Solving Random Quadratic Systems of Equations Is Nearly as Easy as Solving Linear Systems

Authored by:

Emmanuel Candes
Yuxin Chen

Abstract

This paper is concerned with finding a solution x to a quadratic system of equations y_i = —¡ a_i, x ¿—2, i = 1, 2, ..., m. We prove that it is possible to solve unstructured quadratic systems in n variables exactly from O(n) equations in linear time, that is, in time proportional to reading and evaluating the data. This is accomplished by a novel procedure, which starting from an initial guess given by a spectral initialization procedure, attempts to minimize a non-convex objective. The proposed algorithm distinguishes from prior approaches by regularizing the initialization and descent procedures in an adaptive fashion, which discard terms bearing too much influence on the initial estimate or search directions. These careful selection rules—which effectively serve as a variance reduction scheme—provide a tighter initial guess, more robust descent directions, and thus enhanced practical performance. Further, this procedure also achieves a near-optimal statistical accuracy in the presence of noise. Finally, we demonstrate empirically that the computational cost of our algorithm is about four times that of solving a least-squares problem of the same size.

1 Paper Body

Suppose we are given a response vector y = [yi]1?i?m generated from a quadratic transformation of an unknown object x ? Rn /Cn , i.e. 2

yi = —hai , xi— ,
i = 1, ? ? ? , m,
(1)

where the feature/design vectors ai ? Rn /Cn are known. In other words, we acquire measurements about the linear product hai , xi with all signs/phases missing. Can we hope to recover x from this nonlinear system of equations? This problem can be recast as a quadratically constrained quadratic program (QCQP), which subsumes as special cases various classical combinatorial problems with Boolean variables (e.g. the NP-complete stone problem [1, Section

3.4.1]). In the physical sciences, this problem is commonly referred to as phase retrieval [2]; the origin is that in many imaging applications (e.g. X-ray crystallography, diffraction imaging, microscopy) it is infeasible to record the phases of the diffraction patterns so that we can only record —Ax—2 , where x is the electrical field of interest. Moreover, this problem finds applications in estimating the mixture of linear regression, since one can transform the latent membership variables into missing phases [3]. Despite its importance across various fields, solving the quadratic system (1) is combinatorial in nature and, in general, NP complete. To be more realistic albeit more challenging, the acquired samples are almost always corrupted by some amount of noise, namely, 2

yi ? —hai , xi— ,

i = 1, ? ? ? , m. 1

(2)

For instance, in imaging applications the data are best modeled by Poisson random variables ind. 2 yi ? Poisson —hai , xi— , i = 1, ? ? ? , m,

(3)

which captures the variation in the number of photons detected by a sensor. While we shall pay special attention to the Poisson noise model due to its practical relevance, the current work aims to accommodate general?or even deterministic?noise structures. 1.1

Nonconvex optimization

Assuming independent samples, the first attempt is to seek the maximum likelihood estimate (MLE): Xm minimizez ? ‘ (z; yi) , (4) i=1

where ‘ (z; yi) represents the log-likelihood of a candidate z given the outcome yi . As an example, under the Poisson data model (3), one has (up to some constant offset) ‘(z; yi) = yi log(—a?i z—2) ? —a?i z—2 . Computing the MLE, however, is in general intractable, since ‘(z; yi) is not concave in z.

(5)

Fortunately, under unstructured random systems, the problem is not as ill-posed as it might seem, and is solvable via convenient convex programs with optimal statistical guarantees [4?12]. The basic paradigm is to lift the quadratically constrained problem into a linearly constrained problem by introducing a matrix variable X = xx? and relaxing the rank-one constraint. Nevertheless, working with the auxiliary matrix variable significantly increases the computational complexity, which exceeds the order of n3 and is prohibitively expensive for large-scale data. This paper follows a different route, which attempts recovery by minimizing the nonconvex objective (4) or (5) directly (e.g. [2, 13?19]). The main incentive is the potential computational benefit, since this strategy operates directly upon vectors instead of lifting decision variables to higher dimension. Among this class of procedures, one natural candidate is the family of gradient-descent type algorithms developed with respect to the objective (4). This paradigm can be regarded as performing some variant of stochastic gradient descent over the random samples {(yi , ai)}1?i?m as an approximation to maximize the population likelihood L(z) := E(y,a) [‘(z; y)]. While in general nonconvex optimization falls short of performance guarantees, a recently proposed approach called Wirtinger Flow (WF) [13] promises efficiency under

random features. In a nutshell, WF initializes the iterate via a spectral method, and then successively refines the estimate via the following update rule: ?t Xm z (t+1) = z (t) + ¿(z (t) ; yi), i=1 m where z (t) denotes the tth iterate of the algorithm, and ?t is the learning rate. Here, ¿(z; yi) represents the Wirtinger derivative with respect to z, which reduces to the ordinary gradient in the real setting. Under Gaussian designs, WF (i) allows exact recovery from O (n log n) noise-free quadratic equations [13];1 (ii) recovers x up to -accuracy within O(mn2 log 1/) time (or flops) [13]; and (iii) is stable and converges to the MLE under Gaussian noise [20]. Despite these intriguing guarantees, the computational complexity of WF still far exceeds the best that one can hope for. Moreover, its sample complexity is a logarithmic factor away from the information-theoretic limit. 1.2

This paper: Truncated Wirtinger Flow

This paper develops a novel linear-time algorithm, called Truncated Wirtinger Flow (TWF), that achieves a near-optimal statistical accuracy. The distinguishing features include a careful initialization procedure and a more adaptive gradient flow. Informally, TWF entails two stages: 1. Initialization: compute an initial guess z (0) by means of a spectral method applied to a subset T0 of data {yi } that do not bear too much influence on the spectral estimates; 2. Loop: for 0 ? t ¡ T , ?t X ¿(z (t) ; yi) (6) z (t+1) = z (t) + i?Tt+1 m for some index set Tt+1 ? {1, ? ? ? , m} over which ¿(z (t) ; yi) are well-controlled. 1 f (n) = O (g(n)) or f (n) . g(n) (resp. f (n) & g(n)) means there exists a constant c ¿ 0 such that —f (n)— ? c—g(n)— (resp. —f (n)— ? c —g(n)—). f (n) g(n) means f (n) and g(n) are orderwise equivalent.

2
100
-20 -25
Relative MSE (dB)
Relative error
10-1
truncated WF
10-2 10-3 10-4
least squares (CG)
10-5 10-6
truncated WF
-35 -40 -45 -50
MLE w/ phase
-55 -60
10-7
10
-30
-65
-8
0 0
5 20
Iteration

15 60
10 40
15
20
25
30
35
40
45
50
55
SNR (dB) (n =100)

(a) (b) Figure 1: (a) Relative errors of CG and TWF vs. iteration count, where n = 1000 and m = 8n. (b) Relative MSE vs. SNR in dB, where n = 100. The curves are shown for two settings: TWF for solving quadratic equations (blue), and MLE had we observed additional phase information (green). We highlight three aspects of the proposed algorithm, with details deferred to Section 2. (a) In contrast to WF and other gradient descent variants, we regularize both the initialization and the gradient flow in a more cautious manner by operating only upon some iteration-varying index sets Tt . The main point is that enforcing such careful selection rules lead to tighter initialization and more robust descent directions. (b) TWF sets the learning rate ?t in a far more liberal fashion (e.g. ?t ? 0.2 under suitable conditions), as opposed to the situation in WF that recommends ?t = O(1/n). (c) Computationally, each iterative step mainly consists in calculating {¿(z; yi)}, which is inexpensive and can often be performed in linear time, that is, in time proportional to evaluating the data and the constraints. Take the real-valued Poisson likelihood (5) for example:

2 yi ? —a¿ yi i z— ¿ ¿ a a z ? a a z = 2 ai , 1 ? i ? m, ¿(z; yi) = 2 i i 2 i i —a¿ a¿ i z— i z which essentially amounts to two matrix-vector products. To see this, rewrite (y ?—a¿ z (t) —2 X 2 i a¿iz(t) , i ? Tt+1 , ¿ (t) ¿(z ; yi) = A v, vi = i 0, otherwise, i?Tt+1 where A := [a1 , ? ? ? , am]¿ . Hence, Az (t) gives v and A¿ v the desired truncated gradient. 1.3

Numerical surprises

The power of TWF is best illustrated by numerical examples. Since x and e?j? x are indistinguishable given y, we evaluate the solution based on a metric that disregards the global phase [13]: dist (z, x) := min?:?[0,2?) ke?j? z ? xk.

(7)

In the sequel, TWF operates according to the Poisson log-likelihood (5), and takes ?t ? 0.2. We first compare the computational efficiency of TWF for solving quadratic systems with that of conjugate gradient (CG) for solving least square problems. As is well known, CG is among the most popular methods for solving large-scale least square problems, and hence offers a desired benchmark. We run TWF and CG respectively over the following two problems: (a)

find x ? Rn

(b)

n

find x ? R
s.t. bi = a¿ i x, s.t. bi = ind.
—a¿ i x—,
1 ? i ? m, 1 ? i ? m,

where m = 8n, x ? N (0, I), and ai ? N (0, I). This yields a well-conditioned design matrix A, for which CG converges extremely fast [21]. The relative estimation errors of both methods are reported in Fig. 1(a), where TWF is seeded by 10 power iterations. The iteration counts are plotted in different scales so that 4 TWF iterations are tantamount to 1 CG iteration. Since each iteration of CG and TWF involves two matrix vector products Az and A¿ v, the numerical plots lead to a suprisingly positive observation for such an unstructured design: 3

Figure 2: Recovery after (top) truncated spectral initialization, and (bottom) 50 TWF iterations. Even when all phase information is missing, TWF is capable of solving a quadratic system of equations only about 4 times2 slower than solving a least squares problem of the same size! The numerical surprise extends to noisy quadratic systems. Under the Poisson data model, Fig. 1(b) displays the relative mean-square error (MSE) of TWF when the signal-to-noise ratio (SNR) varies; here, the relative MSE and the SNR are defined as3 MSE := dist2 (? x, x) / kxk2

and

SNR := 3kxk2 ,

(8)

? is an estimate. Both SNR and MSE are displayed on a dB scale (i.e. the values of where x 10 log10 (SNR) and 10 log10 (MSE) are plotted). To evaluate the quality of the TWF solution, we compare it with the MLE applied to an ideal problem where the phases (i.e. {?i = sign(a¿ i x)}) are revealed a priori. The presence of this precious side information gives away the phase retrieval problem and allows us to compute the MLE via convex programming. As illustrated in Fig. 1(b), TWF solves the quadratic system with nearly the best possible accuracy, since it only incurs an extra 1.5 dB loss compared to the ideal MLE with all true phases revealed. To demonstrate the scalability of TWF on real data, we apply TWF on a 320?1280 image. Consider a type of physically realizable measurements called coded diffraction patterns (CDP) [22], where y (l) = —F D (l) x—2 ,

1 ? l ? L,

(9)

where m = nL, —z—2 denotes the vector of entrywise squared magnitudes, and F is the DFT matrix. Here, D (l) is a diagonal matrix whose diagonal entries are randomly drawn from {1, ?1, j, ?j}, which models signal modulation before diffraction. We generate L = 12 masks for measurements, and run TWF on a MacBook Pro with a 3 GHz Intel Core i7. We run 50 truncated power iterations and 50 TWF iterations, which in total cost 43.9 seconds for each color channel. The relative errors after initialization and TWF iterations are 0.4773 and 2.2 ? 10?5 , respectively; see Fig. 2. 1.4

Main results

We corroborate the preceding numerical findings with theoretical support. For concreteness, we assume TWF proceeds according to the Poisson log-likelihood (5). We suppose the samples (yi , ai) are independently and randomly drawn from the population, and model the random features ai as ai ? N (0, I n) .

(10)

To start with, the following theorem confirms the performance of TWF under noiseless data. 2 Similar phenomena arise in many other experiments we?ve conducted (e.g. when the sample size m ranges from 6n to 20n). In fact, this factor seems to improve slightly as m/n increases. 3 2 To justify the definition of SNR, note that the signals and noise are captured by ?i = (a¿ i x) and yi ? ?i ,

respectively. The SNR is thus given by

Pm ?2 Pm i=1 i i=1 Var[yi]

=

4

Pm ¿ 4 i=1 —ai x— Pm ¿ 2 i=1 —ai x—

?

3mkxk4 mkxk2

= 3kxk2 .

Theorem 1 (Exact recovery). Consider the noiseless case (1) with an arbitrary x ? Rn . Suppose that the learning rate ?t is either taken to be a constant ?t ? ? ¿ 0 or chosen via a backtracking line search. Then there exist some constants 0 ¡ ?, ? ¡ 1 and ?0 , c0 , c1 , c2 ¿ 0 such that with probability exceeding 1 ? c1 exp (?c2 m), the TWF estimates (Algorithm 1) obey dist(z (t) , x) ? ?(1 ? ?)t kxk, ?t ? N, provided that m ? c0 n and ? ? ?0 . As discussed below, we can take ?0 ? 0.3.

(11)

Theorem 1 justifies two intriguing properties of TWF. To begin with, TWF recovers the ground truth exactly as soon as the number of equations is on the same order of the number of unknowns, which is information theoretically optimal. More surprisingly, TWF converges at a geometric rate, i.e. it achieves -accuracy (i.e. dist(z (t) , x) ? kxk) within at most O (log 1/) iterations. As a result, the time taken for TWF to solve the quadratic systems is proportional to the time taken to read the data, which confirms the linear-time complexity of TWF. These outperform the theoretical guarantees of WF [13], which requires O(mn2 log 1/) runtime and O(n log n) sample complexity. Notably, the performance gain of TWF is the result of the key algorithmic changes. Rather than maximizing the data usage at each step, TWF exploits the samples at hand in a more selective manner, which effectively trims away those components that are too influential on either the initial guess or the search directions, thus reducing the volatility of each movement. With a tighter initial guess and better-controlled search directions in place, TWF is able to proceed with a more aggressive learning rate. Taken collectively these efforts enable the appealing convergence property of TWF. Next, we turn to more realistic noisy data by accounting for a general additive noise model: 2

yi = —hai , xi— + ?i , 1 ? i ? m, (12) where ?i represents a noise term. The stability of TWF is demonstrated in the theorem below. Theorem 2 (Stability). Consider the noisy case (12). Suppose that the learning rate ?t is either taken to be a positive constant ?t ? ? or chosen via a backtracking line search. If 2

m ? c0 n, ? ? ?0 , and k?k? ? c1 kxk , then with probability at least 1 ? c2 exp (?c3 m), the TWF estimates (Algorithm 1) satisfy k?k dist(z (t) , x) . ? + (1 ? ?)t kxk, ?t ? N mkxk for all x ? Rn . Here, 0 ¡ ? ¡ 1 and ?0 , c0 , c1 , c2 , c3 ¿ 0 are some universal constants. Alternatively, if one regards the SNR for the model (12) as follows Xm

SNR := —hai , xi—4 / k?k2 ? 3mkxk4 / k?k2 , i=1

(13)

(14)

(15)

then we immediately arrive at another form of performance guarantee stated in terms of SNR: 1 kxk + (1 ? ?)t kxk, ?t ? N. (16) dist(z (t) , x) . ? SNR As a consequence, the relative error of TWF reaches O(SNR?1/2) within a logarithmic number of iterations. It is worth emphasizing that the above stability guarantee is deterministic, which holds for any noise structure obeying (13). Encouragingly, this statistical accuracy is nearly un-improvable, as revealed by a minimax lower bound that we provide in the supplemental materials. We pause to remark that several other nonconvex methods have been proposed for solving quadratic equations, which exhibit intriguing empirical performances. A partial list includes the error reduction schemes by Fienup [2], alternating minimization [14], Kaczmarz method [17], and generalized approximate message passing [15]. However, most of them fall short of theoretical support. The analytical difficulty arises since these methods employ the same samples in each iteration, which introduces complicated dependencies across all iterates. To circumvent this issue, [14] proposes a sample-splitting version of the alternating minimization method that employs fresh samples in each iteration. Despite the mathematical convenience, the sample complexity of this approach is O(n log3 n + n log2 n log 1/), which is a factor of O(log3 n) from optimal and is empirically largely outperformed by the variant that reuses all samples. In contrast, our algorithm uses the same pool of samples all the time and is therefore practically appealing. Besides, the approach in [14] does not come with provable stability guarantees. Numerically, each iteration of Fienup?s algorithm (or alternating minimization) involves solving a least squares problem, and the algorithm converges in tens or hundreds of iterations. This is computationally more expensive than TWF, whose computational complexity is merely about 4 times that of solving a least squares problem. 5

2

Algorithm: Truncated Wirtinger Flow

This section explains the basic principles of truncated Wirtinger flow. For notational convenience,

M a for any M ? Rn?n . we denote A := [a1 , ? ? ? , am]¿ and A (M) := a¿ i i 1?i?m 2.1

Truncated gradient stage x = (2.7, 8) z = (3, 6)

In the case of independent real-valued data, the descent direction of the WF updates?which is the gradient of the Poisson log-likelihood?can be expressed as follows: m X i=1

¿(z; yi) =

m 2 X yi ? —a¿ i z— ai , 2 ¿ ai z i=1 — {z }

(17)

:=?i

where ?i represents the weight assigned to each feature ai . Figure 3: The locus of ? 12 ¿i (z) for all unit vectors ai . The red arrows depict those directions with large weights.

Unfortunately, the gradient of this form is non-integrable and hence uncontrollable. To see this, consider any fixed z ? Rn . 1 The typical value of min1?i?m —a¿ i z— is on the order of m kzk, leading to some excessively large weights ?i . Notably, an underlying premise for a nonconvex procedure to succeed is to ensure all iterates reside within a basin of attraction, that is, a neighborhood surrounding x within which x is the unique stationary point of the objective. When a gradient is unreasonably large, the iterative step might overshoot and end up leaving this basin of attraction. Consequently, WF moving along the preceding direction might not come close to the truth unless z is already very close to x. This is observed in numerical simulations4 . TWF addresses this challenge by discarding terms having too high of a leverage on the search direction; this is achieved by regularizing the weights ?i via appropriate truncation. Specifically, ?t (18) z (t+1) = z (t) + ¿tr (z (t)), ?t ? N, m where ¿tr (?) denotes the truncated gradient given by ¿tr (z) :=

Xm i=1

2

2 yi ? —a¿ i z— ai 1E1i (z)?E2i (z) ¿ ai z

(19)

for some appropriate truncation criteria specified by E1i (?) and E2i (?). In our algorithm, we take E1i (z) and E2i (z) to be two collections of events given by

lb

ub

E1i (z) := ?z kzk ? a¿ (20) i z ? ?z kzk ;

¿

?h 2

y ? A zz ¿ —ai z— , E2i (z) := —yi ? —a¿ (21) i z— — ? 1 kzk m where ?zlb , ?zub , ?z are predetermined truncation thresholds. In words, we drop components whose size fall outside some confidence range?a range where the magnitudes of both the numerator and denominator of ?i are comparable to their respective mean values. This paradigm could be counter-intuitive at first glance, since one might expect the larger terms to be better aligned with the desired search direction. The issue, however, is that the large terms are extremely volatile and could dominate all other components in an undesired way. In contrast, TWF makes use of only gradient components of typical sizes,

which slightly increases the bias but remarkably reduces the variance of the descent direction. We expect such gradient regularization and variance reduction schemes to be beneficial for solving a broad family of nonconvex problems. 2.2

Truncated spectral initialization

A key step to ensure meaningful convergence is to seed TWF with some point inside the basin of attraction, which proves crucial for other nonconvex procedures as well. An appealing initialization 4

For complex-valued data, WF converges empirically, as mini —a¿ i z— is much larger than the real case.

6

Algorithm 1 Truncated Wirtinger Flow. Input: Measurements {yi — 1 ? i ? m} and feature vectors {ai — 1 ? i ? m}; truncation thresholds ?zlb , ?zub , ?h , and ?y satisfying (by default, ?zlb = 0.3, ?zub = ?h = 5, and ?y = 3) 0 ¡ ?zlb ? 0.5, Initialize z (0) to be

q

?zub ? 5,

?h ? 5,

and ?y ? 3.

(25)

q

Pm 1 ? is the leading eigenvector of where ? = m i=1 yi and z X m 1 Y = yi ai a?i 1{—yi —??2y ?20 } . (22) i=1 m

Pmmn 2 ?? z, i=1 kai k

Loop: for t = 0 : T do

2

2?t Xm yi ? a?i z (t) z = z + ai 1E1i ?E2i , (23) i=1 m z (t)? ai where

? ? n —a?i z (t) — n —a?i z (t) — i lb ub i ? (t) 2 E1 := ?z ? ? ?z , E2 := —yi ? —ai z — — ? ?h Kt , (24) kai k kz (t) k kai k kz (t) k

1 Xm yl ? —a?l z (t) —2 . and Kt := l=1 m Output z (T) . (t+1)

(t)

(0) e procedure Pm is the¿spectral method [14] [13], which initializes z as the leading eigenvector of Y := 1 y a a . This is based on the observation that for any fixed unit vector x, i=1 i i i m

E[Ye] = I + 2xx¿ , whose principal component is exactly x with an eigenvalue of 3. Unfortunately, the success of this method requires a sample complexity exceeding n log n. To see ? k := ak /kak k, one can derive this, recall that maxi yi ? 2 log m. Letting k = arg maxi yi and a

¿ ¿e ?1 ¿ ? k ? (2n log m)/m, ?k ? a ? k m ak ak yk a ?k Y a a ? k is closer to the principal which dominates x¿ Ye x ? 3 unless m & n log m. As a result, a e component of Y than x when m n. This drawback turns out to be a substantial practical issue. 1

Relative MSE

This issue can be remedied if we preclude those data yi with large magnitudes when running the spectral method. Specifically, we propose to initialize z (0) as the leading eigenvector of 1 Xm Pm yi ai a¿ Y := (26) 1 i 1{—yi —??2y (m l=1 yl)} i=1 m

spectral method truncated spectral method
0.9
0.8
(0)
followed by proper scaling so as to ensure kz k ? kxk. As illustrated in Fig. 4, the empirical advantage of the truncated spectral method is increasingly more remarkable as n grows. 2.3
0.7 1000
2000
3000
4000
n: signal dimension (m = 6n)
5000
Figure 4: Relative initialization error when ai ? N (0, I).
Choice of algorithmic parameters
One important implementation detail is the learning rate ?t . There are two alternatives that work well in both theory and practice: 1. Fixed size. Take ?t ? ? for some constant ? ¿ 0. As long as ? is not too large, this strategy always works. Under the condition (25), our theorems hold for any positive constant ? ¡ 0.28. 2. Backtracking line search with truncated objective. This strategy performs a line search along the descent direction and determines an appropriate learning rate that guarantees a sufficient improvement with respect to the truncated objective. Details are deferred to the supplement. Another algorithmic details to specify are the truncation thresholds ?h , ?zlb , ?zub , and ?y . The present paper isolates a concrete set of combinations as given in (25). In all theory and numerical experiments presented in this work, we assume that the parameters fall within this range. 7
-20
TWF (Poisson objective) WF (Gaussian objective)
0.5
-30
0.5
0
3n
4n
5n
m : number of measurements (n =1000)
6n
-35 -40 -45 -50 -55 -60
0
2n
m = 6n m = 8n m = 10n
-25
Empirical success rate
Empirical success rate
1

Relative MSE (dB)
TWF (Poisson objective) WF (Gaussian objective)
1
2n
3n
4n
5n
m : number of measurements (n =1000)
6n
-65 15
20
25
30
35
40
SNR (dB) (n =1000)
45
50
55
(a) (b) (c) Figure 5: (a) Empirical success rates for real Gaussian design; (b) empirical success rates for complex Gaussian design; (c) relative MSE (averaged over 100 runs) vs. SNR for Poisson data.

3

More numerical experiments and discussion

We conduct more extensive numerical experiments to corroborate our main results and verify the applicability of TWF on practical problems. For all experiments conducted herein, we take a fixed step size ?t ? 0.2, employ 50 power iterations for initialization and T = 1000 gradient iterations. The truncation levels are taken to be the default values ?zlb = 0.3, ?zub = ?h = 5, and ?y = 3. We first apply TWF to a sequence of noiseless problems with n = 1000 and varying m. Generate the ind. object x at random, and produce the feature vectors ai in two different ways: (1) ai ? N (0, I); ind. ? obeys (2) ai ? N (0, I) + jN (0, I). A Monte Carlo trial is declared success if the estimate x dist (? x, x) / kxk ? 10?5 . Fig. 5(a) and 5(b) illustrate the empirical success rates of TWF (average over 100 runs for each m) for noiseless data, indicating that m ? 5n are m ? 4.5n are often sufficient under real and complex Gaussian designs, respectively. For the sake of comparison, we simulate the empirical success rates of WF, with the step size ?t = min{1 ? e?t/330 , 0.2} as recommended by [13]. As shown in Fig. 5, TWF outperforms WF under random Gaussian features, implying that TWF exhibits either better convergence rate or enhanced phase transition behavior. ind.

While this work focuses on the Poisson-type objective for concreteness, the proposed paradigm carries over to a variety of nonconvex objectives, and might have implications in solving other problems that involve latent variables, e.g. matrix completion [23?25], sparse coding [26], dictionary learning [27], and mixture problems (e.g. [28, 29]). We conclude this paper with an example on esti-

mating mixtures of linear regression. Imagine ¿ ai ? 1 , with probability p, yi ? 1 ? i ? m, (27) a¿ else, i ?2 , where ? 1 , ? 2 are unknown. It has been shown in [3] that in the noiseless case, the ground truth satisfies

Empirical success rate

Next, we empirically evaluate the stability of TWF under noisy data. Set n = 1000, produce ai ? N (0, I), and generate yi according to the Poisson model (3). Fig. 5(c) shows the relative mean square error?on the dB scale?with varying SNR (cf. (8)). As can be seen, the empirical relative MSE scales inversely proportional to SNR, which matches our stability guarantees in Theorem 2 (since on the dB scale, the slope is about -1 as predicted by the theory (16)). 1

0.5

0 5n

6n

7n

8n

9n

10n

m: number of measurements (n =1000)

Figure 6: Empirical success rate for mixed regression (p = 0.5).

¿ ¿ ¿ fi (?1 , ?2) := yi2 + 0.5a¿ i (? 1 ? 2 + ? 2 ? 1)ai ? ai (? 1 + ? 2) yi = 0,

1 ? i ? m,

which forms a set of quadratic constraints (in particular, if one further knows Pm ? 1 = ?? 2 , then this reduces to the form (1)). Running TWF with a nonconvex objective i=1 fi2 (z1 , z2) (with the assistance of a 1-D grid search proposed in [29] applied right after truncated initialization) yields accurate estimation of ? 1 , ? 2 under minimal sample complexity, as illustrated in Fig. 6. Acknowledgments E. C. is partially supported by NSF under grant CCF-0963835 and by the Math + X Award from the Simons Foundation. Y. C. is supported by the same NSF grant. 8

2 References

[1] A. Ben-Tal and A. Nemirovski. Lectures on modern convex optimization, volume 2. 2001. [2] J. R. Fienup. Phase retrieval algorithms: a comparison. Applied optics, 21:2758?2769, 1982. [3] Y. Chen, X. Yi, and C. Caramanis. A convex formulation for mixed regression with two components: Minimax optimal rates. In Conference on Learning Theory (COLT), 2014. [4] E. J. Cand?s, T. Strohmer, and V. Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. Communications on Pure and Applied Mathematics, 66(8):1017?1026, 2013. [5] I. Waldspurger, A. d?Aspremont, and S. Mallat. Phase recovery, maxcut and complex semidefinite programming. Mathematical Programming, 149(1-2):47?81, 2015. [6] Y. Shechtman, Y. C. Eldar, A. Szameit, and M. Segev. Sparsity based subwavelength imaging with partially incoherent light via quadratic compressed

sensing. Optics express, 19(16), 2011. [7] E. J. Cand?s and X. Li. Solving quadratic equations via PhaseLift when there are about as many equations as unknowns. Foundations of Computational Math., 14(5):1017?1026, 2014. [8] H. Ohlsson, A. Yang, R. Dong, and S. Sastry. Cprl?an extension of compressive sensing to the phase retrieval problem. In Advances in Neural Information Processing Systems (NIPS), 2012. [9] Y. Chen, Y. Chi, and A. J. Goldsmith. Exact and stable covariance estimation from quadratic sampling via convex programming. IEEE Trans. on Inf. Theory, 61(7):4034?4059, 2015. [10] T. Cai and A. Zhang. ROP: Matrix recovery via rank-one projections. Annals of Stats. [11] K. Jaganathan, S. Oymak, and B. Hassibi. Recovery of sparse 1-D signals from the magnitudes of their Fourier transform. In IEEE ISIT, pages 1473?1477, 2012. [12] D. Gross, F. Krahmer, and R. Kueng. A partial derandomization of phaselift using spherical designs. Journal of Fourier Analysis and Applications, 21(2):229?266, 2015. [13] E. J. Cand?s, X. Li, and M. Soltanolkotabi. Phase retrieval via Wirtinger flow: Theory and algorithms. IEEE Transactions on Information Theory, 61(4):1985?2007, April 2015. [14] P. Netrapalli, P. Jain, and S. Sanghavi. Phase retrieval using alternating minimization. NIPS, 2013. [15] P. Schniter and S. Rangan. Compressive phase retrieval via generalized approximate message passing. IEEE Transactions on Signal Processing, 63(4):1043?1055, Feb 2015. [16] A. Repetti, E. Chouzenoux, and J.-C. Pesquet. A nonconvex regularized approach for phase retrieval. International Conference on Image Processing, pages 1753?1757, 2014. [17] K. Wei. Phase retrieval via Kaczmarz methods. arXiv:1502.01822, 2015. [18] C. White, R. Ward, and S. Sanghavi. The local convexity of solving quadratic equations. arXiv:1506.07868, 2015. [19] Y. Shechtman, A. Beck, and Y. C. Eldar. GESPAR: Efficient phase retrieval of sparse signals. IEEE Transactions on Signal Processing, 62(4):928?938, 2014. [20] M. Soltanolkotabi. Algorithms and Theory for Clustering and Nonconvex Quadratic Programming. PhD thesis, Stanford University, 2014. [21] L. N. Trefethen and D. Bau III. Numerical linear algebra, volume 50. SIAM, 1997. [22] E. J. Cand?s, X. Li, and M. Soltanolkotabi. Phase retrieval from coded diffraction patterns. to appear in Applied and Computational Harmonic Analysis, 2014. [23] R. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. Journal of Machine Learning Research, 11:2057?2078, 2010. [24] P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In ACM symposium on Theory of computing, pages 665?674, 2013. [25] R. Sun and Z. Luo. Guaranteed matrix completion via nonconvex factorization. FOCS, 2015. [26] S. Arora, R. Ge, T. Ma, and A. Moitra. Simple, efficient, and neural algorithms for sparse coding. Conference on Learning Theory (COLT), 2015. [27] J. Sun, Q. Qu, and J. Wright. Complete dictionary recovery over the sphere. ICML, 2015. [28] S. Balakrishnan, M. J. Wainwright, and B. Yu. Statistical guarantees for the EM algorithm: From population to sample-based analysis. arXiv preprint arXiv:1408.2156, 2014. [29] X. Yi, C. Caramanis, and S. Sanghavi. Alternating minimization for mixed linear regression. International Conference on Machine Learning, June 2014.

9