

Block-Circulant Constructions for Robust and Efficient Phase Retrieval

Aditya Viswanathan
aditya@math.msu.edu

MICHIGAN STATE

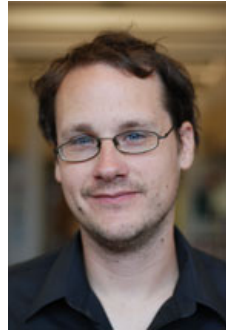
U N I V E R S I T Y

University of Michigan, Dearborn Mathematics Colloquium
8th April 2015

Joint work with



Yang Wang



Mark Iwen

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

Outline

- 1 The Phase Retrieval Problem
- 2 Existing Approaches
- 3 Proposed Computational Framework
- 4 Numerical Results
- 5 Extensions: Sparse Phase Retrieval

The Phase Retrieval Problem

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

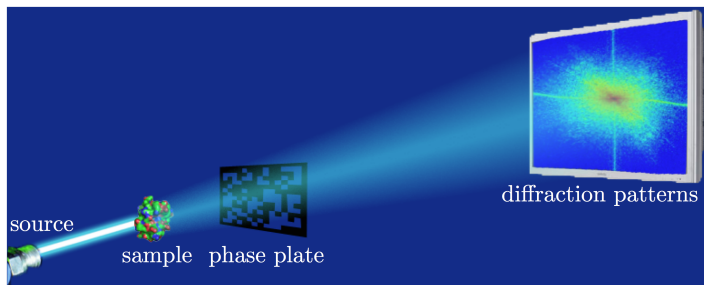
where

- $\mathbf{b} \in \mathbb{R}^D$ 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.

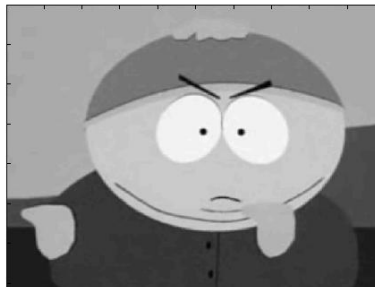
Important applications of Phase Retrieval

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

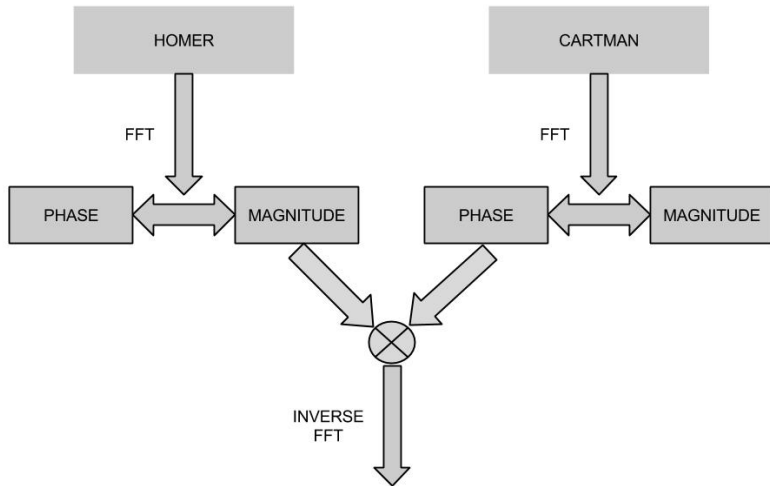
In many molecular imaging applications, the detectors only capture intensity measurements.

The Importance of Phase – An Illustration

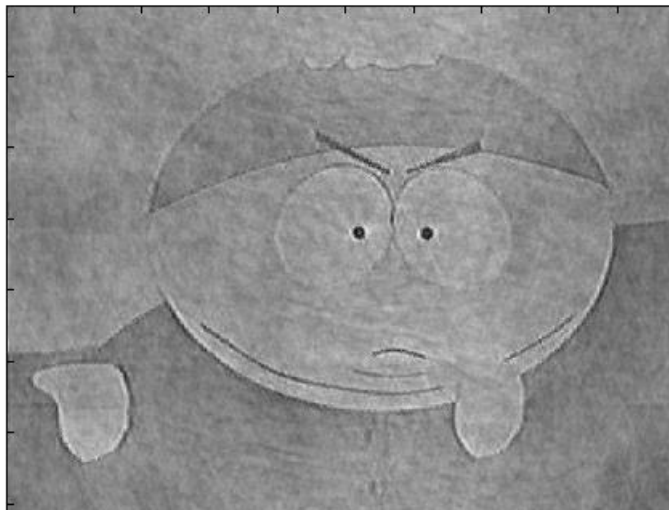
- The phase encapsulates vital information about a signal
- Key features of the signal are retained even if the magnitude is lost



The Importance of Phase – An Illustration



The Importance of Phase – An Illustration



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.

Outline

- 1 The Phase Retrieval Problem
- 2 Existing Approaches**
- 3 Proposed Computational Framework
- 4 Numerical Results
- 5 Extensions: Sparse Phase Retrieval

Alternating Projection Methods

[Gerchberg and Saxton, 1972] and [Fienup, 1978]

- These methods operate by alternately projecting the current iterate of the signal estimate over two sets of constraints.
- One of the constraints is the magnitude of the measurements.
- The other constraint depends on the application – positivity, support constraints, . . .

Alternating Projection Methods

[Gerchberg and Saxton, 1972] and [Fienup, 1978]

Algorithm 1 Gerchberg–Saxton

Input: Measurements $\mathbf{b} = |M\mathbf{x}| \in \mathbb{R}^D$, Initial estimate $\mathbf{x}_0 \in \mathbb{C}^d$.

- 1: **for** $i = 0$ to $N - 1$ **do**
 - 2: Compute $\mathbf{y} = M\mathbf{x}_i$
 - 3: Set $\tilde{\mathbf{y}} = \mathbf{b} \angle \mathbf{y}$
 - 4: Compute $\mathbf{x}_{i+1} = M^\dagger \tilde{\mathbf{y}}$
 - 5: **end for**
-

- N is the number of iterations
- M^\dagger is the Moore-Penrose pseudo-inverse

Alternating Projection Methods

[Gerchberg and Saxton, 1972] and [Fienup, 1978]

Issues

- Convergence is slow – the algorithm is likely to stagnate at stationary points
- Requires careful selection of and tuning of the parameters
- Mathematical aspects of the algorithm not well known. If there is proof of convergence, it is only for special cases.

Applications

- Can be used as a post-processing step to speed up more rigorous (but slow) computational approaches

PhaseLift [Candes et. al., 2012]

- Modify the problem to that of finding the rank-1 matrix $X = \mathbf{x}\mathbf{x}^*$
- Uses multiple random illuminations (or masks) as measurements.
- The resulting problem can be cast as a rank minimization optimization problem (NP hard)
- Instead, solve a convex relaxation – trace minimization problem (SDP)

PhaseLift [Candes et. al., 2012]

Notation

- Let \mathbf{w}^m be a mask. The measurements may be written as

$$|\langle \mathbf{w}^m, \mathbf{x} \rangle|^2 = \text{Tr}(\mathbf{x}^* \mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x}) = \text{Tr}(\mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x} \mathbf{x}^*) \\ := \text{Tr}(W^m X).$$

- Let \mathcal{W} be the linear operator mapping positive semidefinite matrices into $\{\text{Tr}(W^m X) : m = 0, \dots, L\}$.
- The phase retrieval problem then becomes

$$\begin{array}{ll} \text{minimize} & \text{rank}(X) \\ \text{subject to} & \mathcal{W}(X) = b \\ & X \succeq 0 \end{array}$$

PhaseLift [Candes et. al., 2012]

Notation

- Let \mathbf{w}^m be a mask. The measurements may be written as

$$\begin{aligned} |\langle \mathbf{w}^m, \mathbf{x} \rangle|^2 &= \text{Tr}(\mathbf{x}^* \mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x}) = \text{Tr}(\mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x} \mathbf{x}^*) \\ &:= \text{Tr}(W^m X). \end{aligned}$$

- Let \mathcal{W} be the linear operator mapping positive semidefinite matrices into $\{\text{Tr}(W^m X) : k = 0, \dots, L\}$.
- The phase retrieval problem then becomes

$$\begin{array}{ll} \text{minimize} & \text{rank}(X) \\ \text{subject to} & \mathcal{W}(X) = b \\ & X \succeq 0 \end{array}$$

- Unfortunately, this problem is **NP hard!**

PhaseLift [Candes et. al., 2012]

Notation

- Let \mathbf{w}^m be a mask. The measurements may be written as

$$\begin{aligned} |\langle \mathbf{w}^m, \mathbf{x} \rangle|^2 &= \text{Tr}(\mathbf{x}^* \mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x}) = \text{Tr}(\mathbf{w}^m (\mathbf{w}^m)^* \mathbf{x} \mathbf{x}^*) \\ &:= \text{Tr}(W^m X). \end{aligned}$$

- Let \mathcal{W} be the linear operator mapping positive semidefinite matrices into $\{\text{Tr}(W^m X) : k = 0, \dots, L\}$.
- Instead, use the convex relaxation

$$\begin{array}{ll} \text{minimize} & \text{trace}(X) \\ \text{subject to} & \mathcal{W}(X) = b \\ & X \succeq 0 \end{array}$$

- Implemented using a semidefinite program (SDP).

PhaseLift [Candes et. al., 2012]

Advantages

- Recovery guarantees for random measurements
- Optimization problems of the above type are well-understood
- Mature software for solving the resulting optimization problem.

Disadvantages

- SDP solvers are still slow!
- General-purpose solvers have complexity $\mathcal{O}(d^3)$; FFT-based measurements may be solved in $\mathcal{O}(d^2)$ time

Other Approaches

- 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
- Phase Recovery, MaxCut and Complex Semidefinite Programming [Waldspurger et. al. 2013]
 - Related to graph partitioning problems
 - Can be shown to be equivalent to PhaseLift under certain conditions
 - Requires solving a SDP

Outline

- 1 The Phase Retrieval Problem
- 2 Existing Approaches
- 3 Proposed Computational Framework**
- 4 Numerical Results
- 5 Extensions: Sparse Phase Retrieval

Overview of the Computational Framework

- 1 Use compactly supported masks and correlation measurements to obtain phase difference estimates.

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

- \mathbf{w} is a mask or window function 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 Computational Framework

- 1 Use compactly supported masks and correlation measurements to obtain phase difference estimates.

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

- \mathbf{w} is a mask or window function 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) correlation measurements

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

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

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

The system matrix M' is *block circulant*!

Consequences of the Block Circulant Structure

- There exists a unitary decomposition of the system matrix
- The condition number of the system matrix is a function of the individual blocks. In particular, we have

$$\kappa(M') = \frac{\max_{|t|=1} \sigma_1(t)}{\min_{|t|=1} \sigma_\delta(t)},$$

where $\sigma_1(t), \sigma_\delta(t)$ are the largest/smallest singular values of

$$J(t) = M'_0 + tM'_1 + \dots + t^{\delta-1}M'_\delta$$

- The linear system for the phase differences can be solved efficiently using FFTs

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 $0 \leq \ell \leq L$.

Random Measurements

Representative Condition Numbers

	$\delta = 3$	$\delta = 6$	$\delta = 9$
Critical sampling	34.89	124.05	465.32
Oversampling factor 2	3.73	11.73	13.30
Oversampling factor 3	3.29	6.08	8.60

- Critical sampling: $D = (2\delta + 1)d$, where D denotes the number of measurements.
- Oversampling: $D = \gamma \cdot (2\delta + 1)d$, where γ is the oversampling factor
- Typically, we use $\gamma = 1.5$.

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.
- No oversampling necessary!

Angular Synchronization

- 1 Use compactly supported masks and correlation measurements to obtain phase difference estimates.

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

- \mathbf{w} is a mask or window function 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)

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...

Outline

- 1 The Phase Retrieval Problem
- 2 Existing Approaches
- 3 Proposed Computational Framework
- 4 Numerical Results**
- 5 Extensions: Sparse Phase Retrieval

Numerical Results

- Test signals (Real and Complex) – iid Gaussian, uniform random, sinusoidal signals
- Noise model

$$\tilde{\mathbf{b}} = \mathbf{b} + \tilde{\mathbf{n}}, \quad \tilde{\mathbf{n}} \sim U[0, a].$$

Value of a determines SNR

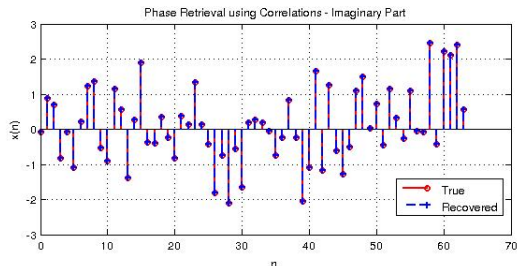
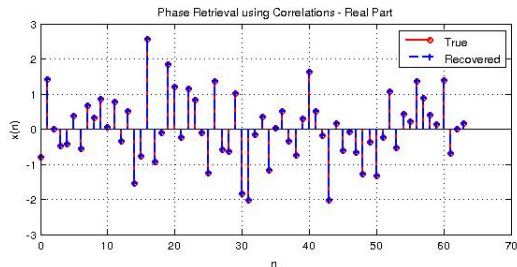
$$SNR = 10 \log_{10} \left(\frac{\text{noise power}}{\text{signal power}} \right) = 10 \log_{10} \left(\frac{a^2/3}{\|b\|^2/d} \right)$$

- Errors reported as SNR (dB)

$$\text{Error (dB)} = 10 \log_{10} \left(\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|^2}{\|\mathbf{x}\|^2} \right)$$

($\hat{\mathbf{x}}$ – recovered signal, \mathbf{x} – true signal)

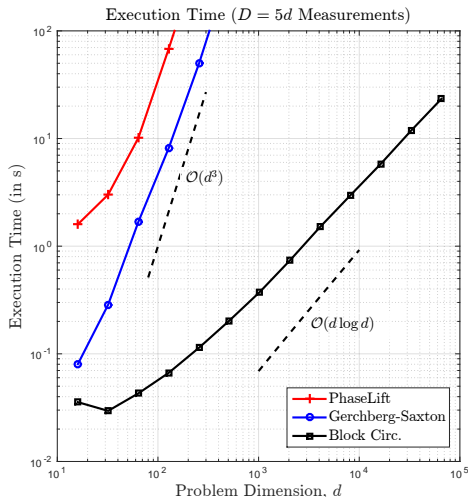
Noiseless Case



- iid Complex Gaussian signal
- $d = 64$
- $\delta = 1$
($3d$ measurements)
- No noise
- Reconstruction Error

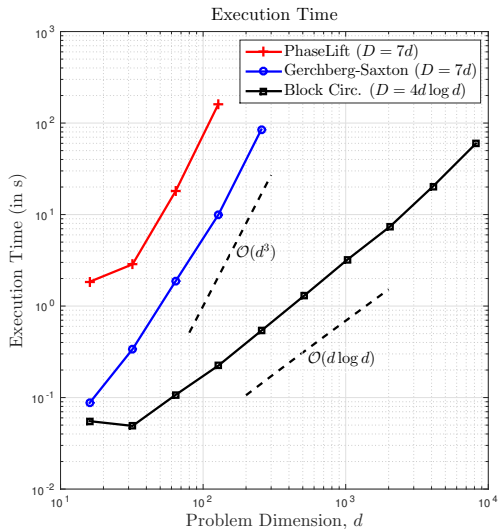
$$\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|^2}{\|\mathbf{x}\|^2} = 6.436 \times 10^{-15}$$

Efficiency



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

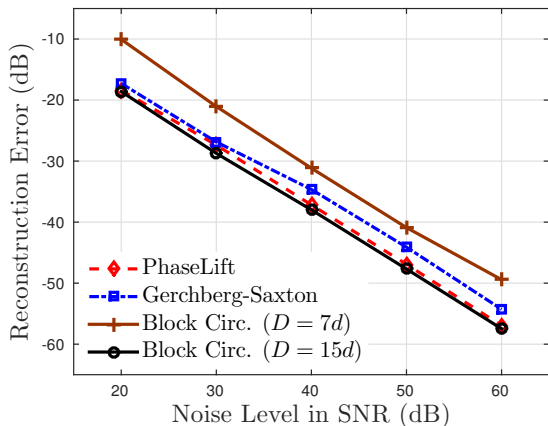
Efficiency



- iid Complex Gaussian signal
- Generic applications (wide range of SNRs)
- $4d \log d$ measurements

Robustness

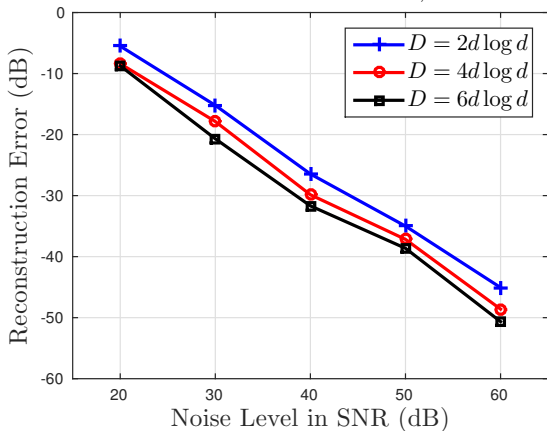
Robustness to Additive Noise, $d = 64, D = 7d$



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

Robustness

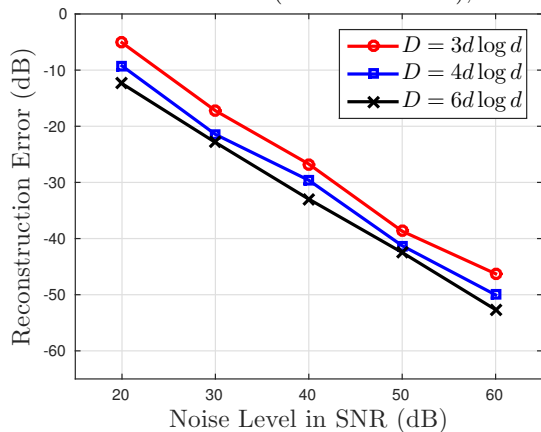
Robustness to Additive Noise, $d = 2048$



- iid complex Gaussian signal
- $d = 2048$
- Not feasible with PhaseLift or Alternating projection methods on a laptop in Matlab
- Deterministic (windowed Fourier-like) measurements

Robustness

Robustness to Noise (Random Masks), $d = 2048$



- iid complex Gaussian signal
- $d = 2048$
- Not feasible with PhaseLift or Alternating projection methods on a laptop in Matlab
- Random measurements

Discussion

(+)

- Well-conditioned deterministic measurement matrices with explicit condition number bounds
- Significantly faster (FFT time) than comparable SDP-based methods

(-)

- Requires $2\times$ to $4\times$ more measurements than equivalent SDP-based methods

Outline

- 1 The Phase Retrieval Problem
- 2 Existing Approaches
- 3 Proposed Computational Framework
- 4 Numerical Results
- 5 Extensions: Sparse Phase Retrieval**

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 k -sparse, with $k \ll d$.
- $\mathbf{b} \in \mathbb{R}^D$ 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.

Sparse Phase Retrieval – Objectives

- Computational Efficiency
- 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. In particular, can we have robust reconstruction for $D = O(k \log(d/k))$ measurements?

Existing Frameworks

- Alternating Projections with Sparsity Constraints [Mukherjee and Seelamantula, 2012]
- Compressive Phase Retrieval via Lifting (CPRL) [Ohlsson et. al., 2012]
- GrEedy Sparse PhAse Retrieval (GESPAR) algorithm [Shechtman et. al., 2014]
- Compressive Phase Retrieval via Generalized Approximate Message Passing [Schniter, Rangan 2014]

Proposed Computational Framework

Let the measurement matrix \mathcal{M} be of the form

$$\mathcal{M} = \mathcal{P}\mathcal{C},$$

where

- $\mathcal{P} \in \mathbb{C}^{D \times \tilde{d}}$ is an admissible phase retrieval matrix, and
- $\mathcal{C} \in \mathbb{C}^{\tilde{d} \times d}$ is an admissible compressive sensing matrix.

Note: We typically have $D = O(\tilde{d})$ and $\tilde{d} = O(k \log(d/k))$, where k is the sparsity of \mathbf{x} .

Proposed Computational Framework

- 1 Solve a (non-sparse) phase retrieval problem (PhaseLift shown here)

$$\begin{array}{ll} \text{minimize} & \text{trace}(Y) \\ \text{subject to} & \mathcal{W}(Y) = b \\ & Y \succeq 0, \end{array}$$

where $Y = \mathbf{y}\mathbf{y}^*$ and $\mathbf{y} \in \mathbb{C}^{\tilde{d}}$ is an intermediate solution.

- 2 Recover \mathbf{x} using a compressive sensing formulation

$$\begin{array}{ll} \text{minimize} & \|\mathbf{x}\|_1 \\ \text{subject to} & \mathcal{C}\mathbf{x} = \mathbf{y}. \end{array}$$

Proposed Computational Framework

- 1 Solve a (non-sparse) phase retrieval problem (PhaseLift shown here)

$$\begin{array}{ll} \text{minimize} & \text{trace}(Y) \\ \text{subject to} & \mathcal{W}(Y) = b \\ & Y \succeq 0, \end{array}$$

where $Y = \mathbf{y}\mathbf{y}^*$ and $\mathbf{y} \in \mathbb{C}^{\tilde{d}}$ is an intermediate solution.

- 2 Recover \mathbf{x} using a compressive sensing formulation

$$\begin{array}{ll} \text{minimize} & \|\mathbf{x}\|_1 \\ \text{subject to} & \mathcal{C}\mathbf{x} = \mathbf{y}. \end{array}$$

Advantages

- Dramatically reduces the problem dimension.
- Recovery guarantees follow naturally from guarantees for the Phase Retrieval method employed and Compressive Sensing
- Not limited to Phase Lift – Can work with any phase retrieval method (Step 1)

Error Guarantee

- Consider noisy measurements of the form

$$\mathbf{b} := |\mathcal{P}\mathcal{C}\mathbf{x}|^2 + \mathbf{n}$$

- $\mathcal{P} \in \mathbb{C}^{D \times \tilde{d}}$ is any phase retrieval matrix with an associated recovery algorithm $\Phi_{\mathcal{P}} : \mathbb{R}^D \rightarrow \mathbb{C}^{\tilde{d}}$ (and error guarantee)
- $\mathcal{C} \in \mathbb{C}^{\tilde{d} \times d}$ is any compressive sensing matrix with an associated recovery algorithm $\Delta_{\mathcal{C}} : \mathbb{C}^{\tilde{d}} \rightarrow \mathbb{C}^d$ (and error guarantee)
- Composition of the two recovery algorithms, $\Delta_{\mathcal{C}} \circ \Phi_{\mathcal{P}} : \mathbb{R}^D \rightarrow \mathbb{C}^d$, should accurately approximate $\mathbf{x} \in \mathbb{C}^d$, up to a global phase factor, from \mathbf{b} whenever \mathbf{x} is sufficiently sparse or compressible.

Error Guarantee

Theorem (PhaseLift: Candes, Li 2014)

Let $\mathcal{P} \in \mathbb{C}^{D \times \tilde{d}}$ have its D rows be independently drawn either uniformly at random from the sphere of radius $\sqrt{\tilde{d}}$ in $\mathbb{C}^{\tilde{d}}$, or else as complex normal random vectors from $\mathcal{N}(0, \mathcal{I}_{\tilde{d}}/2) + \mathbf{i}\mathcal{N}(0, \mathcal{I}_{\tilde{d}}/2)$. Then, \exists universal constants $\tilde{B}, \tilde{C}, \tilde{A} \in \mathbb{R}^+$ such that the PhaseLift procedure $\Phi_{\mathcal{P}} : \mathbb{R}^D \rightarrow \mathbb{C}^{\tilde{d}}$ satisfies

$$\min_{\theta \in [0, 2\pi]} \left\| \Phi_{\mathcal{P}}(\mathbf{b}) - e^{i\theta} \mathbf{x} \right\|_2 \leq \tilde{C} \cdot \min \left(\|\mathbf{x}\|_2, \frac{\|\mathbf{n}\|_1}{D \|\mathbf{x}\|_2} \right)$$

for all $\mathbf{x} \in \mathbb{C}^{\tilde{d}}$ with probability $1 - \mathcal{O}(e^{-\tilde{B}D})$, provided that $D \geq \tilde{A}\tilde{d}$.

Error Guarantee

Theorem (Compressive Sensing – Foucart, Rauhut)

Suppose that the matrix $C \in \mathbb{C}^{\tilde{d} \times d}$ satisfies the ℓ_2 -robust null space property of order k with constants $0 < \rho < 1$ and $\tau > 0$. Then, for any $\mathbf{x} \in \mathbb{C}^d$, the vector

$$\tilde{\mathbf{x}} := \arg \min_{\mathbf{z} \in \mathbb{C}^d} \|\mathbf{z}\|_1 \quad \text{subject to} \quad \|C\mathbf{z} - \mathbf{y}\|_2 \leq \eta,$$

where $\mathbf{y} := C\mathbf{x} + \mathbf{e}$ for some $\mathbf{e} \in \mathbb{C}^{\tilde{d}}$ with $\|\mathbf{e}\|_2 \leq \eta$, will satisfy

$$\|\mathbf{x} - \tilde{\mathbf{x}}\|_2 \leq \frac{C}{\sqrt{k}} \cdot \left(\inf_{\mathbf{z} \in \mathbb{C}^d, \|\mathbf{z}\|_0 \leq k} \|\mathbf{x} - \mathbf{z}\|_1 \right) + A\eta$$

for some constants $C, A \in \mathbb{R}^+$ that only depend on ρ and τ .

Error Guarantee

Theorem (Iwen, V., Wang 2014)

Let $\mathcal{P} \in \mathbb{C}^{D \times \tilde{d}}$ have its D rows be independently drawn either uniformly at random from the sphere of radius $\sqrt{\tilde{d}}$ in $\mathbb{C}^{\tilde{d}}$, or else as complex normal random vectors from $\mathcal{CN}(0, \mathcal{I}_{\tilde{d}})$.

Furthermore, suppose that $\mathcal{C} \in \mathbb{C}^{\tilde{d} \times d}$ satisfies the ℓ_2 -robust null space property of order k with constants $0 < \rho < 1$ and $\tau > 0$.

Then,

$$\min_{\theta \in [0, 2\pi]} \left\| e^{i\theta} \mathbf{x} - \Delta_{\mathcal{C}}(\Phi_{\mathcal{P}}(\mathbf{b})) \right\|_2 \leq \frac{C}{\sqrt{k}} \cdot \left(\inf_{\mathbf{z} \in \mathbb{C}^d, \|\mathbf{z}\|_0 \leq k} \|\mathbf{x} - \mathbf{z}\|_1 \right) + A \cdot \min \left(\|\mathcal{C}\mathbf{x}\|_2, \frac{\|\mathbf{n}\|_1}{D \|\mathcal{C}\mathbf{x}\|_2} \right)$$

holds for all $\mathbf{x} \in \mathbb{C}^d$ with probability $1 - \mathcal{O}(e^{-BD})$, provided that $D \geq E \cdot \tilde{d}$. Here $B, E \in \mathbb{R}^+$ are universal constants, while $C, A \in \mathbb{R}^+$ are constants that only depend on ρ and τ .

Error Guarantee

When \mathcal{C} is a random matrix with i.i.d. subGaussian random entries, we can further show that

$$\min_{\theta \in [0, 2\pi]} \left\| e^{i\theta} \mathbf{x} - \Delta_{\mathcal{C}}(\Phi_{\mathcal{P}}(\mathbf{b})) \right\|_2 \leq \frac{C}{\sqrt{k}} \cdot \left(\inf_{\mathbf{z} \in \mathbb{C}^d, \|\mathbf{z}\|_0 \leq k} \|\mathbf{x} - \mathbf{z}\|_1 \right) + A \|\mathbf{n}\|_2$$

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.

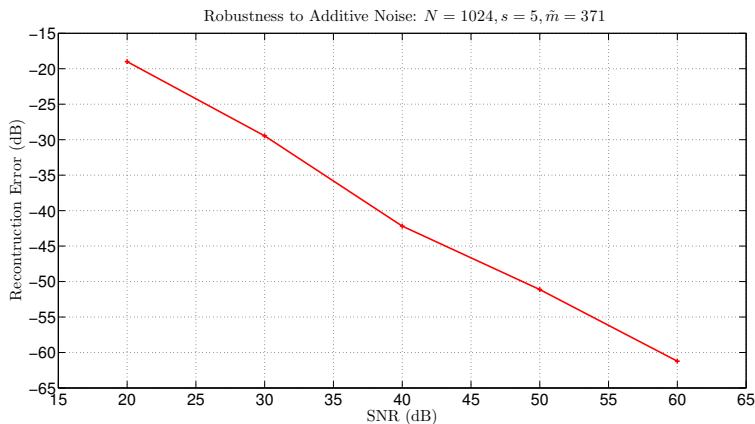
Numerical Results

- Test signals – sparse, unit-norm complex vectors
 - non-zero indices are independently and randomly chosen
 - non-zero entries are i.i.d. standard complex Gaussians
- Noise model – i.i.d. zero-mean additive Gaussian noise at different SNRs
- Errors reported as SNR (dB)

$$\text{Error (dB)} = 10 \log_{10} \left(\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \right)$$

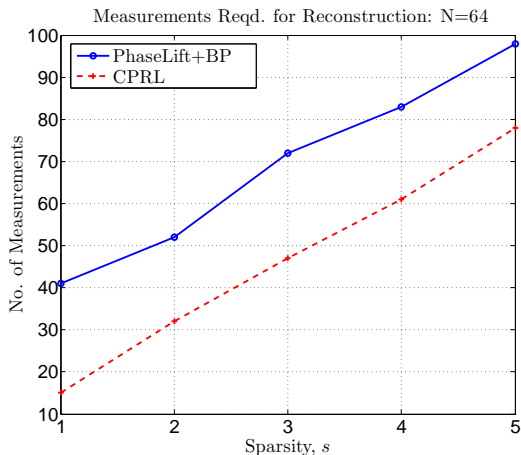
($\hat{\mathbf{x}}$ – recovered signal, \mathbf{x} – true signal)

Robustness



- Signal size: $d = 1024$
- Sparsity: $k = 5$
- 371 measurements ($14k \log(d/k)$)

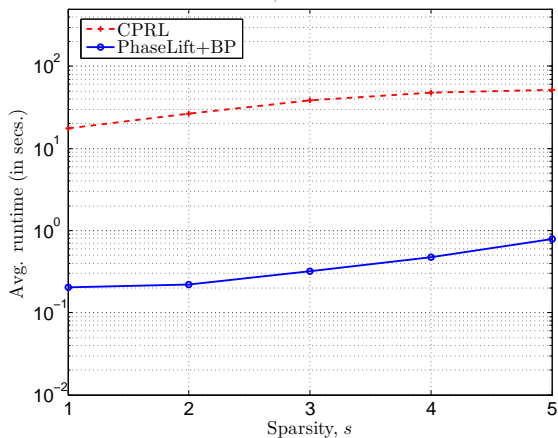
No. of measurements – Comparison with SDP-based CPRL



- $d = 64$, Noiseless measurements
- No. of measurements required for successful (relative ℓ_2 -norm error $\leq 10^{-5}$) reconstruction

Corresponding Runtime

Runtime: $N=64$, Noiseless Measurements



- $d = 64$
- Noiseless measurements
- Averaged over 100 trials

Discussion

(+)

- Requires $O(k \log(d/k))$ measurements.
- Significantly faster than comparable (SDP-based) methods.
- Recovery guarantee

(-)

- May require more (a small linear factor) measurements for small problems.

Summary

- Robust, efficient (FFT-time) phase retrieval algorithm
- Uses compactly supported masks and a block circulant construction in conjunction with angular synchronization
- Deterministic, well conditioned measurements masks
- Simple 2-stage method for sparse signals
- First sublinear time compressive phase retrieval algorithm.

References – Phase Retrieval

- ① J.R. Fienup. *Reconstruction of an object from the modulus of its Fourier transform*. Optics Letters, 3:27–29, 1978.
- ② 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.
- ③ B. Alexeev, A. S. Bandeira, M. Fickus, and D. G. Mixon. *Phase retrieval with polarization*. SIAM J. Imag. Sci., 7(1), 35–66.
- ④ M. Iwen, A. Viswanathan, and Y. Wang. *Fast Phase Retrieval for High-Dimensions*. arXiv:1501.02377, 2015.

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 page layout includes a left sidebar with navigation options like 'Overview', 'Source', 'Commits', 'Branches', 'Pull requests', 'Issues', 'Wiki', and 'Downloads'. The main content area is titled 'Overview' and displays repository statistics: 1 Branch, 1 Tag, 0 Forks, and 2 Watchers. It also features an 'Invite users to this repo' section with a 'Send invitation' button. Below this, the 'Recent activity' section lists several commits pushed by Aditya Viswanathan, including one that fixed typos in the README and another that added TFOCS for later use. The repository description states it contains Matlab code for solving the phase retrieval problem, with details of the method and theoretical guarantees available in the README. It also mentions the software was developed at Michigan State University and is released under the MIT license. The software uses Matlab R2014a and the TFOCS package for convex optimization problems.

charms / blockPR

Overview

SSH- git@bitbucket.org:charms/blockpr.git

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

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
c0916ac Added TFOCS for later use with ...
Aditya Viswanathan · 2015-01-09
- 1 commit
Pushed to charms/blockPR
cb8f96f Misc. format edits; Added Alt. Pro...
Aditya Viswanathan · 2015-01-09
- 12 commits
Pushed to charms/blockPR
bb6756b Merged in complex (pull request ...

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

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 repository name 'charms/sparsepr' is displayed at the top left. Below the name is a sidebar with 'ACTIONS' (Clone, Create branch, Create pull request, Compare, Fork) and 'NAVIGATION' (Overview, Source, Commits, Branches, Pull requests, Downloads, Settings). The main content area is titled 'Overview' and features a summary table with statistics: Last updated (2014-11-08), Language (MATLAB), Access level (Admin (revoke)), 2 Branches, 0 Tags, 0 Forks, and 2 Watchers. Below the table is an 'Edit README' link. The repository title is 'SparsePR: Matlab Software for Sparse Phase Retrieval'. The description states: '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 by 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.' On the right side, there is a 'Recent activity' section showing three commits: 'Pushed to charms/sparsepr' by Aditya Viswanathan (2014-11-08), 'Pushed to charms/sparsepr' by Aditya Viswanathan (2014-10-24), and 'Pushed to charms/sparsepr' by Aditya Viswanathan (2014-10-16). At the bottom of the activity list, it says 'Repository created' by Aditya Viswanathan (2014-10-15).

charms / sparsepr

Overview

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
a469988 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