Fast, sample-efficient algorithms for structured phase retrieval

Gauri Jagatap and Chinmay Hegde
Department of Electrical and Computer Engineering
IOWA STATE UNIVERSITY

Problem Setup

Premise: To devise a sample-efficient linear-convergence algorithm for phase retrieval of (structured) s-sparse signals from Gaussian measurements.

Main Challenges

- Linear-convergence algorithms have sample complexity with quadratic dependence on sparsity $m = \mathcal{O}\left(s^2 \log n\right)$.
- High number of tuning parameters.
- Lower sample complexity algorithms require high run time, not scalable.

Prior Work

- *Convex:* PhaseLift, PhaseMax. **Drawbacks**: Computationally expensive, poor emperical performance.
- *Non-convex:* AltMinPhase, Wirtinger Flow. Faster, scalable. **Drawbacks:** Sample complexity depends on selecting good initial point. Parametric inputs.
- *Designed sensing matrices:* Matrices with low underlying dimension or Fourier-like. **Drawbacks:** Harder to analyze theoretical guarantees.
- *Structured sparsity models:* Utilized for sparse signal recovery, statistical learning applications. **Drawbacks:** No rigorous results for phase retrieval problem.

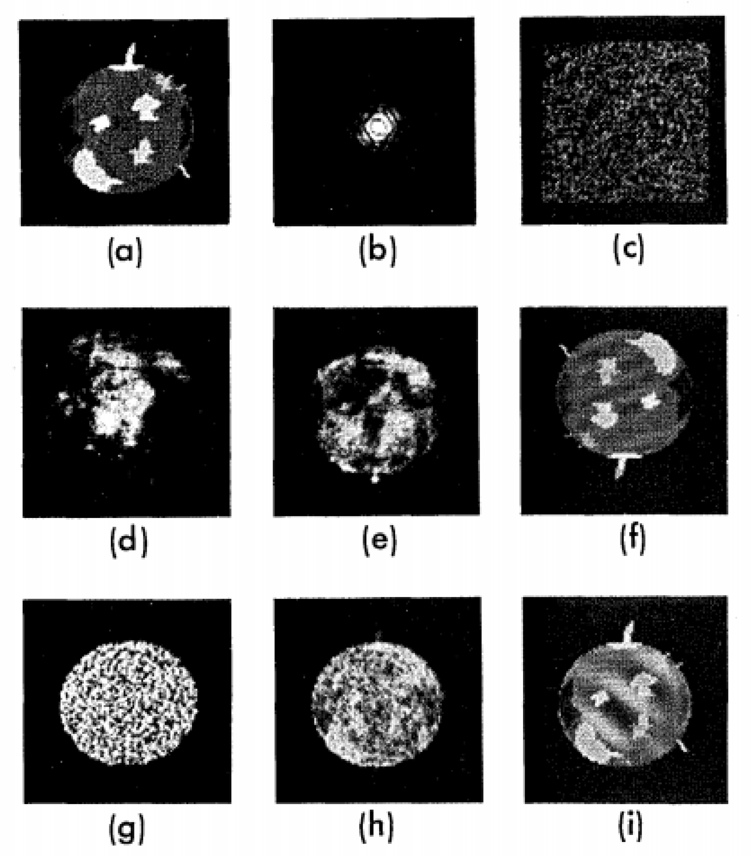


Fig. (a) Test object; (b) modulus of its Fourier transform; (c) initial estimate of the object (first test); (d)–(f) reconstruction results—number of iterations: (d) 20, (e) 230, (f) 600; (g) initial estimate of the object (second test); (h)–(i) reconstruction results—number of iterations: (h) 2, (i) 215.

Reproduced from [F78] J. Fienup, "Reconstruction of an object from the modulus of its Fourier transform." Optics letters, 1978.

Our Objective

We devise a phase-retrieval algorithm that:

- Utilizes underlying **structured sparsity** in signals for efficient analysis.
- Is **naturally compatible** with standard sparse recovery algorithms.
- Is **fast and scalable** to large datasets of large dimensions.
- Has **sub-quadratic** sample complexity $m = \mathcal{O}\left(\frac{s^2}{b} \log n\right)$.
- Requires **no extra parametric inputs** apart from (block) sparsity $k = \frac{s}{b}$.

New Direction: Phase retrieval of structured sparse signals

Idea: Efficient phase retrieval algorithms for structured sparse signals.

Problem Setup: Recover signal $\mathbf{x}^* \in \mathbb{R}^n$, using Gaussian sampling matrix $\mathbf{A} = [\mathbf{a_1} \dots \mathbf{a_m}]^\top$, from measurements $\mathbf{y} \in \mathbb{R}^m$,

$$y_i = |\langle \mathbf{a_i}, \mathbf{x}^* \rangle|, \quad \text{for } i = 1, \dots, m.$$

$\mathbf{x}^*$ is part of model $\mathcal{M}_{s,b}$ formed of uniformly block sparse signals with block length b, effective block sparsity $k = s/b$ and total number of blocks $n_b = n/b$.

Solution Methodology

Formulate the above as a two-step problem, by introducing diagonal phase matrix $\mathbf{P} \in \mathcal{P}$ with $P_{ii} = \text{sign}\left(\mathbf{a}_i{}^\top \mathbf{x}\right) \in \{1, -1\}$, and alternatively minimize the loss function over variables $\mathbf{x}$ and $\mathbf{P}$:

$$\min_{\mathbf{x} \in \mathcal{M}, \mathbf{P} \in \mathcal{P}} \|\mathbf{Ax} - \mathbf{Py}\|_2 .$$

This strategy requires a good initial point, to converge to minimum. For this, we introduce our algorithm Block Compressive Phase Retrieval with Alternating Minimization (Block CoPRAM) [JH17].

Block CoPRAM - Smart Initialization

Setup: Define signal marginals as $M_{jj} = \frac{1}{m}\sum_{i=1}^{n} y_i^2 a_{ij}^2$, for $j \in \{1 \dots n\}$.

Objective: Find good initial estimate $\mathbf{x^0}$ of the true signal $\mathbf{x}^*$.

Challenge: Designing block marginals. Performance guarantees.

Block CoPRAM - Smart Initialization (Cont.)

Solution: (Alg. 1)

- Define block marginals as $M_{j_b j_b} = \sqrt{\sum_{j \in j_b} M_{jj}^2}$, for $j_b \in \{1 \dots n_b\}$.
- Retain top k block marginals; call the index set $\hat{S}$, card$(\hat{S}) = s$.
- Construct matrix $(\in \mathbb{R}^{s \times s})$ $\mathbf{M}_{\hat{S}} = \frac{1}{m}\sum_{i=1}^{m} y_i^2 \mathbf{a}_{i\hat{S}} \mathbf{a}_{i\hat{S}}^\top$.
- Initial estimate $\mathbf{x^0} = \phi \mathbf{v}$, $\mathbf{v}$ is top-singular-vec$(\mathbf{M}_{\hat{S}})$, where $\phi = \sqrt{\frac{1}{m}\sum_{i=1}^{m} y_i^2}$.

Guarantees: Theorem 1

The initial vector $\mathbf{x^0}$, which is the output of Alg. 1, is a small constant distance δ_b away from the true signal $\mathbf{x}^*$, i.e.,

$$\text{dist}\left(\mathbf{x^0}, \mathbf{x}^*\right) \leq \delta_b \|\mathbf{x}^*\|_2 ,$$

where $0 < \delta_b < 1$, as long as the number of measurements satisfy $m \geq C\frac{s^2}{b} \log mn$ with probability greater than $1 - \frac{8}{m}$.

Block CoPRAM - Descent

Setup: Use the initialization $\mathbf{x^0}$ from Alg. 1.

Objective: Gradually descend to k-block sparse solution $\mathbf{x}^*$.

Challenge: Performance guarantees for convergence.

Solution: (Alg. 2)

Use alternating minimization with model CoSAMP to solve the non-convex problem:

- Phase estimation: $\mathbf{P}^t = \text{diag}\left(\text{sign}\left(\mathbf{Ax}^t\right)\right)$.
- Signal estimation: $\mathbf{x}^t \approx \min_{\mathbf{x} \in \mathcal{M}_{s,b}} \|\mathbf{Ax} - \mathbf{Py}\|_2$ via model CoSAMP.

Guarantees: Theorem 2

Given an initialization $\mathbf{x^0}$ satisfying Alg. 1, if number of measurements $m \geq C\left(s + \frac{sb}{n}\log\frac{n}{s}\right)$, then the iterates of Alg. 2 satisfy:

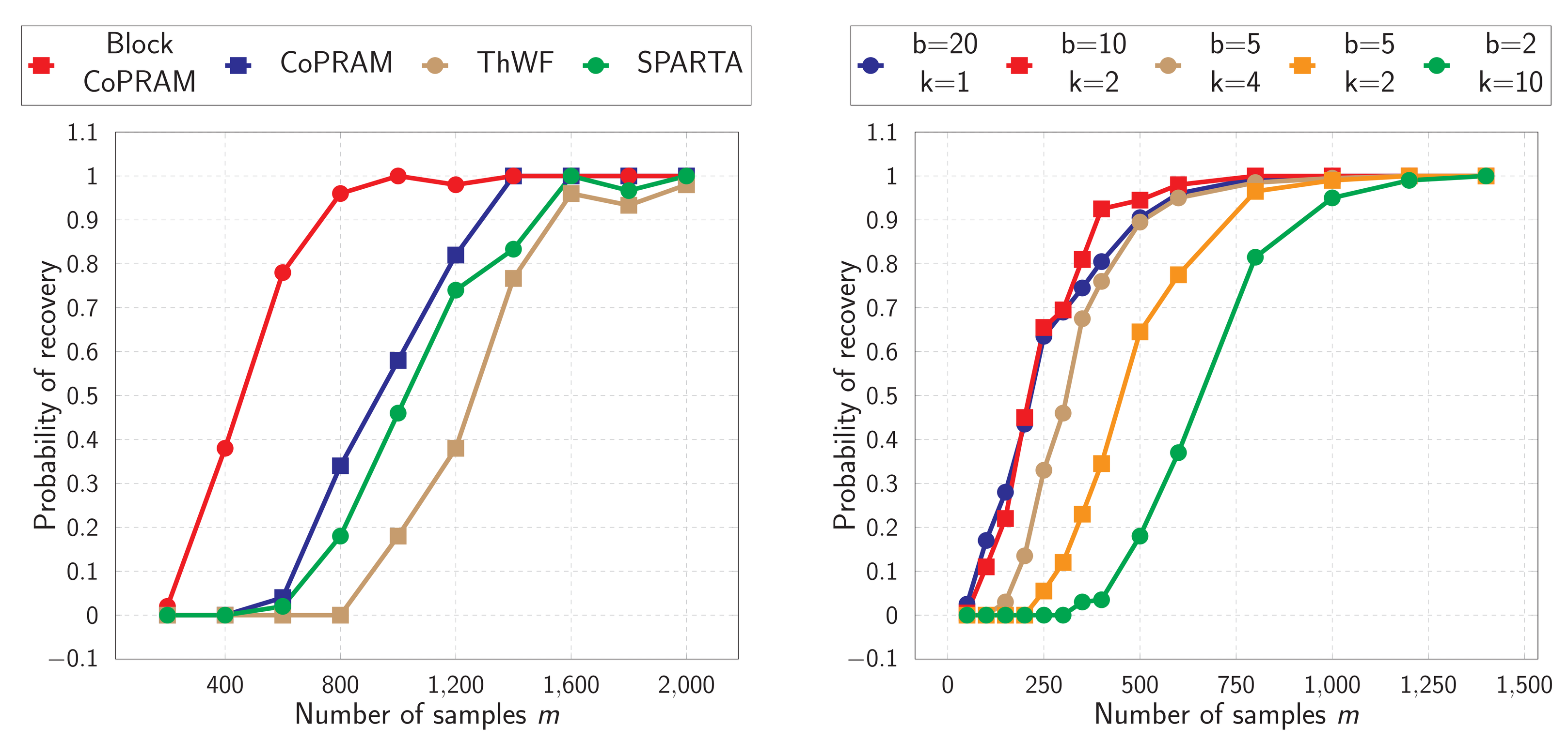
$$\text{dist}\left(\mathbf{x}^{t+1}, \mathbf{x}^*\right) \leq \rho_b \text{dist}\left(\mathbf{x}^t, \mathbf{x}^*\right).$$

where $0 < \rho_b < 1$ is a constant, with probability greater than $1 - e^{-\gamma m}$, for positive constant γ.

Results

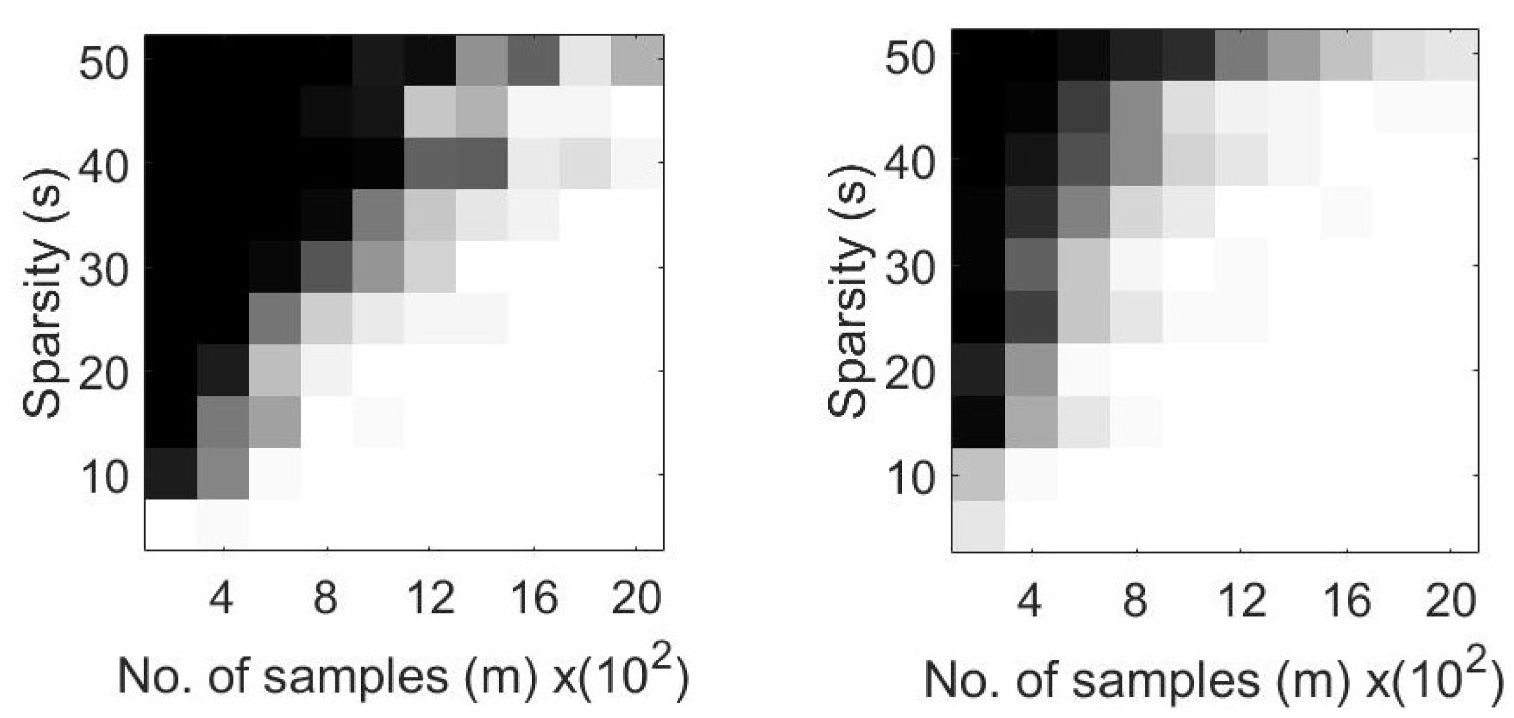
Phase transitions: Block CoPRAM v/s CoPRAM, SPARTA, Thresholded Wirtinger Flow (ThWF).

CoPRAM is a special case of Block CoPRAM with $b = 1$.



(a) Sparsity $s = 30$ and block length $b = 5$. (b) Sparsity $s = 20$ and different block lengths (Block CoPRAM only).

Figure 1: Phase transitions for signal length $n = 3,000$.



(a) CoPRAM (b) Block CoPRAM

Figure 2: Phase transition for signal length $n = 3,000$ and block length $b = 5$ and different sparsity levels.

Acknowledgments

This work was supported in part by grants from the National Science Foundation and NVIDIA.