ECE273 Report: Blind Deconvolution

Guoren Zhong
gzhong@eng.ucsd.edu

***Abstract*—We study the problem of recovering two unknown signals w and x of length L from their convolution $y = w * x$, known as the blind deconvolution problem. Generally speaking, this is an ill-posed, non-linear and non-convex problem where the signals reconstruction is challenging with only a known measurement y. However, with some reasonable assumptions and convex relaxations, we can transform blind deconvolution problem into a probabilistic convex programming (*NNM*) problem. Basically, we assume that w and x lie in two known subspace and equivalently convert this problem to finding two vectors h and m of dimensions N and K in these subspace respectively. Notice that the outer product wx^* is a rank-1 matrix, so we can transform this problem into a low-rank matrix recovery problem and convexify it to a nuclear norm minimization program. It can be shown that this approach is theoretically guaranteed and also robust to some violations of its conditions. In this report, multiple numerical experiments are carried out to measure the performance of this approach and also compare it with some other approaches.**

***Index Terms*—Blind deconvolution, convex optimization, nuclear norm minimization.**

I. Introduction

BLIND deconvolution becomes an increasingly popular topic nowadays, and is necessary to many applications like images deblurring and channel protection [1], where we want to reconstruct the original signals with given measurement of their convolution. This problem is intrinsically a non-convex problem, and recovered signals cannot be found without certain restrictions. Therefore, in this report, we apply the convex relaxation technique introduced by Ahmed [1] to transform it into a solvable convex problem with theoretical guarantees. Then, we will realize blind deconvolution of two 1D signals via convex programming and discuss the conditions for successful recoveries. Besides, we will compare the results with the non-blind deconvolution approach. Additionally, we will also try to implement one of the non-convex approaches to blind deconvolution studied by Li [2], and compare the their results.

A. Notations

In this report, the notations are based on the [1]: uppercase bold ($\boldsymbol{B}, \boldsymbol{C}$), lowercase bold ($\boldsymbol{x}, \boldsymbol{w}$), and not bold letters (N, K, L) stand for matrices, vectors, and scalars respectively. Besides, calligraphic letters such as $\mathcal{A}$ are used as linear operators. In addition, we use $\|\cdot\|_2, \|\cdot\|_*, \|\cdot\|_F$ as ℓ_2, nuclear, and Frobenius norms respectively.

B. Problem statement

In this blind deconvolution problem, there are two unknown length L signals $\boldsymbol{w}$ and $\boldsymbol{x}$ and the measurement $\boldsymbol{y}$ is their convolution:

$$\boldsymbol{y} = \boldsymbol{w} * \boldsymbol{x}. \tag{1}$$

In this work, we will try to recover these two signals from the measurement $\boldsymbol{y}$ by different approaches.

II. Methods and techniques

In this section, we will show the process of simplifying the blind deconvolution problem, and eventually convert it into a convex optimization problem, where theoretical guarantees are provided.

A. Uniqueness of the solution

In general, the solution to blind deconvolution is not unique. For a convolution process $\boldsymbol{y} = \boldsymbol{w} * \boldsymbol{x}$ knowing only the observation $\boldsymbol{y}$, we may have multiple sets of solutions for $\boldsymbol{w}$ and $\boldsymbol{x}$ due to some ambiguities. Firstly, an unconstrained blind deconvolution suffers from shift ambiguity, meaning that we can get the same result when $\boldsymbol{w}$ and $\boldsymbol{x}$ are circularly shifted by θ and $-\theta$ respectively:

$$\boldsymbol{y}[l] = \sum_{l'=1}^{L} \boldsymbol{w}[l']\boldsymbol{x}[l-l'+1] = \sum_{l'=1}^{L} \boldsymbol{w}[l'+\theta]\boldsymbol{x}[l-l'+1-\theta]. \tag{2}$$

In order to solve this, we have the following two constraints for $\boldsymbol{w}$ and $\boldsymbol{x}$:

1) *Subspace constraint*: $\boldsymbol{w}$ and $\boldsymbol{x}$ live in the subspaces, whose bases are provided by the columns of two known matrices $\boldsymbol{B}$ and $\boldsymbol{C}$:
$$\begin{aligned} \boldsymbol{w} &= \boldsymbol{B}\boldsymbol{h}, \quad \boldsymbol{h} \in \mathbb{R}^K \\ \boldsymbol{x} &= \boldsymbol{C}\boldsymbol{m}, \quad \boldsymbol{m} \in \mathbb{R}^N. \end{aligned} \tag{3}$$
This makes the original problem of reconstructing $\boldsymbol{w}$ and $\boldsymbol{x}$ equivalent to recover two vectors $\boldsymbol{h}$ and $\boldsymbol{m}$, which are living in two known subspaces of $\mathbb{R}^L$ respectively.
2) *Sparsity constraint*: The two signals w and x are sparse over these subspaces.

With the above constraints, the shift ambiguity is solved since the signal will not live in these subspaces or remain sparsity if shifted.

Secondly, scaling ambiguity can happen as $\boldsymbol{w}*\boldsymbol{x} = (\alpha\boldsymbol{w})*(\alpha^{-1}\boldsymbol{x}) = y$, where α is any nonzero scalar. To avoid this, we can simply set $\|\boldsymbol{h}\|_2 = \|\boldsymbol{m}\|_2 = 1$ without loss of generality.

B. Convex relaxation

Blind deconvolution is intrinsically a non-convex problem since there might exist many local minima. However, it can be simplified and relaxed as a convex problem. To begin with, the convolution can be expanded as:

$$\boldsymbol{y} = [\text{circ}(\boldsymbol{C}_1)\boldsymbol{B} \ \cdots \ \text{circ}(\boldsymbol{C}_N)\boldsymbol{B}] \begin{bmatrix} m(1)\boldsymbol{h} \\ \vdots \\ m(N)\boldsymbol{h} \end{bmatrix}, \tag{4}$$

where $\text{circ}(\boldsymbol{C}_1)$ corresponds to circular convolution of the nth column of matrix $\boldsymbol{C}$. Then, we define the L-point normalization discrete Fourier transformation (DFT) matrix as:

$$\boldsymbol{F}(\omega,\ell)=\frac{1}{\sqrt{L}}\mathrm{e}^{-\mathrm{j}2\pi(\omega-1)(\ell-1)/L},\quad 1\le\omega,\ell\le L. \tag{5}$$

Using this, we can transform this problem into Fourier domain by defining $\hat{\boldsymbol{C}}=\boldsymbol{F}\boldsymbol{C}$, $\hat{\boldsymbol{B}}=\boldsymbol{F}\boldsymbol{B}$ and $\text{circ}(\boldsymbol{C}_1)=\boldsymbol{F}^*\Delta_n\boldsymbol{F}$, where $\Delta_n=\text{diag}(\sqrt{L}\hat{\boldsymbol{C}})$. Therefore, (4) can be transformed into a linear function as:

$$\hat{\boldsymbol{y}}=\boldsymbol{F}\boldsymbol{y}=[\Delta_1\hat{\boldsymbol{B}}\ \cdots\ \Delta_N\hat{\boldsymbol{B}}]\begin{bmatrix}m(1)\boldsymbol{h}\\ \vdots\\ m(N)\boldsymbol{h}\end{bmatrix}=\mathcal{A}(\boldsymbol{X}_0), \tag{6}$$

with $\boldsymbol{X}_0=\boldsymbol{h}\boldsymbol{m}^*$ and $\mathcal{A}:\mathbb{C}^{K\times N}\to\mathbb{R}^L$. Therefore, the blind deconvolution problem now becomes a linear reverse problem where $\boldsymbol{X}_0$ can be calculated by known $\mathcal{A}$ and $\hat{\boldsymbol{y}}$. Eventually, by applying singular value decomposition (SVD) to matrix $\boldsymbol{X}_0=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$, the recovered signals are obtained by simply taking the first columns of $\boldsymbol{U}$ and $\boldsymbol{V}$.

However, this is a non-convex problem since the set of possible matrices $\boldsymbol{X}_0$ is non-convex. In order to convexify this problem, we need to apply the idea of nuclear norm. As we know, the goal of this problem is to find the correct $\boldsymbol{h}\in\mathbb{R}^K$ and $\boldsymbol{m}\in\mathbb{R}^N$ with given $\hat{\boldsymbol{y}}\in\mathbb{C}^L$, to which least-squares is a good choice:

$$\begin{aligned}&\min_{\boldsymbol{u},\boldsymbol{v}}\quad \|\boldsymbol{u}\|_2^2+\|\boldsymbol{v}\|_2^2\\ &\text{subject to }\hat{\boldsymbol{y}}(\ell)=\langle\hat{\boldsymbol{c}}_\ell,\boldsymbol{u}\rangle\langle\boldsymbol{v},\hat{\boldsymbol{b}}_\ell\rangle,\quad \ell=1,...,L.\end{aligned} \tag{7}$$

where $\hat{\boldsymbol{b}}_\ell$ and $\hat{\boldsymbol{c}}_\ell$ are the columns of $\hat{\boldsymbol{B}}^*$ and $\hat{\boldsymbol{C}}^*$. With a convex objective but a non-convex constraint, this is still a non-convex problem. By taking the dual of this problem twice, we finally reach a convex optimization problem:

$$\begin{aligned}&\min_{\boldsymbol{X}}\quad \|\boldsymbol{X}\|_*\\ &\text{subject to }\hat{\boldsymbol{y}}=\mathcal{A}(\boldsymbol{X}).\end{aligned} \tag{8}$$

Till now we have converted the original non-convex problem into a convex problem by the "dual-dual" relaxation. By optimize the $\boldsymbol{X}$ matrix, we can recover the original $\boldsymbol{h}$ and $\boldsymbol{m}$.

C. Theoretical guarantees

The convex relaxation mentioned above is not deterministic, where some conditions need to be fulfilled in order to generate the true solution. Basically, we need large subspace dimensions K and N. Besides, a $\boldsymbol{B}$ matrix incoherent in the Fourier domain, and a generic matrix $\boldsymbol{C}$ are also required. To make a clear statement, we claim the followings:

1) *Incoherence parameters*: without loss of generality, we assume $\boldsymbol{B}^*\boldsymbol{B}=\hat{\boldsymbol{B}}^*\hat{\boldsymbol{B}}=\mathbb{I}_K$. Then, we define the incoherence parameters as:

$$\mu_{max}^2=\frac{L}{K}\max_{1\le\ell\le L}\|\hat{\boldsymbol{b}}_\ell\|_2^2,\quad \mu_{min}^2=\frac{L}{K}\min_{1\le\ell\le L}\|\hat{\boldsymbol{b}}_\ell\|_2^2 \tag{9}$$

Intrinsically, we we will always have $0\le\mu_{min}^2\le 1$ and $\mu_{min}^2\le\mu_{max}^2$.

2) *Diffusion parameter*: another parameter μ_h^2, named the diffusion parameter, is defined to measure the diffusion of the signal in the Fourier domain:

$$\mu_h^2=L\max_{1\le\ell\le L}|\hat{\boldsymbol{w}}(\ell)|^2=L\max_{1\le\ell\le L}|\langle\boldsymbol{h},\hat{\boldsymbol{b}}_\ell\rangle|^2. \tag{10}$$

From Cauchy-Schwartz inequality, and assuming $\|\boldsymbol{h}\|_2=1$, we have:

$$\mu_h^2\le L\max_{1\le\ell\le L}\|\hat{\boldsymbol{b}}_\ell\|_2^2\|\boldsymbol{h}\|_2^2\le K\mu_{max}^2. \tag{11}$$

Besides, from (9) we can also obtain the following:

$$\sum_{\ell=1}^{L}\mu_h^2=L\sum_{\ell=1}^{L}\max_{1\le\ell\le L}|\langle\boldsymbol{h},\hat{\boldsymbol{b}}_\ell\rangle|^2.$$

Equivalently,

$$\mu_h^2\ge\sum_{\ell=1}^{L}\max_{1\le\ell\le L}|\langle\boldsymbol{h},\hat{\boldsymbol{b}}_\ell\rangle|^2=\|\boldsymbol{h}\|_2^2=1. \tag{12}$$

Therefore, we conclude from (11) and (12) that:

$$1\le\mu_h^2\le\mu_{max}^2. \tag{13}$$

3) *Generic* $\boldsymbol{C}$: c) We take the entries of $\boldsymbol{C}$ to be normal distributed random numbers as $C[\ell,n]\sim\mathcal{N}(0,L^{-1})$, and the rows $\hat{\boldsymbol{c}}_\ell$ of the corresponding $\hat{\boldsymbol{C}}$ are independent.

With the above claims, the theoretical guarantee of (8) can be set up: Take two random normalized vectors $\boldsymbol{h}\in\mathbb{R}^K$ and $\boldsymbol{m}\in\mathbb{R}^N$ and let $\boldsymbol{B},\boldsymbol{C},\mu_{max}^2,\mu_h^2$ satisfy the claims above. Fix $\alpha\ge 1$, then $\boldsymbol{X}_0=\boldsymbol{h}\boldsymbol{m}^*$ is the unique solution (8) with probability of $1-\mathcal{O}(L^{-\alpha+1})$ if

$$\max\left(\mu_{max}^2K,\mu_h^2N\right)\le\frac{L}{C_\alpha\log^3 L}, \tag{14}$$

where $C_\alpha=\mathcal{O}(\alpha)$ depending only on α. With this, we can recover the original signals by $\boldsymbol{w}=\boldsymbol{B}\boldsymbol{h}$ and $\boldsymbol{x}=\boldsymbol{C}\boldsymbol{m}$.

III. Simulation results

In this section, we present the results of simulations on this convex nuclear norm minimization (*NNM*) algorithm. Then, we will compare it with the results of non-blind deconvolution, and also with those of one of the nonconvex approaches to blind deconvolution.

A. Convex programming

We first investigate the necessary conditions for the algorithm to correctly recover the input signals. In this simulation, L is fixed and multiple experiments are run for different value of K and N, with two random vectors $\boldsymbol{w}$ and $\boldsymbol{x}$ as inputs. Besides, $\boldsymbol{h}\in\mathbb{R}^K$ and $\boldsymbol{m}\in\mathbb{R}^N$ are chosen to be Gaussian random vectors. For each experiment, we obtain two recovered signals $\boldsymbol{u}$ and $\boldsymbol{v}$ corresponding to $\boldsymbol{h}$ and $\boldsymbol{m}$, respectively, and compute the error as:

$$Error=\frac{\|\hat{\boldsymbol{X}}-\boldsymbol{h}\boldsymbol{m}^*\|_F}{\|\boldsymbol{h}\boldsymbol{m}^*\|_F},\quad \hat{\boldsymbol{X}}=\boldsymbol{u}\boldsymbol{v}^*. \tag{15}$$

We do the experiments for both "sparse" and "short" input $\boldsymbol{w}$, where the matrix $\boldsymbol{B}$ is formed by randomly selected K

columns and the first K columns of an $L \times L$ identity matrix respectively. Specifically, we fix $L = 256$ and vary N and K from 5 to 200 with a step size of 5. The results for both cases are shown in Fig.1, where the recovery is considered successful if $Error < 0.02$.

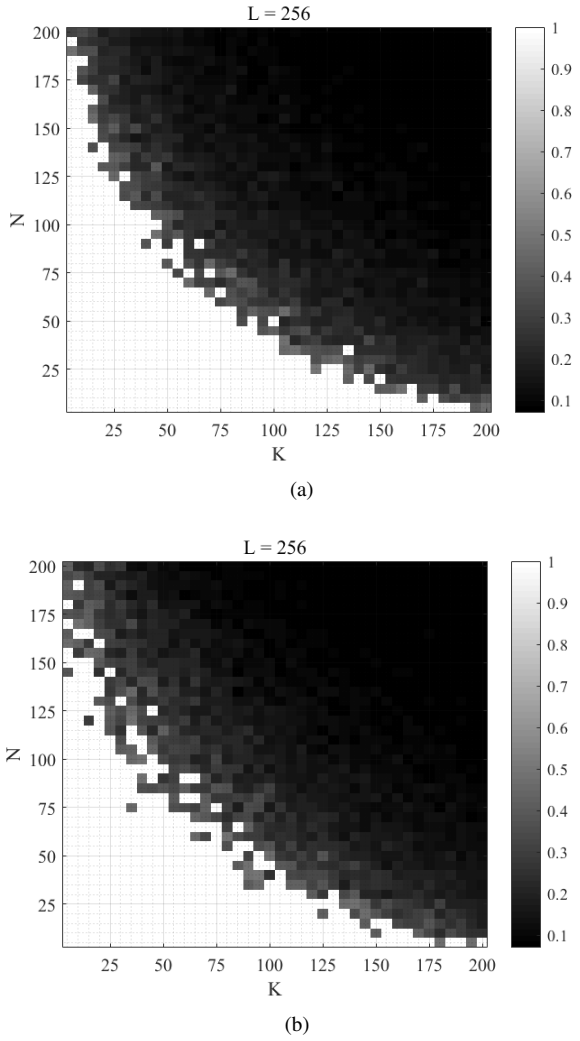


Fig. 1. Success rate for the deconvolution of two unknown signals $\boldsymbol{w}$ and $\boldsymbol{x}$. In part (a), $\boldsymbol{w}$ is "sparse" where nonzero entries are randomly distributed. In part (b), $\boldsymbol{w}$ is "short" where the first K terms are nonzero.

In both cases, we observe a sharp region around which the probability changes from close to 0 to close to 1. Besides, for both "sparse" and "short" input, a successful recovery is guaranteed with high probability when $L \gtrsim 2(K+N)$. This result is not exactly the same as what is given in [1], which may be caused by the smaller K, N and L values we choose in this simulation comparing to those in the paper. But in general, we still obtain a similar pattern in the blind deconvolution.

B. Comparison to non-blind deconvolution

Now, we consider the case of non-blind deconvolution, where one of the constituent signals (noise) is given. In this case, suppose $\boldsymbol{x}$ is known, then the problem of recovery of $\boldsymbol{w}$ can be expressed as the following problem:

$$\arg\min_{\boldsymbol{w}} \quad \|\boldsymbol{x} * \boldsymbol{w} - \boldsymbol{y}\|_2. \tag{16}$$

Thus, theoretically speaking, with one known signal, we would be able to find the other input signal directly using (4) as

$$\begin{bmatrix} m(1)\boldsymbol{h} \\ \vdots \\ m(N)\boldsymbol{h} \end{bmatrix} = [\text{circ}(\boldsymbol{C}_1)\boldsymbol{B} \ \cdots \ \text{circ}(\boldsymbol{C}_N)\boldsymbol{B}]^{\dagger}\boldsymbol{y}. \tag{17}$$

With this, we set up a test for comparison of performance between this non-blind deconvolution and the *NNM* blind deconvolution algorithm. For both cases, we set $K = N = 10$ and let L vary from $K+N$ to $4(K+N)$ in 16 equal steps, and the results are shown in Fig.2.

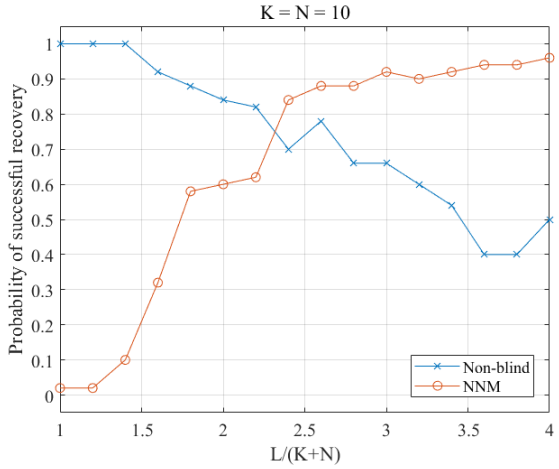


Fig. 2. Comparison of blind deconvolution and non-blind deconvolution.

From this result, we found that when measurement $\boldsymbol{y}$ is small, the non-blind deconvolution has a nearly perfect performance, while *NNM* doesn't seem to work well. However, when the dimension of $\boldsymbol{y}$ increases, *NNM* becomes better while the performance of non-blind deconvolution is going down. One of the possible cause of this decrease in accuracy may be the existence of local minima. In other words, since we are using the *fmincon* function to solve (4), the solver may be stuck in some local minima when $\boldsymbol{y}$ has a larger dimension.

C. A nonconvex approach to blind deconvolution

Here, we introduce a nonconvex approach to blind deconvolution called *regGrad*, which was first proposed in [2]. Briefly speaking, the *regGrad* algorithm consists of three steps:

1) *Approximation:* An operator $\mathcal{A} : \mathbb{C}^{K\times N} \to \mathbb{C}^L$ and its adjoint operator $\mathcal{A}^* : \mathbb{C}^L \to \mathbb{C}^{K\times N}$ are defined as

$$\mathcal{A}(\boldsymbol{Z}) = \{\hat{\boldsymbol{b}}_\ell^* \boldsymbol{Z} \hat{\boldsymbol{c}}_\ell\}_{\ell=1}^L, \quad \mathcal{A}^*(\boldsymbol{z}) = \sum_{\ell=1}^{L} z_\ell \hat{\boldsymbol{b}}_\ell \hat{\boldsymbol{c}}_\ell^*. \tag{18}$$

where $\hat{\boldsymbol{b}}_\ell$ and $\hat{\boldsymbol{c}}_\ell$ are the columns of $\hat{\boldsymbol{B}}^*$ and $\hat{\boldsymbol{C}}^*$. Then, $\boldsymbol{h}\boldsymbol{m}^*$ can be approximated by $\mathcal{A}^*(\hat{\boldsymbol{y}})$ since:

$$\mathbb{E}[\mathcal{A}^*(\hat{\boldsymbol{y}})] = \mathbb{E}\left[\mathcal{A}^*(\mathcal{A}(\boldsymbol{h}\boldsymbol{m}^*))\right] = \boldsymbol{h}\boldsymbol{m}^* \tag{19}$$

where $\boldsymbol{h}$ and $\boldsymbol{m}$ are the ground-truth inputs, and $\hat{\boldsymbol{y}}$ is the observed convolution of $\boldsymbol{h}$ and $\boldsymbol{m}$ in the Fourier domain.

2) *Initialization:* With the approximation above, we can compute the initial values for the gradient descent by

finding the leading singular value, left and right singular vectors of $\mathcal{A}^*(\hat{\boldsymbol{y}})$, denoted by d, $\hat{\boldsymbol{h}}_0$ and $\hat{\boldsymbol{m}}_0$ respectively. Then, our initial values are set as:

$$\boldsymbol{u}_0 = \sqrt{d}\hat{\boldsymbol{h}}_0, \quad \boldsymbol{v}_0 = \sqrt{d}\hat{\boldsymbol{m}}_0 \tag{20}$$

3) *Gradient descent:* The gradients of the cost function F and the regularization term G with respect to $\boldsymbol{h}$ and $\boldsymbol{m}$ can be computed as:

$$F_{\boldsymbol{h}} = \mathcal{A}^*(\mathcal{A}(\boldsymbol{h}\boldsymbol{m}^*) - \boldsymbol{y})\boldsymbol{x},$$
$$F_{\boldsymbol{m}} = [\mathcal{A}^*(\mathcal{A}(\boldsymbol{h}\boldsymbol{m}^*) - \boldsymbol{y})]^*\boldsymbol{h},$$
$$G_{\boldsymbol{h}} = \frac{\rho}{2d}\left[G_0\left(\frac{\|\boldsymbol{h}\|_2^2}{2d}\right)\boldsymbol{h} + \frac{L}{4\mu^2}G_0\left(\frac{L|\hat{\boldsymbol{b}}_\ell^*\boldsymbol{h}|^2}{8d\mu^2}\right)\hat{\boldsymbol{b}}_\ell\hat{\boldsymbol{b}}_\ell^*\boldsymbol{h}\right],$$
$$G_{\boldsymbol{m}} = \frac{\rho}{2d}G_0\left(\frac{\|\boldsymbol{m}\|_2^2}{2d}\right)\boldsymbol{m},$$

where $G_0(z) = 2\sqrt{\max\{z-1,0\}^2}$. Thus, with the initial values obtained above, we can do gradient descent iteratively for $t = 1, 2, ...$:

$$\begin{aligned}\boldsymbol{u}_t &= \boldsymbol{u}_{t-1} - \eta\nabla\tilde{F}_{\boldsymbol{h}}(\boldsymbol{u}_{t-1}, \boldsymbol{v}_{t-1}) \\ \boldsymbol{v}_t &= \boldsymbol{v}_{t-1} - \eta\nabla\tilde{F}_{\boldsymbol{m}}(\boldsymbol{u}_{t-1}, \boldsymbol{v}_{t-1})\end{aligned} \tag{21}$$

where

$$\tilde{F}_{\boldsymbol{h}} = F_{\boldsymbol{h}} + G_{\boldsymbol{h}}, \quad \tilde{F}_{\boldsymbol{m}} = F_{\boldsymbol{m}} + G_{\boldsymbol{m}}$$

D. Comparison of convex and nonconvex approach

In this part, we make a comparison of the convex (*NNM*) algorithm and the nonconvex *regGrad* algorithm mentioned above. In order to be fair in the comparison, both algorithms take "short" inputs, with $K = N = 10$ fixed, and L changing from $K + N$ to $4(K + N)$ in 16 equal steps. Besides, for the *regGrad*, we set the parameters as

$$\rho = \frac{d^2}{100}, \quad \mu = \frac{6\sqrt{L/(K+N)}}{\log L}, \quad \eta = \frac{1}{N\log L + \rho L/\mu^2}.$$

The results are shown in Fig.3, where we again consider successful recovery if $Error < 0.02$. Generally in these simulations, we can conclude that the *regGrad* algorithm has a higher successful rate than the *NNM* algorithm. In other words, *regGrad* requires a smaller number of measurements than *NNM* in order to achieve successful recovery of high probability. However, each simulation of *NNM* takes about 13 iterations to give the final result, while for *regGrad*, it needs 2000 iterations. Therefore, behind the better performance of *regGrad*, it also costs higher, for both computation and time.

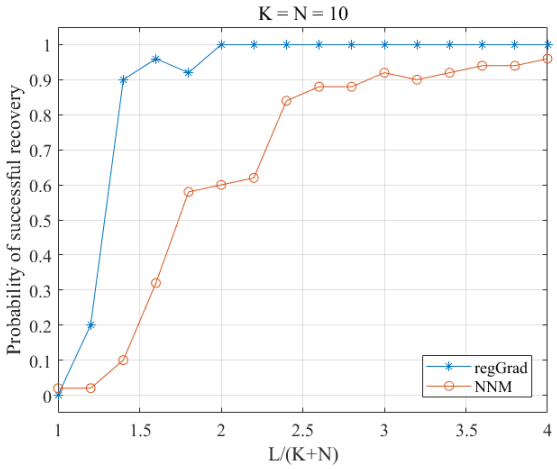


Fig. 3. Performance comparison of the nonconvex *regGrad* algorithm and the convex *NNM* algorithm. For the vertical axis, we do 50 random tests for each value of $L/(K+N)$ for both algorithms, and calculate their probability of successful recovery.

IV. Discussion of Robustness

We also explore the robustness of this algorithm against some violations of its conditions, such as sparsity and low-rank.

A. Robustness against violations of sparsity

First, we test the robustness against the violation of sparsity. Here, we set up an experiment where we fixed $K = N = 10$ and let L vary from $K + N$ to $4(K + N)$ in 16 equal steps again. Then, three simulations are run under these conditions, which are "dense $\boldsymbol{B}$ and $\boldsymbol{C}$", "dense $\boldsymbol{B}$ and sparse $\boldsymbol{C}$", and "sparse $\boldsymbol{B}$ and $\boldsymbol{C}$". For the "dense" matrices, we randomly choose numbers from a Gaussian distribution $\mathcal{N}(0, 1)$, and the probabilities of successful recovery are shown in Fig.4.

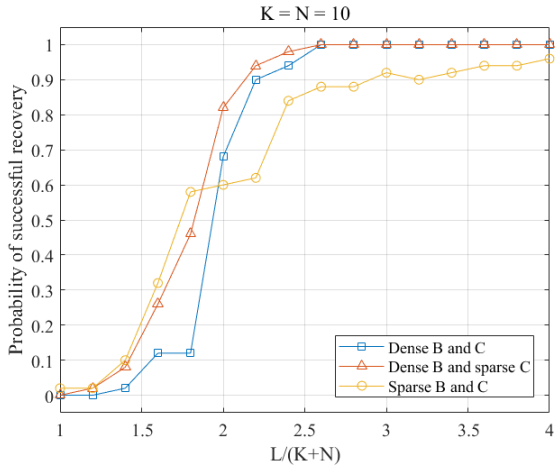
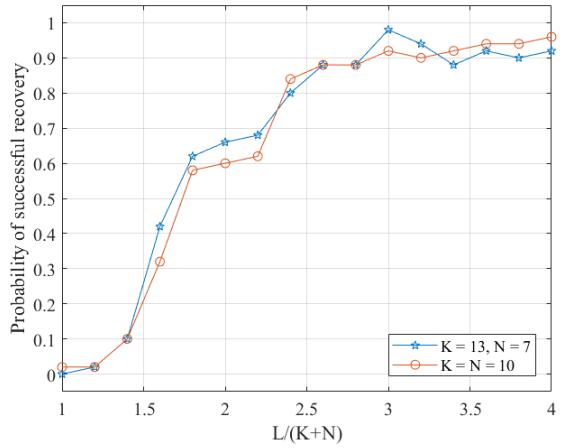


Fig. 4. Robustness against violations of sparsity condition. 50 random tests are done for each case of each condition.

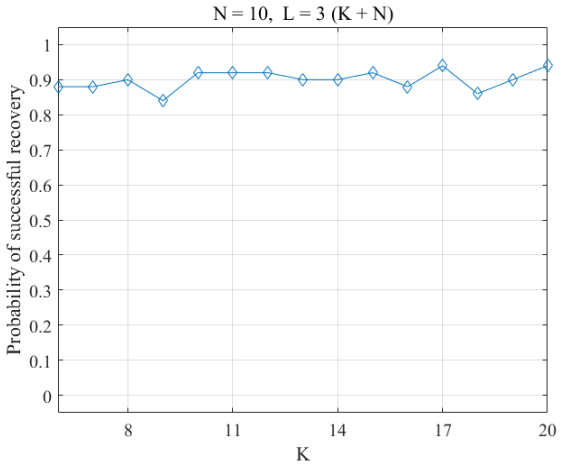
From Fig.4, we can conclude that *NNM* algorithm is robust against the violation of sparsity condition, where the probabilities of successful recovery do not show significant influence by the violation. However, one thing that is worth mentioning is that the success rates of dense cases are even higher than that of sparse case when $L > 2(K+N)$. One possible reason for this is the value of K and N here is relatively small, which may cause inconsistency in simulations. Besides, when checking the results of the sparse case, we found that there exists one particular random state, for which the recovery fails for nearly all values of L. Therefore, randomness may be another reason for this phenomenon.

B. Robustness against violations of low-rank

Besides, we also consider the robustness against the violation of the low-rank condition. For the *NNM* algorithm, we claim that the input signal $\boldsymbol{w}$ has a lower dimension than the noise $\boldsymbol{x}$, i.e. $K \leq N$. Here, we design two numerical simulations for testing the robustness of the algorithm against the violation of low-rank condition. For the first simulation, we fixed the sum of K and N, and also the range of L, from $K+N$ to $4(K+N)$. Under these conditions, we run two tests for $K = N = 10$ and $K = 13, N = 7$ respectively. For the second simulation, we fixed the value of $N = 10$, and also fixed the relation between L and K, N as $L = 3(K + N)$. Then, we run tests for K varying from 6 to 20 in 15 equal steps. The results for both simulations are shown in Fig.5.



(a)



(b)

Fig. 5. Robustness against violations of low-rank condition. In part (a), two sets of different values for K and N are applied, with L having the same range. In part (b), $N = 10$ is fixed, and K varies from 6 to 20 in 15 equal steps. In both parts, the vertical axes are the probability of successful recovery based on 50 random tests for each case.

As is shown in Fig.5, *NNM* algorithm is robust against the violation of low-rank condition, where the probability of successful recovery is not affected due to this violation.

V. Conclusion

In conclusion, we implement the *NNM* blind deconvolution algorithm and compare it with other approaches. In the comparison with non-blind deconvolution, we found that *NNM* tends to do better when the number of measurements increase, while the non-blind deconvolution approach has a decreasing accuracy that may be caused by local minima. In the comparison with the nonconvex *regGrad* algorithm, *NNM* requires a larger number measurements than *regGrad* in order to give accurate recovery. However, *NNM* has a lower computational and time cost. Finally, we verify the robustness of the *NNM* algorithm against the violations of some conditions, such as sparsity and low-rank. Yet, the simulations in this report are all done with relatively low dimensions (small K, N and L) due to the technical restriction, so the results may vary a little when applying larger values for K, N and L.

References

[1] A. Ahmed, B. Recht and J. Romberg, "Blind Deconvolution Using Convex Programming," in IEEE Transactions on Information Theory, vol. 60, no. 3, pp. 1711-1732, March 2014

[2] X. Li, S. Ling, T. Strohmer and K. Wei. (2016). Rapid, Robust, and Reliable Blind Deconvolution via Nonconvex Optimization. Applied and Computational Harmonic Analysis.

[3] Y. Li, K. Lee and Y. Bresler, "Identifiability and Stability in Blind Deconvolution Under Minimal Assumptions," in IEEE Transactions on Information Theory, vol. 63, no. 7, pp. 4619-4633, July 2017.

[4] A. Levin, Y. Weiss, F. Durand and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 1964-1971.

[5] L. Wang and Y. Chi, "Blind Deconvolution From Multiple Sparse Inputs," in IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1384-1388, Oct. 2016,.

[6] Y. Li and Y. Bresler. 2018. Global geometry of multichannel sparse blind deconvolution on the sphere. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18). Curran Associates Inc., Red Hook, NY, USA, 1140–1151.