

Conjugate Gradient Iterative Hard Thresholding: Observed Noise Stability for Compressed Sensing

Jeffrey D. Blanchard

Dept. of Mathematics and Statistics, Grinnell College
Grinnell, Iowa, USA 50112
Email: jeff@math.grinnell.edu

Jared Tanner and Ke Wei

Mathematics Institute, University of Oxford
Radcliffe Observatory Quarter
Oxford OX2 6GG, UK

Abstract—Conjugate Gradient Iterative Hard Thresholding (CGIHT) for compressed sensing combines the low per iteration complexity of fast greedy sparse approximation algorithms with the improved convergence rates of more complicated, projection based algorithms. This article shows that CGIHT is robust to additive noise and is typically the fastest greedy algorithm in the presence of noise.

I. INTRODUCTION

Many greedy algorithms have been proposed for finding a sparse solution to the underdetermined, linear system of equations in compressed sensing [1], [2]. Iterative greedy algorithms attempt to identify the solution to the combinatorial optimization problem

$$\hat{x} = \arg \min_{z \in \mathbb{R}^n} \|z\|_0 \quad \text{subject to} \quad y = Az \quad (1)$$

where $y = Ax + e \in \mathbb{R}^m$ is the vector of linear measurements of the k -sparse vector $x \in \mathbb{R}^n$ obtained by the matrix $A \in \mathbb{R}^{m \times n}$ with $e \in \mathbb{R}^m$ representing model misfit between representing y with k columns of A and/or additive noise. The vector x is k -sparse when the number of nonzeros entries in x is at most k , denoted $\|x\|_0 \leq k$.

Many greedy algorithms have recovery guarantees (including stability to noise) based on the now ubiquitous *restricted isometry property* (RIP) [3]. The matrix $A \in \mathbb{R}^{m \times n}$ satisfies a RIP of order ck if the lower and upper *restricted isometry constants* L_{ck} and U_{ck} are adequately bounded. The restricted isometry constants are defined as the smallest values which satisfy

$$(1 - L_{ck})\|z\|_2^2 \leq \|Az\|_2^2 \leq (1 + U_{ck})\|z\|_2^2 \quad (2)$$

for all ck -sparse vectors $z \in \mathbb{R}^n$. Leading greedy algorithms with sufficient RIP conditions for uniform recovery guarantees include Normalised Iterative Hard Thresholding (NIHT) [4], Hard Thresholding Pursuit (HTP) [5], Compressive Sampling Matching Pursuit (CoSaMP) [6], and Subspace Pursuit (SP) [7]. The authors introducing these algorithms, as well as other numerical investigations [8], [9], have shown that the RIP-based sufficient conditions establishing uniform recovery over all x are much more pessimistic than the observed recovery ability of the algorithms when a single x is drawn from a random vector ensemble.

Algorithm 1 CGIHT for Compressed Sensing

Initialization: Set $l = 1$, $T_{-1} = \{\}$, $w_{-1} = A^*y$,

$T_0 = \text{PrincipalSupport}_k(|w_{-1}|)$, $x_0 = \text{Proj}_{T_0}(w_{-1})$.

Iteration: During iteration l , **do**

1: $r_{l-1} = A^*(y - Ax_{l-1})$ (compute the residual)
(set orthogonalization weight)

2: **if** restarting **then**

3: **if** $T_{l-1} \neq T_{l-2}$ **then**

4: $\beta_{l-1} = 0$

5: **else**

6: $\beta_{l-1} = \frac{\|\text{Proj}_{T_{l-1}} r_{l-1}\|_2^2}{\|\text{Proj}_{T_{l-1}} r_{l-2}\|_2^2}$

7: **end if**

8: **else**

9: **if** $l = 1$, $\beta_0 = 0$, **else**

10: $\beta_{l-1} = -\frac{\langle A(\text{Proj}_{T_{l-1}}(r_{l-1})), A(\text{Proj}_{T_{l-1}}(p_{l-2})) \rangle}{\langle A(\text{Proj}_{T_{l-1}}(p_{l-2})), A(\text{Proj}_{T_{l-1}}(p_{l-2})) \rangle}$

11: **end if**

12: $p_{l-1} = r_{l-1} + \beta_{l-1}p_{l-2}$ (define search direction)
(compute step size)

13: **if** restarting **then**

14: $\alpha_{l-1} = \frac{\|\text{Proj}_{T_{l-1}}(r_{l-1})\|_2^2}{\|A(\text{Proj}_{T_{l-1}}(p_{l-1}))\|_2^2}$

15: **else**

16: $\alpha_{l-1} = \frac{\langle \text{Proj}_{T_{l-1}}(r_{l-1}), \text{Proj}_{T_{l-1}}(p_{l-1}) \rangle}{\langle A(\text{Proj}_{T_{l-1}}(p_{l-1})), A(\text{Proj}_{T_{l-1}}(p_{l-1})) \rangle}$

17: **end if**

18: $w_{l-1} = x_{l-1} + \alpha_{l-1}p_{l-1}$ (update along search direction)
(support set of k largest entries)

19: $T_l = \text{PrincipalSupport}_k(|w_{l-1}|)$

20: $x_l = \text{Proj}_{T_l}((w_{l-1}))$ (restriction to support set T_l)

II. CGIHT FOR COMPRESSED SENSING

The relative empirical performance of NIHT, HTP, and an algorithm denoted CSMPSP (combining CoSaMP and SP) were extensively tested in [9] using the software *GAGA: GPU Accelerated Greedy Algorithms for Compressed Sensing* [10], [11]. In particular, [9] presents *algorithm selection maps* which determine the fastest algorithm in the region of the $(\delta, \rho) = (m/n, k/m)$ phase space where at least one algorithm typically recovers a k -sparse vector from (y, A) . When multiple algorithms are able to recover the sparse vector, the

simplest algorithm generally provides the fastest solution; in fact, for three different matrix ensembles NIHT was the fastest algorithm throughout the majority of the phase space.

The lower per iteration complexity of NIHT coupled with its flexible option to change the subspace in which it is searching leads to the improved speed for recovery. However, the projections inherent in the iterations of HTP and CSMSPSP provide a substantially improved convergence rate¹ once the supporting subspace (or support set) is identified. In [12], we present Conjugate Gradient Iterative Hard Thresholding (Alg. 1) which combines the two advantages into a single algorithm². By allowing the support set to change after each iteration, CGIHT is able to rapidly identify the correct support while exploiting the computational advantages of a subspace restricted conjugate gradient method when the proxy for the support set remains unchanged. This enables CGIHT to quickly capture the correct values on the support while updating the support proxy in order to identify the correct support. Once the true support is found, CGIHT shares the same convergence rate as HTP and CSMSPSP.

In iteration l , NIHT computes a step size that is the optimal steepest descent step if the true support T is contained in the current proxy for the support T_{l-1} . Similarly, CGIHT computes an orthogonalization weight, search direction, and step size that executes a subspace restricted conjugate gradient (CG) method when the true support is contained in the current support proxy and the support proxy remains unchanged. Since orthogonality is only preserved if the initial step on a given support proxy is the steepest descent step, it is natural to restart the subspace restricted conjugate gradient search when the support changes. Also essential for low per iteration complexity, implementing CGIHT with restarting permits one to employ the lower cost implementation of CG. While the orthogonalization is lost without restarting, a non-restarted version of CGIHT frequently identifies the true support set in fewer iterations than restarting. In Alg. 1, the pseudo code identifies the differences in computing the orthogonalization weight and step size for both CGIHT with restarting (CGI-HTrestarted) and without restarting (CGIHT).

CGIHT with restarting also has a guaranteed stability to noise under and appropriate RIP condition.

Theorem 1 (CGIHT [12]): Let A be an $m \times n$ matrix with $m < n$, and $y = Ax + e$ for any x with at most k nonzeros. If the restricted isometry constants of A satisfy

$$\frac{(L_{3k} + U_{3k})(5 - 2L_k + 3U_k)}{(1 - L_k)^2} < 1, \quad (3)$$

then there exist constants $0 < \mu < 1$ and $\kappa, \xi > 0$ depending only on the restricted isometry constants of A such that

$$\|x_l - x\|_2 \leq \kappa(\mu^l) \|x_0 - x\|_2 + \frac{\xi}{1 - \mu} \|e\|_2 \quad (4)$$

where x_l is the l^{th} iterate of CGIHT with restarting.

¹The projection in HTP and CSMSPSP is implemented as a subspace restricted conjugate gradient method.

²CGIHT for matrix completion is also presented in [12]

CGIHT's sufficient RIP condition is worse than the known conditions for NIHT, HTP, and CoSaMP; as stated previously, it is well known that such uniform recovery conditions are not indicative of the often more important average case performance of these algorithms.

In this paper, we explore the empirical recovery performance of CGIHT in the presence of noise corrupted signals. Utilizing the GAGA software [11], we identify the empirical recovery phase transitions for CGIHT under multiple noise levels (Sec. IV-A). Furthermore, we compare the algorithm with NIHT, CSMSPSP, and Fast IHT (FIHT) based on Cevhar and Kyrillidis's 1-ALPS(2) which tailors Nesterov's steepest descent acceleration to the sparse approximation problem [13]. In Section IV-B we present algorithm selection maps of the phase space which identify the fastest algorithm. This can be interpreted as an expansion of the work in [9] which includes CGIHT and FIHT. The two main claims are:

- CGIHT is stable in the presence of additive noise;
- typically, CGIHT accurately approximates a sparse vector from noise corrupted measurements in less time than other greedy algorithms.

III. EMPIRICAL TESTING ENVIRONMENT

The testing is conducted with the algorithm implementations and testing suites in the software GAGA [11]. The testing was conducted on a Linux machine with Intel XEON E5-2643 CPUs @ 3.30 GHz, NVIDIA Tesla K10 GPUs, and executed from Matlab R2013a. The algorithms are tested with three random matrix ensembles:

- \mathcal{N} : dense Gaussian matrices with entries drawn i.i.d. from $\mathcal{N}(0, m^{-1})$;
- \mathcal{S}_7 : sparse matrices with 7 nonzero entries per column drawn with equal probability from $\{-1/\sqrt{7}, 1/\sqrt{7}\}$ and locations in each column chosen uniformly;
- DCT : m rows chosen uniformly from the $n \times n$ Discrete Cosine Transform matrix.

These three random matrix ensembles are representative of the random matrices frequently encountered in compressed sensing: \mathcal{N} represents dense matrices, \mathcal{S}_7 represents sparse matrices, and DCT represents subsampling structured matrices with fast matrix-vector products.

The measured vectors are taken from the random binary vector distribution, B , and are formed by uniformly selecting k locations for nonzero entries with values $\{-1, 1\}$ chosen with equal probability. For noisy measurements, we denote the sparse vector distribution by B_ϵ . In B_ϵ , the vectors are drawn from the sparse vector ensemble B and the measurements are corrupted by non adversarial, additive noise with noise level ϵ : $y = Ax + e$ for e drawn from $\mathcal{N}(0, \epsilon \|Ax\|_2)$. A problem class consists of a matrix ensemble and a sparse vector ensemble and is denoted (Mat, vec) with $Mat \in \{\mathcal{N}, \mathcal{S}_7, DCT\}$ and $vec \in \{B, B_\epsilon\}$ with $\epsilon \in \{0.1, 0.2\}$. Clearly, the problem class B_ϵ with $\epsilon = 0$ is equivalent to the problem class B ; these notations are used interchangeably.

A standard theoretical stability result for an algorithm approximating x with the vector $\hat{x} = x^{l^*}$ from the measurements

$y = Ax + e$ has the form

$$\|\hat{x} - x\|_2 \leq \kappa(\mu^{l^*}) \|x_0 - x\|_2 + \frac{\xi}{1 - \mu} \|e\|_2$$

for constants κ , μ , and ξ . Note that $\|e\|_2 = \epsilon \|Ax\|_2 \leq \epsilon \|A_T\|_2 \|x\|_2$ where A_T is the submatrix formed by the columns of A with indices in $T = \text{supp}(x)$. In this empirical study, we measure successful approximation with the somewhat strict condition on the relative approximation error:

$$\frac{\|\hat{x} - x\|_2}{\|x\|_2} < .001 + 2\epsilon, \quad (5)$$

so that $\kappa(\mu^{l^*}) \frac{\|x_0 - x\|_2}{\|x\|_2} \lesssim .001$ and $\frac{\xi}{1 - \mu} \|A_T\|_2 \lesssim 2$.

As described in [9], the algorithms terminate when any one of several stopping criteria are met: the residual is small, the residual is no longer improving, the algorithm is diverging, the convergence rate is near 1, or a maximum number of iterations is met. Through empirical testing, it was observed that when noise is present in the measurements, CGIHT frequently cycles between support sets. A simple additional stopping criterion alleviates this problem: if the ℓ_2 norm of the residual is identical (to single precision) to the ℓ_2 norm of one of the previous 15 residuals, a cycle is declared and the algorithm terminates.

IV. OBSERVED NOISE STABILITY

In this paper, we infer an observed stability to noise for CGIHT which is on par with the stability of other greedy algorithms such as NIHT, CoSaMP, and HTP. From [9], the empirical performance of HTP is similar to NIHT. To avoid cluttering the figures, we omit HTP from the comparisons and compare the two versions of CGIHT with NIHT, CSMPSP, and FIHT. The pseudo code for each of these algorithms is included in [12]. Recall from (5) that we define \hat{x} to be a successful approximate reconstruction of the target vector x when $\frac{\|\hat{x} - x\|_2}{\|x\|_2} < .001 + 2\epsilon$, where $\epsilon \in \{0, 0.1, 0.2\}$ is the noise level.

A. Empirical Recovery Phase Transitions

For each problem class, the empirical recovery phase transitions are identified by first determining a phase transition region $[k_{\min}, k_{\max}]$ where an algorithm successfully recovers 10 of 10 random vectors at k_{\min} but recovers 0 of 10 random vectors at k_{\max} . The phase transition region is then extensively tested and the *recovery phase transition* is the curve defined by a logistic regression for a 50% success rate. The *recovery region* is the region below the recovery phase transition and identifies the region of the phase space where an algorithm typically recovers every (random) vector.

In Fig. 1, we observe that the recovery phase transition curves for CGIHT with and without restarting smoothly decrease as the noise level ϵ increases. Moreover, for the moderate noise level $\epsilon = 0.1$, the recovery phase transition curves for CGIHT closely track the recovery phase transitions for $\epsilon = 0$. Even for $\epsilon = 0.2$, the recovery region captures a significant portion of the noise free ($\epsilon = 0$) recovery region.

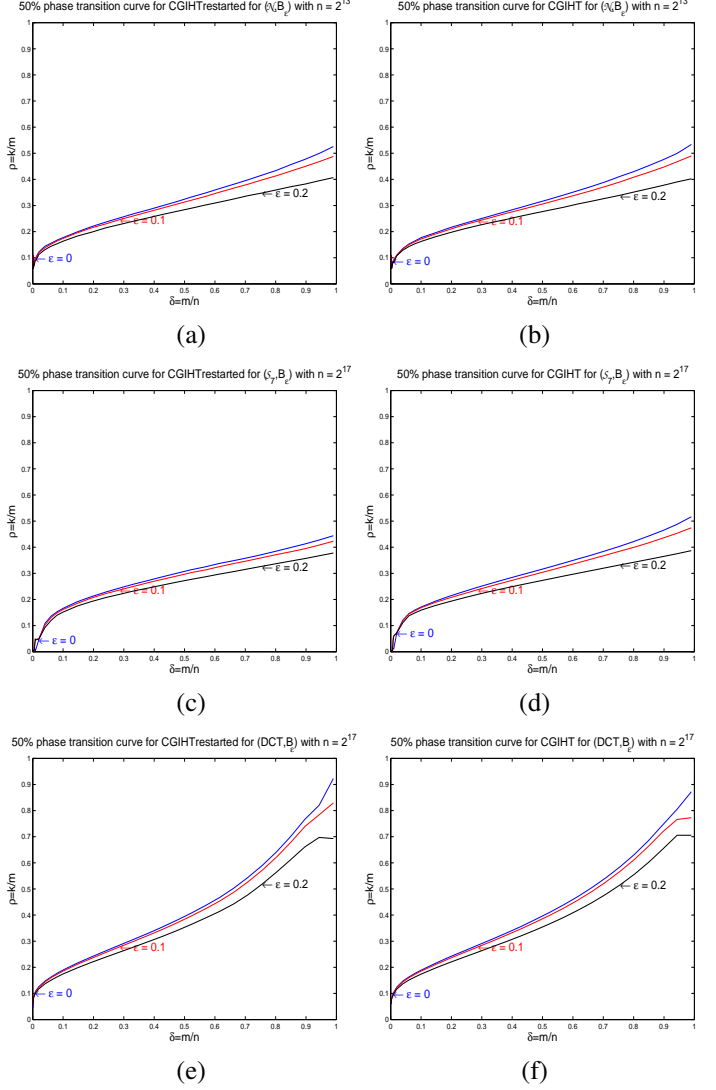


Fig. 1. 50% recovery probability logistic regression curves for problem classes (Mat, B_ϵ) with $\epsilon = 0, 0.1, 0.2$. Left column: CGIHT with restarting. Right column: CGIHT without restarting. (a,b) $Mat = \mathcal{N}$, $n = 2^{13}$; (c,d) $Mat = S_7$, $n = 2^{17}$; (e,f) $Mat = DCT$, $n = 2^{17}$.

Thus, we claim that through the majority of the noise-free recovery region, the relative ℓ_2 norm of the approximations from both the restarted and non-restarted variants of CGIHT scales with the noise level ϵ . This observation holds over three representative matrix ensembles. These empirical observations show that the average case recovery of CGIHT is substantially beyond the region where the uniform recovery condition of Thm. 1 is valid, and show that the recovery regions degrade slowly with additive noise.

In Fig. 2, we compare the empirical recovery phase transitions of CGIHT with and without restarting, CSMPSP, NIHT, and FIHT. In the most interesting region of the phase space for compressed sensing, namely $\delta = m/n < 0.5$, the recovery phase transitions for CGIHT are superior to NIHT and FIHT for all three matrix ensembles. For matrix ensembles \mathcal{N} and S_7 , CSMPSP boasts the highest (best) recovery phase transi-

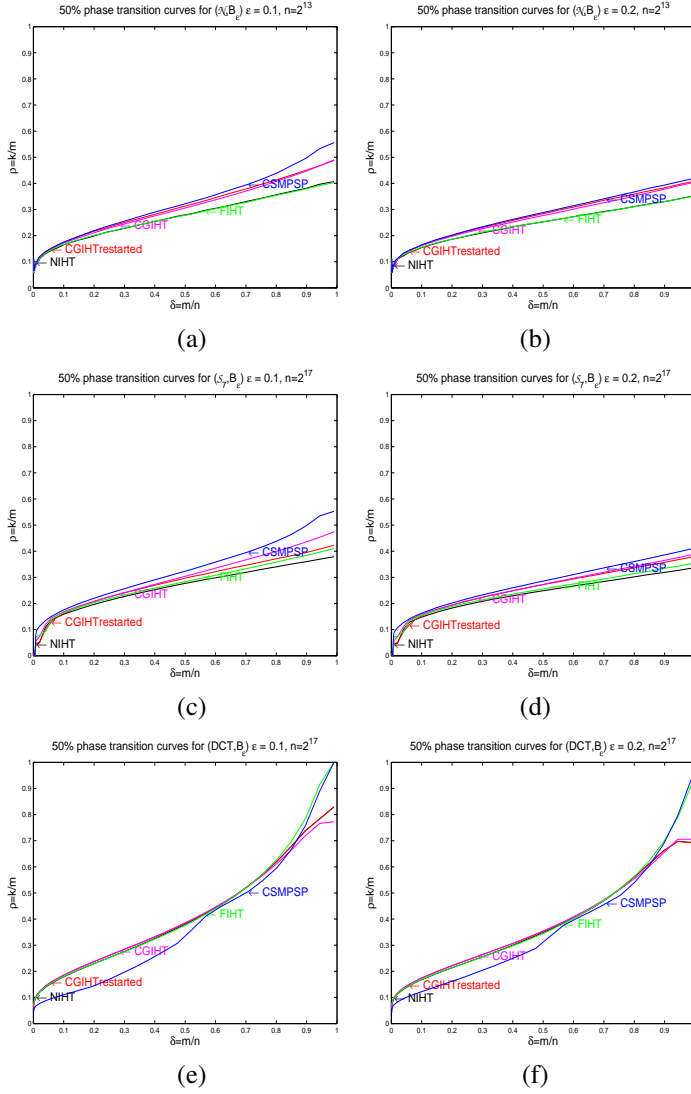


Fig. 2. 50% recovery probability logistic regression curves of CGIHT, NIHT, FIHT and CSMPSP for problem classes (Mat, B_ϵ) . Left column: $\epsilon = 0.1$. Right column: $\epsilon = 0.2$. (a,b) $Mat = \mathcal{N}$, $n = 2^{13}$; (c,d) $Mat = S_7$, $n = 2^{17}$; (e,f) $Mat = DCT$, $n = 2^{17}$.

tion for all values of $\delta = m/n$ with the advantage increasing as $\delta \rightarrow 1$. In contrast, for $\delta < 0.5$ the recovery phase transitions for both variants of CGIHT become increasingly similar to that of CSMPSP as δ decreases, except for matrix ensemble DCT where CSMPSP has a substantially lower phase transition. As the noise level ϵ increases, the differences between these phase transition curves decrease. In terms of location of the recovery phase transition, we claim the empirical recovery phase transition of CGIHT with and without restarting is competitive with the other algorithms.

The recovery phase transition curves from Figs. 1 and 2 show the location of the 50% success rate for each of the algorithms. These clearly delineate the recovery region for each algorithm and inform practitioners if an algorithm is likely to succeed in recovering a sparse vector. In the region of the phase space between the algorithm with the highest phase

transitions and the next highest phase transition curve, one should clearly employ the only successful algorithm. In the region of the phase space below the phase transition curves for multiple algorithms, one must choose which algorithm best serves their application.

B. Algorithm Selection Maps

When selecting from multiple algorithms, a natural choice is to select the algorithm with the fastest recovery time. In [9], the first two authors introduced algorithm selection maps which identify the algorithm with the fastest recovery time. In that empirical study, the dominant algorithms under consideration were NIHT, HTP, CSMPSP. While the noise-free algorithm selection maps included a mix of these algorithms, NIHT was the overwhelming selection in the presence of additive noise with $\epsilon = 0.1$. In this section, we extend the algorithm selection maps to consider CGIHT with and without restarting and FIHT.

In Fig. 3, we present algorithm selection maps for all three matrix ensembles and noise levels $\epsilon \in \{0, 0.1, 0.2\}$. In the noise-free setting (Fig. 3(a),(d),(g)) FIHT is often the fastest algorithm when $\rho = k/m < 0.2$ or $\delta = m/n < 0.1$. In the region where CSMPSP is the only successful algorithm, it is obviously also the fastest algorithm. For all three matrix ensembles, one of the two versions of CGIHT is often the fastest algorithm for problem instances (k, m, n) with $(\delta, \rho) = (k/m, m/n)$ just below the phase transition. This is a similar finding to the recovery behavior of NIHT for noiseless measurements presented in [9]. Essentially, for noiseless signals, CGIHT or FIHT replaces NIHT in the algorithm selection maps of all three matrix ensembles.

However, when the measurements are corrupted with additive noise, Fig. 3(b,c),(e,f),(h,i) show that CGIHT is almost always the fastest algorithm. This is similar to the findings for NIHT in [9]; it is clear from these updated selection maps that CGIHT is an improvement over NIHT. FIHT's noise sensitivity stems from reintroducing the noise in each iteration at the computation of the extrapolated point; this is the likely cause of its nearly complete absence from the algorithm selection maps with noisy measurements.

Interestingly, the choice of restarting CGIHT when the support set changes provides a performance gain for matrix ensembles \mathcal{N} and DCT but is clearly slower for the sparse matrices from S_7 . For the dense matrices from \mathcal{N} , CGIHT with restarting is the fastest algorithm under noise levels $\epsilon = 0.1, 0.2$ as $\delta \rightarrow 0$ and for all δ when $\rho < 0.2$. For $\delta > 0.2$, the non-restarted variant of CGIHT is fastest as ρ approaches the algorithm's recovery phase transition curve. For the problem class (DCT, B_ϵ) with $\epsilon = 0.1, 0.2$, CGIHT with restarting is the clear choice for $\delta < 0.4$. As the noise level increases, the advantage of CGIHT over NIHT for problem class (DCT, B_ϵ) is eliminated for large δ . On the other hand, for problem classes (S_7, B_ϵ) with $\epsilon \in \{0, 0.1, 0.2\}$, the non-restarted version of CGIHT is superior to the restarted version except for small values of ρ . Since the cost per iteration of the non-restarted version of CGIHT is often twice that of the

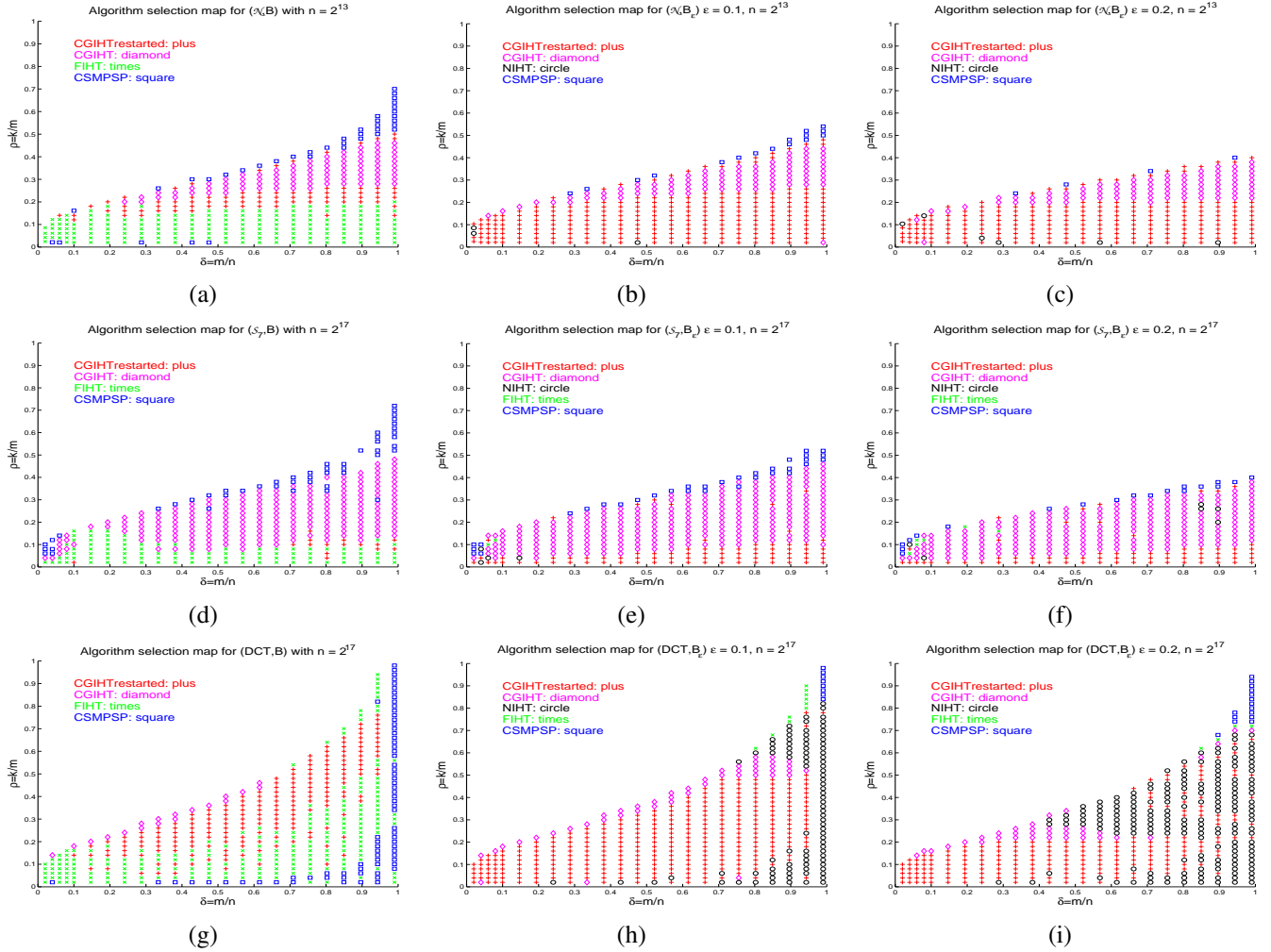


Fig. 3. Algorithm selection maps for problem classes (Mat, B_ϵ) with $\epsilon = 0, 0.1, 0.2$. Left column: $\epsilon = 0$. Center column: $\epsilon = 0.1$. Right column: $\epsilon = 0.2$. (a-c) $Mat = \mathcal{N}$, $n = 2^{13}$; (d-f) $Mat = S_7$, $n = 2^{17}$; (g-i) $Mat = DCT$, $n = 2^{17}$.

restarted version, it is clear that restarting CGIHT for problem class (S_7, B_ϵ) requires many more iterations to find the correct support set.

V. CONCLUSION

Under three different measurement matrix ensembles CGIHT exhibits a strong stability against additive noise. CGIHT's flexibility to easily change support sets after each iteration and the efficiency of the subspace restricted conjugate gradient search allows the algorithm to rapidly and accurately recover the measured vector. The phase space maps establish that CGIHT is typically the fastest greedy algorithm for compressed sensing under additive noise.

ACKNOWLEDGMENT

JDB was supported by NSF DMS 1112612.

REFERENCES

- [1] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [2] E. J. Candès, "Compressive sampling," in *Inter. Congress of Mathematicians. Vol. III*. Eur. Math. Soc., Zürich, 2006, pp. 1433–1452.
- [3] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Trans. Inform. Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.
- [4] T. Blumensath and M. E. Davies, "Normalised iterative hard thresholding: guaranteed stability and performance," *IEEE J. Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298–309, 2010.
- [5] S. Foucart, "Hard thresholding pursuit: an algorithm for compressive sensing," *SIAM J. Num. Anal.*, vol. 49, no. 6, pp. 2543–2563, 2011.
- [6] D. Needell and J. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, 2009.
- [7] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Inform. Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.
- [8] A. Maleki and D. Donoho, "Optimally tuned iterative reconstruction algorithms for compressed sensing," *IEEE J. Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 330–341, 2010.
- [9] J. D. Blanchard and J. Tanner, "Performance comparisons of greedy algorithms for compressed sensing," 2013, Submitted.
- [10] —, "GPU accelerated greedy algorithms for compressed sensing," *Math. Prog. Computation*, vol. 5, no. 3, pp. 267–304, 2013.
- [11] —, "GAGA: GPU Accelerated Greedy Algorithms," 2013, version 1.1.0. [Online]. Available: www.gaga4cs.org
- [12] J. D. Blanchard, J. Tanner, and K. Wei, "Conjugate gradient iterative hard thresholding for compressed sensing and matrix completion," 2013, submitted.
- [13] A. Kyrillidis and V. Cevher, "Recipes on hard thresholding methods," in *4th IEEE Inter. Workshop on CAMSAP*, 2011, pp. 353–356.