LOW RANK PHASE RETRIEVAL

Seyedehsara Nayer, Namrata Vaswani*, Yonina C. Eldar***

*Iowa State University, Ames, IA, {sarana,namrata}@iastate.edu, **Technion, Haifa, Israel

ABSTRACT

We study the problem of recovering a low-rank matrix, $\boldsymbol{X}$, from phaseless measurements of random linear projections of its columns. We develop a novel solution approach, called AltMinTrunc, that consists of a two-step truncated spectral initialization step, followed by a three-step alternating minimization algorithm. We obtain sample complexity bounds for the AltMinTrunc initialization to provide a good approximation of the true $\boldsymbol{X}$. When the rank of $\boldsymbol{X}$ is low enough, these are significantly smaller than what existing single vector phase retrieval algorithms need. Via extensive experiments, we demonstrate the same for the entire algorithm.

***Index Terms*—** Phase retrieval, non-convex methods

1. INTRODUCTION

In the last several years there has been a large amount of work on the phase retrieval (PR) problem and on its generalization. The original PR problem involves recovering an n length signal $\boldsymbol{x}$ from the magnitudes of its discrete Fourier transform (DFT) coefficients. Generalized PR replaces the DFT by inner products with any set of measurement vectors, $\boldsymbol{a}_i$. Thus, the goal is to recover $\boldsymbol{x}$ from $|\boldsymbol{a}_i{}'\boldsymbol{x}|^2$, $i = 1, 2, \dots, m$. These magnitude-only measurements are referred to as *phaseless measurements*. PR is a classical problem that occurs in many applications such as optics, X-ray crystallography and astronomy because the phase information is either difficult or impossible to obtain [1]. Algorithms for solving it have existed since the work of Gerchberg and Saxton and of Fineup [2, 3]. In recent years, there has been much renewed interest in PR [4, 5, 6, 7, 8, 9, 1]. The sparse PR problem has also been studied, e.g., see [10, 11, 12].

Early provably correct approaches [4, 5] to PR involved a 'lifting solution': instead of recovering $\boldsymbol{x}$, these recover the rank-one matrix $(\boldsymbol{x}\boldsymbol{x}')$ from $\boldsymbol{y}_i := \text{trace}(\boldsymbol{a}_i\boldsymbol{a}_i{}'(\boldsymbol{x}\boldsymbol{x}'))$ by solving a convex optimization problem; followed by estimating $\boldsymbol{x}$ as the top eigenvector of the recovered matrix. These recover $\boldsymbol{x}$ (upto a global phase ambiguity) with only $m = cn$ independent identically distributed (iid) Gaussian phaseless measurements; however because of the 'lifting', the computational and storage complexity is high - it depends on n^2 instead of on n. In more recent work, non-convex methods, that do not lift the problem to higher dimensions, have been explored along with provable guarantees - AltMinPhase [7], Wirtinger Flow (WF) [8] and truncated WF (TWF) [9]. TWF had the best sample complexity and speed. It needs only cn iid Gaussian phaseless measurements. *Here, and throughout, c is reused to denote different numerical constants.*

Problem Definition. We consider phase retrieval of columns of low-rank matrices in the following setting. Instead of a single vector $\boldsymbol{x} \in \mathbb{R}^n$, we have a set of q vectors, $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_q$ that are such that the $n \times q$ matrix, $\boldsymbol{X} := [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_q]$, has rank $r \ll \min(n, q)$. For each $\boldsymbol{x}_k$, we observe a set of m measurements of the form

$$\boldsymbol{y}_{i,k} := |\boldsymbol{a}_{i,k}{}'\boldsymbol{x}_k|^2, \; i = 1, 2, \dots m, \; k = 1, 2, \dots, q. \qquad (1)$$

Our goal is to recover the matrix $\boldsymbol{X}$ from these mq measurements. Since we have magnitude-only measurements of each column $\boldsymbol{x}_k$, we can only hope to recover each column $\boldsymbol{x}_k$ up to a *global phase ambiguity*. A motivating example for this problem is a dynamic astronomical imaging application such as solar imaging where the sun's surface properties gradually change over time. Each image arranged as a 1D vector forms one column $\boldsymbol{x}_k$ of $\boldsymbol{X}$. The changes are usually due to a much smaller number of factors, r, than the size of the image, n, or the total number of images, q. Because of this, the matrix $\boldsymbol{X}$ is exactly (or approximately) low rank.

Contributions. We develop a novel algorithm, called AltMinTrunc, that is partly inspired by AltMinPhase and TWF, to solve the above problem. AltMinTrunc relies on the fact that a rank r matrix $\boldsymbol{X}$ can always be expressed (non-uniquely) as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$ where $\boldsymbol{U}$ is an $n \times r$ matrix with mutually orthonormal columns. It consists of a truncated spectral initialization step for first initializing $\boldsymbol{U}$, and then, $\boldsymbol{B}$, followed by an alternating minimization algorithm. Extensive simulation and real video experiments demonstrate its advantage over existing work.

We also obtain sample complexity bounds for the AltMinTrunc initialization to get within an ε ball of the true $\boldsymbol{X}$. As seen in many earlier works, e.g., [7] or resampled WF [8, Algorithm 2], for a fixed error level, the sample complexity of the entire algorithm is governed by that of the initialization step and hence our results are important. We show that, if the goal is to just recover $\text{range}(\boldsymbol{U})$ with subspace error below a fixed error level, then a total of $mq = cnr^2$ iid Gaussian measurements suffice with high probability (whp). When r is small, nr^2 is only slightly larger than nr which is the minimum required to recover $\boldsymbol{U}$. If the goal is to also recover the $\boldsymbol{x}_k$'s, then we need more measurements, but still significantly fewer than cn measurements per column. For example, if $q = c\sqrt{n}$ and $r = c\log n$, then we need only $c\sqrt{n}(\log n)^7$ measurements per column. If $q = cn^2$, then this reduces to $c\sqrt{n}$. To our knowledge, our results provide the first set of guarantees for low-rank matrix recovery from column-wise phaseless measurements.

In [13], we introduced another algorithm called PreLow to solve the above problem and evaluated it using simulation experiments. The initialization step of AltMinTrunc is essentially the same as that of PreLow and hence our guarantees also apply to PreLow's initialization. The rest of PreLow involves an alternating truncated gradient descent over $\boldsymbol{U}$ and $\boldsymbol{B}$ that requires many parameters to be carefully set. We compare with PreLow in Sec. 3.

In other somewhat related work [14, 15], the authors study the problem of recovering a rank r matrix $\boldsymbol{X}$ from measurements of the form $\boldsymbol{y}_i = \text{trace}(\boldsymbol{A}_i{}'\boldsymbol{X})$. In the special case when $\boldsymbol{A}_i = \boldsymbol{a}_i\boldsymbol{a}_i'$, our problem may *get wrongly confused* with theirs. However, *the two problems are, in fact, completely different* because we have access to phaseless measurements of *each column* $\boldsymbol{x}_k$ of the low-rank matrix $\boldsymbol{X}$; moreover, the measurements of each $\boldsymbol{x}_k$ are obtained using a *different* set of measurement vectors $\boldsymbol{a}_{i,k}$ for each column. It should be mentioned that the latter is critical for the improved sample com-

A longer version of this work is under submission to IEEE Transactions on Signal Processing.

978-1-5090-4117-6/17/$31.00 ©2017 IEEE

plexity of AltMinTrunc over single-vector PR methods because this is what allows it to exploit averaging over all the m measurements of all the q vectors $\boldsymbol{x}_k$ while estimating $\boldsymbol{U}$.

2. ALTMINTRUNC: ALTERNATING MINIMIZATION WITH TRUNCATED SPECTRAL INITIALIZATION

We use the following metrics in our algorithm design and analysis.

1. For two matrices $\hat{\boldsymbol{U}}$ and $\boldsymbol{U}$ with mutually orthonormal columns, $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) := \|(\boldsymbol{I} - \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}')\boldsymbol{U}\|$ quantifies the *subspace error* between their range spaces.
2. The *phase-invariant distance* between two vectors is quantified using [8, 9] $\mathrm{dist}(\boldsymbol{z}_1, \boldsymbol{z}_2) := \min_{\phi\in[0,2\pi]} \|\boldsymbol{z}_1 - e^{\sqrt{-1}\phi}\boldsymbol{z}_2\|$. When $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are both real, the phase is only $+1$ or -1. Thus, in this case, $\mathrm{dist}(\boldsymbol{z}_1, \boldsymbol{z}_2) = \min(\|\boldsymbol{z}_1 - \boldsymbol{z}_2\|, \|\boldsymbol{z}_1 + \boldsymbol{z}_2\|) = \|\boldsymbol{z}_1 - \mathrm{sign}({\boldsymbol{z}_1}'\boldsymbol{z}_2)\boldsymbol{z}_2\|$ is a sign-invariant measure of distance.

Here, and throughout, $\|\cdot\|$ denotes the l_2 norm of a vector or the induced l_2 norm of a matrix. For other l_p norms, we use $\|\cdot\|_p$.

AltMinTrunc consists of two steps, initialization and alternating minimization (alt-min), both of which use the fact that a rank r matrix $\boldsymbol{X}$ can be expressed as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$ where $\boldsymbol{U}$ is an $n \times r$ matrix with mutually orthonormal columns and $\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \dots \boldsymbol{b}_q]$ is an $r \times q$ matrix. Of course, the decomposition is not unique since we can always rewrite $\boldsymbol{X}$ as $\boldsymbol{X} = (\boldsymbol{U}\boldsymbol{R})(\boldsymbol{R}'\boldsymbol{B})$ where $\boldsymbol{R}$ is a rotation matrix. The initialization step of AltMinTrunc first computes an estimate of $\mathrm{range}(\boldsymbol{U})$, i.e., it returns $\hat{\boldsymbol{U}}$ that may be very different from $\boldsymbol{U}$ in Frobenius norm, but their spans are close, i.e., $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U})$ is small. The next step is to initialize the $\boldsymbol{b}_k$'s. Since the goal is to only recover $\boldsymbol{x}_k$'s accurately, we find estimates $\hat{\boldsymbol{b}}_k$ so that $\mathrm{dist}(\hat{\boldsymbol{U}}\hat{\boldsymbol{b}}_k, \boldsymbol{x}_k)$ is small. This is done efficiently by actually estimating $\boldsymbol{g}_k := \hat{\boldsymbol{U}}'\boldsymbol{x}_k$ for each k, and setting $\hat{\boldsymbol{b}}_k = \hat{\boldsymbol{g}}_k$. The rest of the algorithm consists of alternating minimization over three sets of variables: the missing phase of $({\boldsymbol{a}_{i,k}}'\boldsymbol{x}_k)$, $\boldsymbol{U}$, and the $\boldsymbol{b}_k$'s. We explain both steps below.

Initialization. Let $\frac{1}{q}\sum_{k=1}^{q} \boldsymbol{x}_k{\boldsymbol{x}_k}' \overset{\mathrm{EVD}}{=} \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}'$ denote the reduced eigenvalue decomposition (EVD) of $\boldsymbol{X}\boldsymbol{X}'/q$. Define

$$\boldsymbol{Y}_{U,0} := \frac{1}{mq}\sum_{i=1}^{m}\sum_{k=1}^{q} \boldsymbol{y}_{i,k}\boldsymbol{a}_{i,k}{\boldsymbol{a}_{i,k}}'.$$

It is not hard to see that [8, Lemma A.1],

$$\mathbb{E}[\boldsymbol{y}_{i,k}\boldsymbol{a}_{i,k}{\boldsymbol{a}_{i,k}}'] = 2\boldsymbol{x}_k{\boldsymbol{x}_k}' + \|\boldsymbol{x}_k\|^2\boldsymbol{I}$$

and hence

$$\mathbb{E}[\boldsymbol{Y}_{U,0}] = 2\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}' + \mathrm{trace}(\boldsymbol{\Lambda})\boldsymbol{I}.$$

Clearly, the subspace spanned by the top r eigenvectors of this matrix is equal to $\mathrm{range}(\boldsymbol{U})$ and the gap between its r-th and $(r+1)$-th eigenvalue is $2\lambda_{\min}(\boldsymbol{\Lambda})$. If m and q are large enough, one can use an appropriate law of large numbers result to argue that $\boldsymbol{Y}_{U,0}$ will be close to its expected value whp. By the $\sin\theta$ theorem [16], as long as $2\lambda_{\min}(\boldsymbol{\Lambda})$ is large compared to the difference between $\boldsymbol{Y}_{U,0}$ and its expectation, one can argue that the same will also be true for the span of the top r eigenvectors of $\boldsymbol{Y}_{U,0}$.

However, as explained in [9], because $\boldsymbol{y}_{i,k}\boldsymbol{a}_{i,k}{\boldsymbol{a}_{i,k}}'$ can be written out as $\boldsymbol{w}\boldsymbol{w}'$ where $\boldsymbol{w}$ is a heavy-tailed random vector, more samples will be needed for law of large numbers to take effect than if $\boldsymbol{w}$ were not heavy-tailed. One approach to convert this into a matrix of the form $\boldsymbol{w}\boldsymbol{w}'$ where $\boldsymbol{w}$ is sub-Gaussian (and hence not heavy-tailed) is to use the truncation idea suggested in [9]. Using this, we compute $\hat{\boldsymbol{U}}$ as the top r eigenvectors of

$$\boldsymbol{Y}_U := \frac{1}{mq}\sum_i\sum_k \boldsymbol{y}_{i,k}\boldsymbol{a}_{i,k}{\boldsymbol{a}_{i,k}}'\mathbb{1}_{\{\boldsymbol{y}_{i,k}\le 9\frac{\sum_i \boldsymbol{y}_{i,k}}{m}\}}. \tag{2}$$

Here $\mathbb{1}_{\xi}$ is the indicator function for the statement ξ. The idea of truncation is to average over only those (i,k)'s for which $\boldsymbol{y}_{i,k}$ is not too far from its empirical mean.

To understand how to initialize the $\boldsymbol{b}_k$'s, consider the matrix

$$\boldsymbol{Y}_{b,k} := \hat{\boldsymbol{U}}'\boldsymbol{M}_k\hat{\boldsymbol{U}} \text{ where } \boldsymbol{M}_k := \frac{1}{m}\sum_i \boldsymbol{y}_{i,k}\boldsymbol{a}_{i,k}{\boldsymbol{a}_{i,k}}'. \tag{3}$$

Suppose that $\hat{\boldsymbol{U}}$ is independent of $\boldsymbol{M}_k$. Then, conditioned on $\hat{\boldsymbol{U}}$, the expectation of the above matrix is

$$\hat{\boldsymbol{U}}'(2\boldsymbol{x}_k{\boldsymbol{x}_k}' + \|\boldsymbol{x}_k\|^2\boldsymbol{I})\hat{\boldsymbol{U}} = 2\boldsymbol{g}_k{\boldsymbol{g}_k}' + \|\boldsymbol{x}_k\|^2\boldsymbol{I}$$

Clearly, the top eigenvector of this expectation is proportional to $\boldsymbol{g}_k$ and the gap between its first and second eigenvalues is $2\|\boldsymbol{g}_k\|^2 = 2\|\hat{\boldsymbol{U}}'\boldsymbol{U}\boldsymbol{b}_k\|^2$. Thus, as long as $\mathrm{range}(\hat{\boldsymbol{U}})$ is a good estimate of $\mathrm{range}(\boldsymbol{U})$, the eigen-gap will be close to 2. Let $\boldsymbol{g}_k = \boldsymbol{v}_k\nu_k$ where $\nu_k = \|\boldsymbol{g}_k\|$. So, once again, if m is large enough, one can argue that the top eigenvector of $\hat{\boldsymbol{U}}'\boldsymbol{M}_k\hat{\boldsymbol{U}}$, denoted $\hat{\boldsymbol{v}}_k$, will be a good estimate of $\boldsymbol{v}_k$. Using this idea, we initialize the $\boldsymbol{x}_k$'s as $\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{U}}\hat{\boldsymbol{v}}_k\hat{\nu}_k$ where $\hat{\nu}_k = \sqrt{\sum_i \boldsymbol{y}_{i,k}/m}$ is an estimate of ν_k. We do not use truncation here because $\boldsymbol{g}_k$ is an r length vector, with $r \ll n$, and we anyway need to use many more than r measurements for accurate recovery.

The complete initialization algorithm is summarized in Algorithm 1. Here we use the same set measurements to recover both $\boldsymbol{U}$ and $\boldsymbol{b}_k$'s and hence the independence required above is not ensured, but the algorithm still works in practice.

Algorithm 1 AltMinTrunc-init: initialization step of AltMinTrunc

```
1. Compute $\hat{\boldsymbol{U}}$ as top $r$ eigenvectors of $\boldsymbol{Y}_U$ defined in (2).
2. For each $k = 1, 2, \dots, q$,
    (a) compute $\hat{\boldsymbol{v}}_k$ as the top eigenvector of $\boldsymbol{Y}_{b,k}$ defined in (3).
    (b) compute $\hat{\nu}_k := \sqrt{\frac{1}{m}\sum_i \boldsymbol{y}_{i,k}}$
    (c) set $\hat{\boldsymbol{b}}_k = \hat{\boldsymbol{g}}_k = \hat{\boldsymbol{v}}_k\hat{\nu}_k$ and $\hat{\boldsymbol{x}}_k := \hat{\boldsymbol{U}}\hat{\boldsymbol{g}}_k$.
Output $\hat{\boldsymbol{U}}$ and $\hat{\boldsymbol{x}}_k$'s for all $k = 1, 2, \dots, q$.
```

Alternating Minimization. The rest of the algorithm is an intuitive modification of the original Gerchberg-Saxton algorithm [2, 3], or the later AltMinPhase algorithm [7], to the current problem. Let $\boldsymbol{y}_k := [\boldsymbol{y}_{1,k}, \boldsymbol{y}_{2,k}, \dots, \boldsymbol{y}_{m,k}]'$ and $\boldsymbol{A}_k := [\boldsymbol{a}_{1,k}, \boldsymbol{a}_{2,k}, \dots, \boldsymbol{a}_{m,k}]$. Then $\sqrt{\boldsymbol{y}_k} = |{\boldsymbol{A}_k}'\boldsymbol{x}_k|$. Suppose that the phase information were available, i.e., suppose that we had access to a diagonal matrix $\boldsymbol{C}_k$ so that $\boldsymbol{C}_k\sqrt{\boldsymbol{y}_k} = {\boldsymbol{A}_k}'\boldsymbol{x}_k$. Then recovering $\boldsymbol{X}$ from these linear measurements would be an example of a low-rank matrix recovery problem. This itself can be solved by minimizing over $\boldsymbol{U}$ and $\boldsymbol{B}$ alternatively, see, e.g., [17] and references therein. With $\boldsymbol{B}$ fixed, this is a least squares (LS) recovery problem for $\boldsymbol{U}$ and vice versa. With estimates of $\boldsymbol{U}$ and $\boldsymbol{B}$, we can estimate the phase matrix $\boldsymbol{C}_k$ as $\hat{\boldsymbol{C}}_k = \mathrm{diag}(\mathrm{phase}({\boldsymbol{A}_k}'\hat{\boldsymbol{U}}\hat{\boldsymbol{b}}_k))$. AltMinTrunc, summarized in Algorithm 2, alternates between these three steps. We show the power of AltMinTrunc over TWF in Fig. 1.

3. NUMERICAL EXPERIMENTS

We compare AltMinTrunc (AMT) with TWF [9] and with PreLow [13]. TWF does not use the low rank property of the data. To obtain a TWF-based benchmark method that does use this property, we developed and evaluated TWF-proj. At the end of the TWF initialization for all $\boldsymbol{x}_k$'s and at the end of each TWF iteration, TWF-proj

Algorithm 2 Complete AltMinTrunc algorithm

```
- Let $\hat{U}$ and $\hat{b}_k$ denote the output of the initialization step (Algorithm 1).
- For $t = 1$ to $T$, repeat the following three steps:
    1. for all $k = 1, 2, \dots, q$, $\hat{C}_k \leftarrow \text{diag}(\text{phase}(\hat{U}\hat{b}_k))$
    2. $\hat{U} \leftarrow \arg\min_{\tilde{U}} \sum_k \|\hat{C}_k \sqrt{y_k} - A_k{}'\tilde{U}\hat{b}_k\|^2$
    3. for all $k = 1, 2, \dots, q$,
       $\hat{b}_k \leftarrow \arg\min_{\tilde{b}_k} \|\hat{C}_k \sqrt{y_k} - A_k{}'\hat{U}\tilde{b}_k\|^2$
- Output $\hat{U}$ and $\hat{x}_k = \hat{U}\hat{b}_k$ for all $k = 1, 2, \dots, q$.
```

The LS step of steps 2 and 3 can be solved in closed form as follows.

- Step 2: let $\hat{U}_{vec}$ be columnwise vectorized version of $\hat{U}$; compute $\hat{U}_{vec} = (M'M)^{-1}M'y_c$ where $y_c = [\hat{C}_1\sqrt{y_1}; \hat{C}_2\sqrt{y_2}; \dots; \hat{C}_q\sqrt{y_q}]$, $M = [M_1; M_2; \dots M_q]$ and $M_k = [A_k{}'(\hat{b}_k)_1, A_k{}'(\hat{b}_k)_2, \dots, A_k{}'(\hat{b}_k)_r]$; reshape it to get $\hat{U}$. Here ; means we concatenate the M_k's or the $\hat{C}_k\sqrt{y_k}$'s column-wise.
- Step 3: $\hat{b}_k = (M'M)^{-1}M'\hat{C}_k\sqrt{y_k}$ where $M = A_k{}'\hat{U}$.

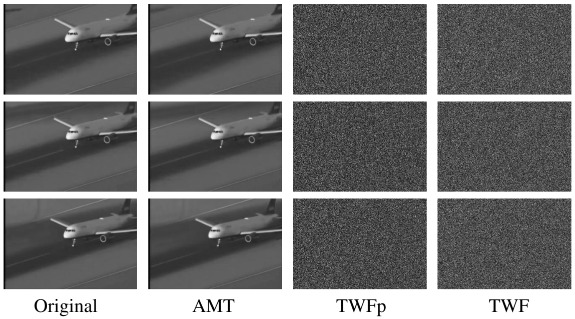

Fig. 1: First column: frame number 63, 33 and 3, of the original plane video (not low-rankified). Next three columns: frames recovered using AltMinTrunc (AMT), TWF-proj (TWFp) and TWF from $m = 3n$ phaseless coded diffraction pattern (CDP) measurements.

projects the current estimated X onto the space of rank r matrices. Lastly, we also developed and evaluated AMT2, which uses AMT-init (Algorithm 1) for initialization, but TWF-proj to replace the alt-min iterations.

Simulation experiments. U is generated by orthonormalizing an $n \times r$ matrix with iid Gaussian entries; $b_k \overset{\text{iid}}{\sim} \mathcal{N}(0, I)$; $x_k = Ub_k$; and $y_{i,k}$ satisfied (1) with $a_{i,k} \overset{\text{iid}}{\sim} \mathcal{N}(0, I)$. Recovery error is quantified using $\text{NormErr}(X, \hat{X})$ defined in Theorem 4.1. We report NormErr (NE) averaged over a 100 time Monte Carlo.

We first demonstrate the power of the proposed initialization, AMT-init, given in Algorithm 1. We compare it with TWF initialization (TWF-init) and TWF-proj-init. Here PreLow init is essentially the same as Algorithm 1 and hence it is not compared with. Data was generated with $n = 100$, $r = \log_{10} n = 2$ and varying q and m. We report the NormErr in Table 1. As can be seen, AMT-init significantly outperforms TWF-init and TWF-proj-init, e.g., when $q = 1000$, even with $m = \sqrt{n} = 10$, the NormErr of AMT-init is 0.46 while that of TWF is 1.62. The reason for this is that AMT first estimates U and then B; and $\hat{U}$ is computed as the top r eigenvectors of Y_U which exploits averaging of $y_{i,k}a_{i,k}a_{i,k}{}'$ over both i and k. TWF initializes each x_k as the top eigenvector of $y_{i,k}a_{i,k}a_{i,k}{}'$ averaged over i and thus the averaging over k is not exploited at all.

	$q = 100$			$q = 1000$			
m	AMT	TWF	TWFp	AMT	TWF	TWFp	AMT-Same
10	1.32	1.62	1.00	0.46	1.62	0.73	0.99
50	0.53	1.48	0.77	0.11	1.48	0.57	0.99
100	0.23	1.35	0.59	0.06	1.35	0.49	0.98

Table 1: Initialization error comparison. NormErr is displayed. AMT-Same: AMT applied to measurements $y_{i,k} := (a_i'x_k)^2$.

The first step of TWF-proj is TWF for each x_k and this, again, does not utilize averaging over k. For $q = 1000$, we also show errors of *AMT-Same*. This refers to AMT operating on measurements of the form $y_{i,k} := (a_i'x_k)^2$. Because it uses the same a_i's for all x_k's, it does not gain from the averaging over k. Hence its errors are large.

Next, we compare the complete AltMinTrunc (AMT) algorithm given in Algorithm 2 with TWF [9], TWF-proj (TWFp), AMT2 and PreLow [13]. TWF was implemented using the authors' code. TWFp was implemented by including the projection step after the initialization and after each iteration in the TWF code. PreLow was implemented exactly as stated in [13]. Data was generated using $n = 100$, $r = 2$ and varying q and m. All algorithms were run until either $\text{NormErr} \leq 0.0001$ or for a total of 300 iterations, whichever came first. We display the averaged NE and execution time in Table 2. The execution times are shown in parentheses. Clearly, AMT has the smallest error with AMT2 error being only a little larger. PreLow has the third largest error, followed by TWFp and then TWF.

Time complexity. Both the initialization and per iteration time complexity of AMT is at least r times that of TWF. However, since the AMT initialization error is much smaller, the total number of iterations needed by AMT to get to $\text{NormErr} < 0.0001$ is smaller than those needed by TWF and TWF-proj. This why the complete AMT algorithm is actually faster than TWF or TWFp. A similar argument applies for AMT2 as well. In fact, since its per iteration time complexity is governed by that of TWFp (which is faster per iteration than AMT), it is the fastest.

m	AMT	AMT2	TWF	TWFp	Prelow
			$q = 100$		
50	0.08 (6)	0.240 (3.2)	46 (7)	41 (9)	0.741 (1)
100	0.003 (2)	0.002 (0.54)	1 (9)	0.09 (4)	0.065 (1)
			$q = 1000$		
50	0.052 (39)	0.070 (40)	2e55 (48)	1e40 (54)	0.34 (7)
100	0.0009 (10)	0.003 (5)	40 (54)	0.14 (53)	0.003 (6)

Table 2: Results for simulated data, full algorithm: the table is displayed as NormErr (time in seconds).

Real videos and Coded Diffraction Pattern (CDP) measurements. We used real videos that are approximately low rank and CDP measurements of their images. Briefly, the CDP model consists of masked-Fourier measurements: one applies a different random mask (random diagonal matrix) to each x_k followed by DFT of the resultant vector. More details are explained in [18]. Each image (arranged as a 1D vector) corresponds to one x_k and hence the entire video corresponds to the matrix X. We show results on a moving mouse video and on the moving airplane video shown in Fig. 1. We show results both with "low-rankified videos" and with the original airplane video. The airplane images were of size $n_1 \times n_2$ with $n_1 = 240$, $n_2 = 320$; the mouse images had $n_1 = 180$, $n_2 = 319$. Thus, $n = n_1n_2 = 76800$ and $n = 57420$ respectively. Mouse video had $q = 90$ frames and airplane one had $q = 105$ frames. A detailed explanation of this experiment is given in [18].

We display the NormErr for AMT, AMT2, TWF and TWF-proj in Table 3. Execution times are again shown in parentheses. Three frames of the results corresponding to the last row of this table are shown in Fig. 1 in Sec. 2. As can be seen, AMT has the smallest er-

AMT	AMT2	TWF	TWFp
Mouse, Low-rankified video, $r = 15, m = 2n$			
8.0e-04 (18776)	0.07 (905)	2.2 (103)	13 (394)
Plane, Low-rankified video, $r = 6, m = 2n$			
7.8e-10 (1036)	6.9e-07 (574)	2.2 (137)	14 (327)
Plane, Original video, $r = 25, m = 3n$			
0.146 (13472)	0.150 (3451)	2.0 (207)	14 (950)

Table 3: Results for videos with CDP measurements: the table is displayed as NormErr (time in seconds).

ror in all cases. AMT is also the slowest. TWF and TWFp fail when m is small: notice that NormErr is much more than one even with $m = 3n$. AMT2 is slower than TWF and TWFp but is much faster than AMT. Its errors are only a little larger than AMT and hence it may offer the best compromise between speed and performance.

4. PERFORMANCE GUARANTEES

In this section, we show that we can get a provably accurate initial estimate of both $\boldsymbol{U}$ and of the $\boldsymbol{x}_k$'s whp when using iid Gaussian measurement vectors, $\boldsymbol{a}_{i,k}$. The proof consists of two parts. We first bound the subspace recovery error $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U})$. Next, we use this to bound the error in estimating the $\boldsymbol{x}_k$'s, $\mathrm{dist}(\hat{\boldsymbol{x}}_k, \boldsymbol{x}_k)$. To do this, we show that, if $\hat{\boldsymbol{U}}$ is a given matrix with $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) \le \varepsilon_U$, and if the measurement vectors that are used to estimate the $\boldsymbol{b}_k$'s are independent of $\hat{\boldsymbol{U}}$, then, whp, $\mathrm{dist}^2(\hat{\boldsymbol{x}}_k, \boldsymbol{x}_k) \le c\varepsilon$. To ensure that the independence assumption holds, we borrow a standard trick used in many earlier works, e.g., [7]. We partition the measurements into two disjoint sets of size m and $\tilde{m}$ respectively; we use the first set for estimating $\boldsymbol{U}$ and the second set for estimating the $\boldsymbol{b}_k$'s. Denote the first set of measurements and measurement vectors by $\boldsymbol{y}_{i,k}$ and $\boldsymbol{a}_{i,k}$ respectively. Denote the second set by $\boldsymbol{y}_{i,k}^{new}$ and $\boldsymbol{a}_i^{new}$ respectively. Since the $\boldsymbol{b}_k$'s are recovered independently, in the second set, we can use the same measurement vectors, $\boldsymbol{a}_i^{new}$, for all $\boldsymbol{x}_k$'s.

Let $\frac{1}{q}\sum_k \boldsymbol{x}_k \boldsymbol{x}_k{}' \overset{\mathrm{EVD}}{=} \boldsymbol{U}\bar{\boldsymbol{\Lambda}}\boldsymbol{U}'$ denote its reduced EVD. Since $\boldsymbol{x}_k = \boldsymbol{U}\boldsymbol{b}_k$, this implies that $\bar{\boldsymbol{\Lambda}} = \frac{1}{q}\sum_k \boldsymbol{b}_k \boldsymbol{b}_k{}'$. Let $\bar{\lambda}_{\max}$ denote its maximum eigenvalue and $\bar{\lambda}_{\min}$ its minimum eigenvalue. Define

$$\rho := \frac{\max_k \|\boldsymbol{b}_k\|_\infty}{\bar{\lambda}_{\max}} \text{ and } \kappa := \frac{\bar{\lambda}_{\max}}{\bar{\lambda}_{\min}}.$$

This definition of ρ implies that $\max_k \|\boldsymbol{b}_k\|^2 \le r\rho\bar{\lambda}_{\max}$. κ denotes the condition number of $\bar{\boldsymbol{\Lambda}}$ and hence also of $\boldsymbol{X}\boldsymbol{X}'$. With these definitions, we can state our results in terms of just κ and ρ.

Theorem 4.1. *For each $\boldsymbol{x}_k$, $k = 1, 2, \ldots, q$, we observe m phaseless measurements $\boldsymbol{y}_{i,k} := (\boldsymbol{a}_{i,k}{}'\boldsymbol{x}_k)^2$ with $\boldsymbol{a}_{i,k} \overset{\mathrm{iid}}{\sim} \mathcal{N}(0, \boldsymbol{I})$; and $\tilde{m}$ phaseless measurements $\boldsymbol{y}_{i,k}^{new} := (\boldsymbol{a}_i^{new}{}'\boldsymbol{x}_k)^2$ with $\boldsymbol{a}_i^{new} \overset{\mathrm{iid}}{\sim} \mathcal{N}(0, \boldsymbol{I})$, and with $\boldsymbol{a}_i^{new}$'s are independent of $\boldsymbol{a}_{i,k}$'s. Consider the output of Algorithm 1 with step 2 modified as follows: replace $\boldsymbol{y}_{i,k}$ and $\boldsymbol{a}_{i,k}$ by $\boldsymbol{y}_{i,k}^{new}$ and $\boldsymbol{a}_i^{new}$ respectively. Suppose that $r \le cn^{1/5}$. For an $\varepsilon < 1$, if*

$$\tilde{m} \ge \frac{c\sqrt{n}}{\varepsilon^2},\ m \ge \frac{c\kappa^2 \cdot r^4 \log n (\log \tilde{m})^2}{\varepsilon^2},$$
$$mq \ge \frac{c\rho^2\kappa^2 \cdot nr^4 (\log \tilde{m})^2}{\varepsilon^2},$$

then, with probability (w.p.) at least $1 - 2\exp(-cn) - \frac{16q}{n^4}$,

1. *$\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) := \|(\boldsymbol{I} - \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}')\boldsymbol{U}\| \le \frac{c\varepsilon}{r \log \tilde{m}}$;*
2. *for all $k = 1, 2, \ldots, q$, $\mathrm{dist}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)^2 \le c\varepsilon\|\boldsymbol{x}_k\|^2$, and so*

$$\mathrm{NormErr}(\boldsymbol{X}, \hat{\boldsymbol{X}}) := \frac{\sum_{k=1}^q \mathrm{dist}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)^2}{\sum_{k=1}^q \|\boldsymbol{x}_k\|^2} \le c\varepsilon.$$

Further, if $q \le cn^2$, then the above holds w.p. at least $1 - c/n^2$.

Proof: See [18].

Observe that the lower bounds on m and mq depend on κ^2. This is pretty typical, e.g., it is also the case in [17]. Note also that the probability of the good event depends inversely on q. This comes from needing to ensure that each of the q $\boldsymbol{x}_k$'s are accurately recovered. However, it is a weak dependence; it can be removed if $q \le cn^2$.

For the rest of our discussion, assume that ε, κ and ρ are fixed. Since Theorem 4.1 provides a bound for AltMinTrunc which exploits the low-rank property of $\boldsymbol{X}$, when r/q is small, its per column sample complexity is significantly smaller than what single vector PR methods need. For example, if $r = c\log n$ and $q = cr^4(\log n)^3 = (\log n)^7$, AltMinTrunc needs $\tilde{m} = c\sqrt{n}$ and $m = cn/\log n$ and hence a total of $cn/\log n$ measurements per column. For $q = cn^2$, this number reduces to just $c\sqrt{n}$. On the other hand, single vector PR methods need this number to be at least cn.

When the goal is to only recover $\boldsymbol{U}$ with subspace error at most ε, AltMinTrunc needs even fewer measurements.We have the following corollary.

Corollary 4.2. *In the setting of Theorem 4.1, if $\tilde{m} = 0$, $m \ge \frac{c\kappa^2 r^2 \log n}{\varepsilon^2}$ and $mq \ge \frac{c\rho^2\kappa^2 \cdot nr^2}{\varepsilon^2}$, then with probability at least $1 - 2\exp(-cn) - \frac{2q}{n^4}$, $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) \le c\varepsilon$.*

Recall that $\boldsymbol{U}$ is an $n \times r$ matrix. From Corollary 4.2, for a fixed ε, ρ, and κ, one needs a total of only $mq = cnr^2$ measurements to recover $\boldsymbol{U}$. When r is small, e.g., $r = c\log n$, this is only slightly more than the minimum required which would be nr.

To recover the $\boldsymbol{b}_k$'s, notice from Theorem 4.1 that we need an extra set of $\tilde{m} \ge c\sqrt{n}$ measurements and an extra factor of $(r \log \tilde{m})^2$ in the lower bounds on m and mq. A few points should be mentioned. First, $\tilde{m} \ge c\sqrt{n}$ can be replaced by $\tilde{m} \ge cn^{1/5}$, or in fact $cn^{1/d}$ for any integer $d > 2$, and our result will not change, except for numerical constants. Second, we could also completely replace this lower bound by $\tilde{m} \ge cr\log^4 r$ which is much nicer, but then Theorem 4.1 will hold with probability only $1 - \frac{8q}{\tilde{m}^8} - 2\exp(-cn) - \frac{8q}{n^4}$. Third, if $\tilde{m}$ is equal to its current lower bound of $c\sqrt{n}$, and r is small, e.g., if $r = c\log n$, the extra factor of $(r\log\tilde{m})^2$ is only $c(\log n)^4$. The reason that we need this extra factor is because our algorithm recovers $\boldsymbol{g}_k := \hat{\boldsymbol{U}}'\boldsymbol{U}\boldsymbol{b}_k$ and sets $\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{U}}\hat{\boldsymbol{g}}_k$. Thus, for it to give an accurate enough estimate of $\boldsymbol{x}_k$, we need to ensure that $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U})$ is very small so that $\|\hat{\boldsymbol{U}}'\boldsymbol{U}\|$ is close to one. In particular we need $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) \le \varepsilon/r\log\tilde{m}$. Ensuring this requires a larger lower bound on mq and m than just ensuring $\mathrm{SE}(\hat{\boldsymbol{U}}, \boldsymbol{U}) \le \varepsilon$.

Probabilistic model on $\boldsymbol{x}_k$'s. The results given above treat the $\boldsymbol{x}_k$'s as deterministic unknowns and hence give a high probability result for one unknown matrix $\boldsymbol{X}$. We can also assume a probabilistic model on the $\boldsymbol{x}_k$'s and obtain high probability results over a class of matrices $\boldsymbol{X}$. This has been done in our long version [18].

5. CONCLUSIONS AND FUTURE WORK

We presented a novel phase retrieval (PR) algorithm, AltMinTrunc (AMT), for recovering a set of q unknown vectors, $\boldsymbol{x}_k$, lying in a low (r) dimensional subspace of $\mathbb{R}^n$ from their phaseless measurements. We obtained sample complexity bounds for the initialization step of AMT; and argued that, when r/q is small, these are much smaller than those for TWF or any other single-vector PR method.

The first step in future work will be to analyze the entire AMT algorithm. Another goal will be develop a practical version of AMT that can estimate the rank, r, automatically from the measurements, and that uses AMT whenever $\hat{r}/q$ is small, but uses TWF otherwise.

6. REFERENCES

[1] Y. Shechtman, Y. C. Eldar, O. Cohen, H. N. Chapman, J. Miao, and M. Segev, "Phase Retrieval with Application to Optical Imaging," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 87–109, 2015.

[2] R. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik*, 1972.

[3] J. R. Fienup et al, "Phase retrieval algorithms: a comparison," *Applied Optics*, pp. 2758–2769, 1982.

[4] E. J. Candes, Y. C Eldar, T. Strohmer, and V. Voroninski, "Phase retrieval via matrix completion," *SIAM J. Imaging Sci.*, vol. 6, no. 1, pp. 199–225, 2013.

[5] E. J. Candes, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Comm. Pure Appl. Math.*, 2013.

[6] Y. C. Eldar and S. Mendelson, "Phase retrieval: Stability and recovery guarantees," *Appl. Comput. Harmon. Anal.*, vol. 36, no. 3, pp. 473–94, May 2014.

[7] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2013, pp. 2796–2804.

[8] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval via wirtinger flow: Theory and algorithms," *IEEE Trans. Info. Th.*, vol. 61, no. 4, pp. 1985–2007, 2015.

[9] Y. Chen and E. Candes, "Solving random quadratic systems of equations is nearly as easy as solving linear systems," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2015, pp. 739–747.

[10] K. Jaganathan, S. Oymak, and B. Hassibi, "Recovery of sparse 1-d signals from the magnitudes of their fourier transform," in *IEEE Intl. Symp. on Information Theory (ISIT)*. IEEE, 2012, pp. 1473–1477.

[11] Y. Shechtman, A. Beck, and Y. C. Eldar, "Gespar: Efficient phase retrieval of sparse signals," *IEEE Trans. Sig. Proc.*, vol. 62, no. 4, pp. 928–938, 2014.

[12] A. Szameit, Y. Shechtman, E. Osherovich, E. Bullkich, P. Sidorenko, H. Dana, S. Steiner, E. B. Kley, S. Gazit, T. Cohen-Hyams, S. Shoham, M. Zibulevsky, I. Yavneh, Y. C. Eldar, O. Cohen, and M. Segev, "Sparsity-based single-shot subwavelength coherent diffractive imaging," *Nature Materials*, vol. 11, pp. 455–9, Apr. 2012.

[13] S. Nayer, N. Vaswani, and Y. Eldar, "Low rank matrix recovery from column-wise phaseless measurements," in *IEEE Statistical Signal Processing Workshop*, 2016.

[14] S. Tu, R. Boczar, M. Soltanolkotabi, and B. Recht, "Low-rank solutions of linear matrix equations via procrustes flow," *arXiv preprint arXiv:1507.03566*, 2015.

[15] Q. Zheng and J. Lafferty, "A convergent gradient descent algorithm for rank minimization and semidefinite programming from random linear measurements," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2015.

[16] C. Davis and W. M. Kahan, "The rotation of eigenvectors by a perturbation. iii," *SIAM J. Numer. Anal.*, vol. 7, pp. 1–46, Mar. 1970.

[17] P. Netrapalli, P. Jain, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Symposium on Theory of Computing (STOC)*, 2013.

[18] N. Vaswani, S. Nayer, and Y. Eldar, "Low rank phase retrieval," *submitted to IEEE Trans. Sig. Proc., also at arXiV:1608.04141*, 2016.