no surprise that the computation would be much faster. Also, the recovery

will be exact when the circular matrix is invertible, and low error level confirms that it is indeed the most common case.

4.5 Non-Convex Blind Deconvolution

[SL19] is referenced as the non-convex counterpart of blind deconvolution. This paper addresses the weakness on computation time of [AAR14], by proposing a gradient descent method that is reliable in terms of finding the global minima.

This paper also assumes that the subspaces of the vectors $\boldsymbol{w}$ and $\boldsymbol{x}$ are already known. The algorithm is composed of two parts. The first part is initializing the values of $\boldsymbol{h}$ and $\boldsymbol{m}$, so that the initial value falls in a 'basin of attraction' that contains the global minima. The second part is the conducting gradient descent from the initial value, which will lead to the convergence to the global minima.

Unfortunately, my implementation of the algorithm was not correct, and the gradient descent could not converge at all. The code is included in the submission as a proof of effort.

The success rate of recovery in [SL19] is significantly better than that of [AAR14]. Both display a sharp boundary around $L = 2.7(K + N)$, however, the tolerance value for error used in [SL19] is 0.01 instead of 0.02 in [AAR14].

References

[AAR14] B. Recht A. Ahmed and J. Romberg. Blind deconvolution using convex programming. *IEEE Transactions on Information Theory*, 60(3):1711–1732, 2014.

[SL19] T. Strohmer S. Ling. Regularized gradient descent: a non-convex recipe for fast joint blind deconvolution and demixing. *Information and Inference: A Journal of the IMA*, 8:1–49, 2019.