

Fast Phase Retrieval for High-Dimensions

Aditya Viswanathan
aditya@math.msu.edu

MICHIGAN STATE

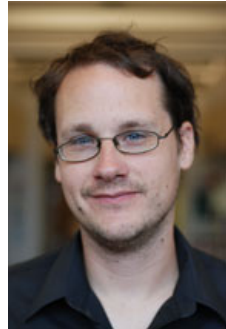
U N I V E R S I T Y

AMS Spring Central Sectional Meeting
Saturday, March 14th, 2015

Joint work with



Yang Wang



Mark Iwen

Research supported in part by National Science Foundation grant
DMS 1043034.

The Phase Retrieval Problem

$$\text{find } \mathbf{x} \in \mathbb{C}^d \text{ given } |M\mathbf{x}| = \mathbf{b} \in \mathbb{R}^D,$$

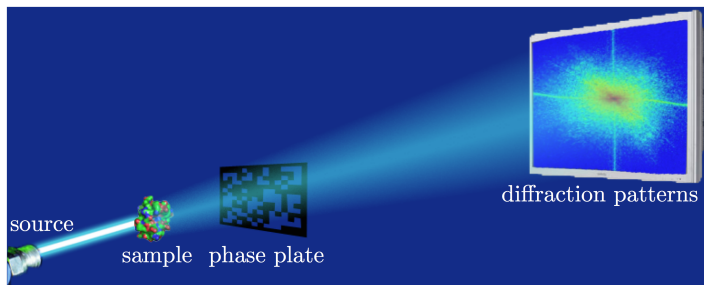
where

- $\mathbf{b} \in \mathbb{R}^D$ are the magnitude or intensity measurements.
- $M \in \mathbb{C}^{D \times d}$ is a measurement matrix associated with these measurements.

Let $\mathcal{A} : \mathbb{R}^D \rightarrow \mathbb{C}^d$ denote the recovery method.

The phase retrieval problem involves designing measurement matrix and recovery method pairs.

Applications of Phase Retrieval



From "Phase Retrieval from Coded Diffraction Patterns" by E. J. Candes, X. Li, and M. Soltanolkotabi.

Important applications of Phase Retrieval

- X-ray crystallography
- Diffraction imaging
- Transmission Electron Microscopy (TEM)

In many molecular imaging applications, the detectors only capture intensity measurements.

Objectives

- Computational Efficiency – Can the recovery algorithm \mathcal{A} be computed in $O(d \log^c d)$ -time? Here, c is a small constant.
- Computational Robustness: The recovery algorithm, \mathcal{A} , should be robust to additive measurement errors (i.e., noise).
- Minimal Measurements: The number of linear measurements, D , should be minimized to the greatest extent possible.

Some Previous Approaches

- Alternating Projection Methods [Gerchberg and Saxton, 1972] and [Fienup, 1978]
 - Work well in practice, not well understood theoretically
- PhaseLift [Candes et. al., 2012]
 - Recovery guarantees for random measurements
 - Requires $\mathcal{O}(d)$ measurements
 - Requires solving a SDP – $\mathcal{O}(d^3)$ -time
- Phase Retrieval with Polarization [Alexeev et. al. 2014]
 - Graph-theoretic frame-based approach
 - Requires $\mathcal{O}(d \log d)$ measurements
 - Error guarantee similar to PhaseLift

Overview of the Our Computational Framework

- 1 Use shifted compactly supported masks to obtain phase difference estimates.

$$|\text{Circ}(\mathbf{w})\mathbf{x}|^2 \xrightarrow[\text{linear system}]{\text{solve}} x_j \bar{x}_{j+k}, \quad k = 0, \dots, \delta$$

- \mathbf{w} is a mask, or window, with $\delta + 1$ non-zero entries.
 - $x_j \bar{x}_{j+k}$ gives us the (scaled) difference in phase between entries x_j and x_{j+k} .
- 2 Solve an angular synchronization problem on the phase differences to obtain the unknown signal.

$$x_j \bar{x}_{j+k} \xrightarrow[\text{synchronization}]{\text{angular}} x_j$$

Constraints on \mathbf{x} : We require \mathbf{x} to be non-sparse.
(The number of consecutive zeros in \mathbf{x} should be less than δ)

Overview of the Our Computational Framework

- 1 Use shifted compactly supported masks to obtain phase difference estimates.

$$|\text{Circ}(\mathbf{w})\mathbf{x}|^2 \xrightarrow[\text{linear system}]{\text{solve}} x_j \bar{x}_{j+k}, \quad k = 0, \dots, \delta$$

- \mathbf{w} is a mask, or window, with $\delta + 1$ non-zero entries.
 - $x_j \bar{x}_{j+k}$ gives us the (scaled) difference in phase between entries x_j and x_{j+k} .
- 2 Solve an angular synchronization problem on the phase differences to obtain the unknown signal.

$$x_j \bar{x}_{j+k} \xrightarrow[\text{synchronization}]{\text{angular}} x_j$$

Constraints on \mathbf{x} : We require \mathbf{x} to be non-sparse.
(The number of consecutive zeros in \mathbf{x} should be less than δ)

Correlations with Support-Limited Functions

- Let $\mathbf{x} = [x_0 \ x_1 \ \dots \ x_{d-1}]^T \in \mathbb{C}^d$ be the unknown signal.
- Let $\mathbf{w} = [w_0 \ w_1 \ \dots \ w_\delta \ 0 \ \dots 0]^T$ denote a support-limited mask. It has $\delta + 1$ non-zero entries.
- We are given the (squared) magnitude measurements

$$(b^m)^2 = |\text{Circ}(\mathbf{w}^m)\mathbf{x}|^2, \quad m = 0, \dots, L$$

corresponding to $L + 1$ distinct masks.

Correlations with Support-Limited Functions

Explicitly writing out each measurement, we have

$$\begin{aligned}(b_k^m)^2 &= \left| \sum_{j=0}^{\delta} \bar{w}_j^m \cdot x_{k+j} \right|^2 \\ &= \sum_{i,j=0}^{\delta} w_i^m \bar{w}_j^m x_{k+j} \bar{x}_{k+i}\end{aligned}$$

We can also lift these equations to a set of **linear equations!**

Solving for Phase Differences

Ordering $x_n \bar{x}_{n+l}$ lexicographically, we obtain a linear system of equations for the phase differences.

Example: $\mathbf{x} \in \mathbb{R}^d$, $d = 4$, $\delta = 1$

$$\underbrace{\begin{pmatrix}
 \begin{matrix} (w_0^0)^2 & 2w_0^0 w_1^0 \\ (w_0^1)^2 & 2w_0^1 w_1^1 \end{matrix} & \begin{matrix} (w_1^0)^2 & 0 \\ (w_1^1)^2 & 0 \end{matrix} & 0 & 0 & 0 & 0 \\
 0 & 0 & \begin{matrix} (w_0^0)^2 & 2w_0^0 w_1^0 \\ (w_0^1)^2 & 2w_0^1 w_1^1 \end{matrix} & \begin{matrix} (w_1^0)^2 & 0 \\ (w_1^1)^2 & 0 \end{matrix} & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & \begin{matrix} (w_0^0)^2 & 2w_0^0 w_1^0 \\ (w_0^1)^2 & 2w_0^1 w_1^1 \end{matrix} & \begin{matrix} (w_1^0)^2 & 0 \\ (w_1^1)^2 & 0 \end{matrix} & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & \begin{matrix} 2w_0^0 w_1^0 & (w_1^0)^2 \\ 2w_0^1 w_1^1 & (w_1^1)^2 \end{matrix} & 0 \\
 \begin{matrix} (w_1^0)^2 & 0 \\ (w_1^1)^2 & 0 \end{matrix} & 0 & 0 & 0 & 0 & 0 & \begin{matrix} 2w_0^0 w_1^0 & (w_1^0)^2 \\ 2w_0^1 w_1^1 & (w_1^1)^2 \end{matrix}
 \end{pmatrix}
 }_{M'}
 \underbrace{\begin{pmatrix}
 x_0 \bar{x}_0 \\
 x_0 \bar{x}_1 \\
 x_1 \bar{x}_1 \\
 x_1 \bar{x}_2 \\
 x_2 \bar{x}_2 \\
 x_2 \bar{x}_3 \\
 x_3 \bar{x}_3 \\
 x_3 \bar{x}_0
 \end{pmatrix}}_{\mathbf{y}}
 = \begin{pmatrix}
 b_0^0 \\
 b_1^0 \\
 b_1^1 \\
 b_2^0 \\
 b_2^1 \\
 b_2^2 \\
 b_3^0 \\
 b_3^1
 \end{pmatrix}$$

The system matrix M' is *block circulant*!

Some Entries of the Measurement Matrix

Two strategies

- Random entries (Gaussian, Uniform, Bernoulli, ...)
- Structured measurements, e.g.,

$$w_i^\ell = \begin{cases} \frac{e^{-i/a}}{\sqrt[4]{2\delta+1}} \cdot e^{\frac{2\pi i \cdot i \cdot \ell}{2\delta+1}}, & i \leq \delta \\ 0, & i > \delta \end{cases} .$$

where $a := \max \left\{ 4, \frac{\delta-1}{2} \right\}$, and $0 \leq \ell \leq L$.

Structured Measurements

Theorem (Iwen, V., Wang 2015)

Choose entries of the measurement mask w^m as follows:

$$w_i^\ell = \begin{cases} \frac{e^{-i/a}}{\sqrt[4]{2\delta+1}} \cdot e^{\frac{2\pi i \cdot i \cdot \ell}{2\delta+1}}, & i \leq \delta \\ 0, & i > \delta \end{cases}, \quad a := \max \left\{ 4, \frac{\delta-1}{2} \right\}, \ell \in [0, L].$$

Then, the resulting system matrix for the phase differences, M' , has condition number

$$\kappa(M') < \max \left\{ 144e^2, \frac{9e^2}{4} \cdot (\delta-1)^2 \right\}.$$

Note:

- w_i^ℓ are scaled entries of a DFT matrix.
- δ is typically chosen to be 6–12.

Angular Synchronization

The Angular Synchronization Problem

Estimate d unknown angles $\theta_1, \theta_2, \dots, \theta_d \in [0, 2\pi)$ from $d(\delta + 1)$ noisy measurements of their differences

$$\Delta\theta_{ij} := \angle x_i - \angle x_j = \angle \left(\frac{x_i \bar{x}_j}{\sqrt{x_i \bar{x}_i \cdot x_j \bar{x}_j}} \right) \bmod 2\pi.$$

Angular Synchronization

The unknown phases (modulo a global phase offset) may be obtained by solving a simple greedy algorithm.

- 1 Set the largest magnitude component to have zero phase angle; i.e.,

$$\angle x_k = 0, \quad k := \operatorname{argmax}_i x_i \bar{x}_i.$$

- 2 Use this entry to set the phase angles of the next δ entries; i.e.,

$$\angle x_j = \angle x_k - \Delta\theta_{kj}, \quad j = 1, \dots, \delta.$$

- 3 Use the next largest magnitude component from these δ entries and repeat the process.

Recovering Arbitrary Vectors

- Recall: Due to compact support of our masks, only "flat" vectors can be recovered
- Arbitrary vectors can be "flattened" by multiplication with a random unitary matrix such as $W = PFB$, where
 - $P \in \{0, 1\}^{d \times d}$ is a permutation matrix selected uniformly at random from the set of all $d \times d$ permutation matrices
 - F is the unitary $d \times d$ discrete Fourier transform matrix
 - $B \in \{-1, 0, 1\}^{d \times d}$ is a random diagonal matrix with i.i.d. symmetric Bernoulli entries on its diagonal

A Noiseless Recovery Result

Theorem (Iwen, V., Wang 2015)

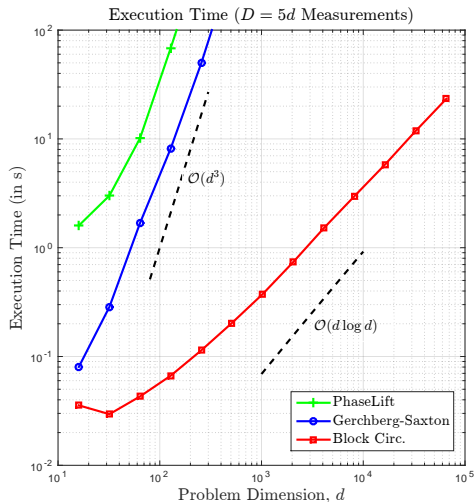
Let $\mathbf{x} \in \mathbb{C}^d$ with d sufficiently large. Then, one can select a random measurement matrix $\tilde{M} \in \mathbb{C}^{D \times d}$ such that the following holds with probability at least $1 - \frac{1}{c \cdot \ln^2(d) \cdot \ln^3(\ln d)}$: Our algorithm will recover an $\tilde{\mathbf{x}} \in \mathbb{C}^d$ with

$$\min_{\theta \in [0, 2\pi]} \left\| \mathbf{x} - e^{i\theta} \tilde{\mathbf{x}} \right\|_2 = 0$$

when given the noiseless magnitude measurements $|\tilde{M}\mathbf{x}|^2 \in \mathbb{R}^D$. Here D can be chosen to be $\mathcal{O}(d \cdot \ln^2(d) \cdot \ln^3(\ln d))$. Furthermore, the algorithm will run in $\mathcal{O}(d \cdot \ln^3(d) \cdot \ln^3(\ln d))$ -time in that case.

To do: Robustness to measurement noise...

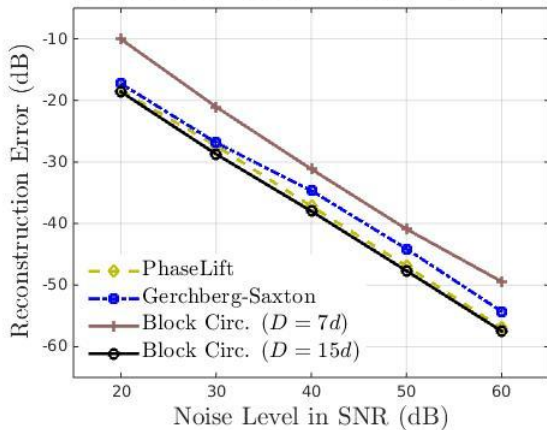
Efficiency



- iid Complex Gaussian signal
- High SNR applications
- $5d$ measurements
- $64k$ problem in ~ 20 s in Matlab!

Robustness

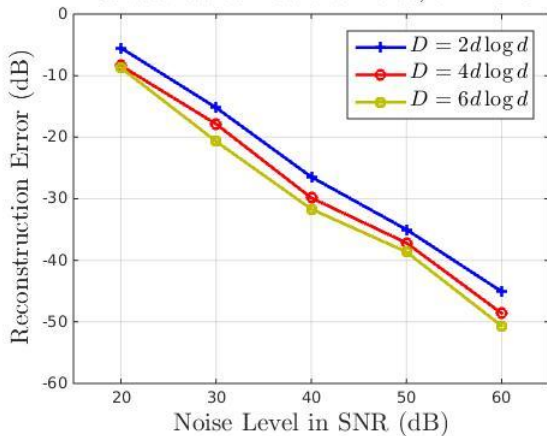
Robustness to Additive Noise, $d = 64$, $D = 7d$



- iid complex Gaussian signal
- $d = 64$
- $7d$ measurements
- Deterministic (windowed Fourier-like) measurements

Robustness

Robustness to Additive Noise, $d = 2048$



- iid complex Gaussian signal
- $d = 2048$
- Not computationally feasible using PhaseLift (on a laptop in Matlab)
- Deterministic (windowed Fourier-like) measurements

The *Sparse* Phase Retrieval Problem

$$\text{find } \mathbf{x} \in \mathbb{C}^d \quad \text{given} \quad |\mathcal{M}\mathbf{x}| = \mathbf{b} \in \mathbb{R}^D,$$

where

- \mathbf{x} is s -sparse, with $s \ll d$.
- $\mathbf{b} \in \mathbb{R}^D$ are the magnitude or intensity measurements.
- $\mathcal{M} \in \mathbb{C}^{D \times d}$ is a measurement matrix associated with these measurements.

Let $\mathcal{A} : \mathbb{R}^D \rightarrow \mathbb{C}^d$ denote the recovery method.

The sparse phase retrieval problem involves designing measurement matrix and recovery method pairs.

Sublinear-time Results

Theorem (Iwen, V., Wang 2015)

There exists a deterministic algorithm $\mathcal{A} : \mathbb{R}^D \rightarrow \mathbb{C}^d$ for which the following holds: Let $\epsilon \in (0, 1]$, $\mathbf{x} \in \mathbb{C}^d$ with d sufficiently large, and $s \in [d]$. Then, one can select a random measurement matrix $\tilde{M} \in \mathbb{C}^{D \times d}$ such that

$$\min_{\theta \in [0, 2\pi]} \left\| e^{i\theta} \mathbf{x} - \mathcal{A} \left(|\tilde{M} \mathbf{x}|^2 \right) \right\|_2 \leq \left\| \mathbf{x} - \mathbf{x}_s^{\text{opt}} \right\|_2 + \frac{22\epsilon \left\| \mathbf{x} - \mathbf{x}_{(s/\epsilon)}^{\text{opt}} \right\|_1}{\sqrt{s}}$$

is true with probability at least $1 - \frac{1}{C \cdot \ln^2(d) \cdot \ln^3(\ln d)}$.^a Here D can be chosen to be $\mathcal{O} \left(\frac{s}{\epsilon} \cdot \ln^3 \left(\frac{s}{\epsilon} \right) \cdot \ln^3 \left(\ln \frac{s}{\epsilon} \right) \cdot \ln d \right)$. Furthermore, the algorithm will run in $\mathcal{O} \left(\frac{s}{\epsilon} \cdot \ln^4 \left(\frac{s}{\epsilon} \right) \cdot \ln^3 \left(\ln \frac{s}{\epsilon} \right) \cdot \ln d \right)$ -time in that case.^b

^aHere $C \in \mathbb{R}^+$ is a fixed absolute constant.

^bFor the sake of simplicity, we assume $s = \Omega(\log d)$ when stating the measurement and runtime bounds above.

References – Phase Retrieval

- ① R.W. Gerchberg and W.O. Saxton. *A practical algorithm for the determination of phase from image and diffraction plane pictures*. *Optik*, 35:237–246, 1972
- ② J.R. Fienup. *Reconstruction of an object from the modulus of its Fourier transform*. *Optics Letters*, 3:27–29, 1978.
- ③ E. J. Candes, T. Strohmer, and V. Voroninski. *Phaselift: exact and stable signal recovery from magnitude measurements via convex programming*. *Comm. Pure Appl. Math.*, 66, 1241–1274, 2012.
- ④ B. Alexeev, A. S. Bandeira, M. Fickus, and D. G. Mixon. *Phase retrieval with polarization*. *SIAM J. Imag. Sci.*, 7(1), 35–66.

References – Sparse Phase Retrieval

- ① H. Ohlsson, A. Yang, R. Dong, and S. Sastry. *CPRL: An Extension of Compressive Sensing to the Phase Retrieval Problem*. Proc. 26th Conf. Adv. Neural Inf. Proc. Sys., 1376–1384, 2012.
- ② Y. Shechtman, A. Beck, and Y. C. Eldar. *GESPAR: Efficient Phase Retrieval of Sparse Signals*. IEEE Tran. Sig. Proc., 62(4):928–938, 2014.
- ③ P. Schniter and S. Rangan, *Compressive Phase Retrieval via Generalized Approximate Message Passing*. arXiv:1405.5618, 2014.
- ④ M. Iwen, A. Viswanathan, and Y. Wang, *Robust Sparse Phase Retrieval Made Easy*. arXiv:1410.5295, 2014.

Software Repository

The screenshot shows a Bitbucket repository page for 'charms/blockPR'. The browser address bar shows the URL 'https://bitbucket.org/charms/blockpr'. The repository is owned by 'charms' and is named 'blockPR'. The overview section displays statistics: 1 Branch, 1 Tag, 0 Forks, and 2 Watchers. The repository description is 'BlockPR: Matlab Software for Phase Retrieval using Block-Circulant Measurement Constructions'. It mentions that the repository contains Matlab code for solving the phase retrieval problem and provides details of the method, theoretical guarantees, and representative numerical results. The software was developed at the Department of Mathematics, Michigan State University and is released under the MIT license. It was developed and tested using Matlab R2014a and uses TFOCS, a Matlab software package for the efficient construction and solution of convex optimization problems. A copy of the TFOCS package is included under the third party software directory at third/.

Overview

SSH `git@bitbucket.org:charms/blockpr.git` Share

Last updated	2015-01-09	1	1
Language	MATLAB	Branch	Tag
Access level	Admin (revoke)	0	2
		Forks	Watchers

[Edit README](#)

BlockPR: Matlab Software for Phase Retrieval using Block-Circulant Measurement Constructions

This repository contains Matlab code for solving the phase retrieval problem. Details of the method, theoretical guarantees and representative numerical results can be found in

Fast Phase Retrieval for High-Dimensions
Mark Iwen, Aditya Viswanathan and Yang Wang
2015

This software was developed at the Department of Mathematics, Michigan State University and is released under the MIT license.

The software was developed and tested using Matlab R2014a and uses TFOCS, a Matlab software package for the efficient construction and solution of convex optimization problems. A copy of the TFOCS package is included under the third party software directory at third/.

Directory Structure and Contents

ACTIONS

- Clone
- Create branch
- Create pull request
- Compare
- Fork

NAVIGATION

- Overview
- Source
- Commits
- Branches
- Pull requests
- Issues
- Wiki
- Downloads

Invite users to this repo

Send invitation

Recent activity

- 1 commit
Pushed to charms/blockPR
1ed5f91 Fixed typos in readme
Aditya Viswanathan - 2015-01-09
- 1 commit
Pushed to charms/blockPR
c9916ac Added TFOCS for later use with ...
Aditya Viswanathan - 2015-01-09
- 1 commit
Pushed to charms/blockPR
c88f96f Misc. format edits; Added Alt. Pro...
Aditya Viswanathan - 2015-01-09
- 12 commits
Pushed to charms/blockPR
bb6756b Merged in complex (pull request ...

Software Repository

The screenshot shows a Bitbucket repository page for 'charms/sparsepr'. The browser address bar shows the URL 'https://bitbucket.org/charms/sparsepr'. The page header includes the Bitbucket logo, navigation tabs (Dashboard, Teams, Repositories, Create), and a search bar. The left sidebar contains navigation options: Overview (selected), Source, Commits, Branches, Pull requests, Downloads, and Settings.

Overview

SSH `git@bitbucket.org:charms/sparsepr.git` Share

Last updated	2014-11-08	2	0
Language	MATLAB	Branches	Tags
Access level	Admin (revoke)	0	2
		Forks	Watchers

Edit README

SparsePR: Matlab Software for Sparse Phase Retrieval

This repository contains Matlab code for solving the sparse phase retrieval problem. Details of the method, theoretical guarantees and representative numerical results can be found in

Robust Sparse Phase Retrieval Made Easy
Mark Iwen, Aditya Viswanathan and Yang Wang
[arXiv](#)
2014

This software was developed at the [Department of Mathematics, Michigan State University](#) and is released under the MIT license.

The software was developed and tested using Matlab R2014a and uses TFOCS, a Matlab software package for the efficient construction and solution of convex optimization problems. A copy of the TFOCS package is included under the third party software directory at [third/](#). A selection of scripts also uses the CVX optimization software, which can be downloaded [here](#).

Recent activity

- 1 commit
Pushed to charms/sparsepr
4c3f31d Now uses a single routine to gen...
Aditya Viswanathan - 2014-11-08
- 1 commit
Pushed to charms/sparsepr
d469888 Added link to preprint at arXiv in L...
Aditya Viswanathan - 2014-10-24
- 1 commit
Pushed to charms/sparsepr
469782a Fixed SparsePR runtime/no. of m...
Aditya Viswanathan - 2014-10-16
- 1 commit
Pushed to charms/sparsepr
257f75d Initial commit
Aditya Viswanathan - 2014-10-15
- charms/sparsepr
Repository created
Aditya Viswanathan - 2014-10-15