Optimally Tuned Iterative Reconstruction Algorithms for Compressed Sensing

Arian Maleki
Department of Electrical Engineering
Stanford University
arianm@stanford.edu

David L. Donoho
Department of Statistics
Stanford University
donoho@stanford.edu

Abstract— **We conducted an extensive computational experiment, lasting multiple CPU-years, to optimally select parameters for two important classes of algorithms for finding sparse solutions of underdetermined systems of linear equations. We make the optimally tuned implementations available at `sparselab.stanford.edu`; they run 'out of the box' with no user tuning: it is not necessary to select thresholds or know the likely degree of sparsity.**

Our class of algorithms includes iterative hard and soft thresholding with or without relaxation, as well as CoSaMP, subspace pursuit and some natural extensions. As a result, our optimally tuned algorithms dominate such proposals.

Our notion of optimality is defined in terms of phase transitions, i.e. we maximize the number of nonzeros at which the algorithm can successfully operate. We show that the phase transition is a well-defined quantity with our suite of random underdetermined linear systems. Our tuning gives the highest transition possible within each class of algorithms. We verify by extensive computation the robustness of our recommendations to the amplitude distribution of the nonzero coefficients as well as the matrix ensemble defining the underdetermined system.

Our findings include: (a) For all algorithms, the worst amplitude distribution for nonzeros is generally the constant-amplitude random-sign distribution, where all nonzeros are the same amplitude. (b) Various random matrix ensembles give the same phase transitions; random partial isometries may give different transitions and require different tuning; (c) Optimally tuned subspace pursuit dominates optimally tuned CoSaMP, particularly so when the system is almost square.

I. Introduction

A recent flood of publications offers numerous schemes for obtaining sparse solutions of underdetermined systems of linear equations; a long list of useful ideas and suggestions can be gleaned from the papers [1]-[29], with new proposals appearing regularly. Popular methods have been developed from many viewpoints: ℓ_1-minimization [1], [2], [3], [4], [5], [6], matching pursuit [7], [8], [9], [10], iterative thresholding methods [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], subspace methods [10], [23], [20], [24], convex regularization [25], [26] and nonconvex optimization [27], [28], [29]. The specific proposals are often tailored to different viewpoints, ranging from formal analysis of algorithmic properties [12], [9], [30], [10], [6], to particular application requirements [13], [14], [15]. Such algorithms have potential applications in fields ranging from medical imaging to astronomy [31], [32].

The potential user now has a bewildering variety of ideas and suggestions that might be helpful, but this, paradoxically creates uncertainty and may cause said potential user of such algorithms to avoid the topic entirely.

This paper develops a common framework for comparing properties of reconstruction algorithms in compressed sensing, and delivers specific, highly optimized recommendations implemented in freely-available software. It considers several popular iterative thresholding algorithms, and abstracts them into a few different types, each with tunable parameters. It defines a quantitative notion of performance and after a large-scale computational study, identifies a tuned version of each algorithm type offering the best performance guarantee across a universe of test suites. Performance is measured by the undersampling-sparsity tradeoff, or 'phase-transition curve', and for each algorithm type, we identify the optimally tuned instance that maximizes the worst-case tradeoff, yielding one recommended algorithm for each algorithm type.

Our tuning is based on a comprehensive study of parameter variations and options. It would have required several years to complete our study on a single modern desktop computer. Optimal tuning manages to make some very simple and unsexy ideas perform surprisingly well, reducing the need for more ambitious and impressive sounding ones (even if optimally tuned). Our tuning is based on quantitative principles; it can be used for other algorithms as well and implicitly establishes the 'current state of the art' which future proposals may be compared against. It also generates insights previously unavailable about performance comparisons of methods and performance comparisons of different matrix ensembles.

The empirical tuning approach has a larger significance for the field of sparse representations and compressed sensing. Many of the better known papers in this field discuss *what can be proved rigorously*, using mathematical analysis. It requires real mathematical maturity to understand what is being claimed and what the interpretation must be, and to compare claims in competing papers. Often, what can be proved is vague (with unspecified constants) or very weak (unrealistically strong conditions are assumed, far from what can be met in applications). For practical engineering applications it is important to know *what really happens* rather than what can be proved. Empirical studies provide a direct method to give engineers useful guidelines about what really does happen.

The empirical tuning approach also addresses a difficulty many potential users face in addressing the large and growing literature in sparse representations and compressed sensing. In that literature, a rich variety of brand names is developing, where small variations in some already well-known algorithmic schema may lead to the introduction of extravagant new acronyms and phrases. The outsider will be wary of investing the time to digest all this literature and form an accurate understanding of the differences. Empirical tuning efforts group together several differently-named ideas within one family of algorithmic schemas and optimize settings across the whole family, thereby simplifying the situation for many potential users, since the recommended algorithm both runs 'out of the box' and has been tuned to supersede several different earlier proposals.

II. Iterative Algorithms

Our problem setting will be described with the following notation. An unknown vector $x_0 \in \mathbb{R}^N$ is of interest; we have measurements $y = Ax_0$. Here A is an $n \times N$ matrix and $N > n$. Although the system is underdetermined, it has been shown that, when it exists,

sufficient *sparsity* of x_0 may allow unique identification of x_0. We say that x_0 is k-sparse if it has at most k nonzeros. In many cases one can exactly recover such a sparse solution x_0 as the solution to

$$(P_1) \qquad \min \|x\|_1 \text{ subject to } y = Ax,$$

where $\|x\|_1$ denotes the ℓ_1 norm. This amounts to a large-scale linear programming problem. Unfortunately in some interesting potential applications [33], [34], the matrix A and vector x_0 may contain millions of entries and standard linear programming codes may be too slow in those applications. Hence there is widespread interest in finding fast algorithms that work essentially as well; in particular application work by Starck and co-authors [13], [14], [35], [32] and by Elad and co-authors [15], [17] has shown that some very simple iterative algorithms can be strikingly successful on very large problems. In this note we consider two families of such iterative algorithms.

A. Simple Iterative Algorithms

The first family is inspired by the classical relaxation method for approximate solution of large linear systems. In classical relaxation, one iteratively applies A and its transpose A' to appropriate vectors and under appropriate conditions, the correct solution is obtained as a limit of the process. While the classical theory is inapplicable to underdetermined systems, it has been found both empirically and in theory that a sparsity-promoting variant of relaxation can correctly solve such systems, when they have sufficiently sparse solutions. Starting from $x_1 = 0$, one repeatedly applies this rule:

$$x_{i+1} = \eta_{t_i}(x_i + \kappa \cdot (A' r_i)); \qquad r_i = y - Ax_i;$$

Here κ is a relaxation parameter $(0 < \kappa < 1)$ and we assume throughout that A is normalized so that its columns have unit length. $\eta_t(\cdot)$ denotes a scalar nonlinearity, applied entrywise; we consider both *hard* thresholding – $\eta_t^H(y) = y1_{\{|y|>t\}}$ and *soft* thresholding $\eta_t^S(y) = \mathrm{sgn}(y)(|y| - t)_+$. In the above functions t is called the threshold value. Note that if we set $\eta(y) = y$ we would just have classical relaxation. Iterative Soft Thresholding (IST) with a fixed threshold has been used in various settings more than a decade ago – see for example published work of Sylvain Sardy and co-authors [11]. A formal convergence analysis was given by [12] in the determined case. Iterative Hard Thresholding (IHT) was reported useful for several underdetermined cases by Starck, Elad, and their co-authors in papers appearing as early as 2004, [13], [14], [15], [17], [35] – often outperforming IST. Other recent examples of such ideas include [16], [19], [18], [36], [22], [21].

These iterative schemes are easy to implement: they require only two matrix-vector products per iteration and some vector additions and subtractions. For certain very large matrices we can rapidly apply A and A' without representing A as a full matrix – examples include partial Fourier and Hadamard transforms. In such settings, the work required scales very favorably with N (e.g. $N \log(N)$ flops rather than $O(N^2)$).

Actually using such a scheme in practice requires choosing a parameter vector $\theta = (\text{type}, \kappa, t)$; here type = S or H depending as soft or hard thresholding is required; the other parameters are as earlier. Moreover the threshold value t needs to vary from iteration to iteration. The general form in which such schemes are often discussed does not give a true ready-to-run algorithm. This is akin to presenting a cooking recipe listing ingredients for a dish, without listing the needed amounts; it keeps potential users from successfully exploiting the idea.

B. Composite Iterative Algorithms

In solving determined linear systems, relaxation can often be outperformed by other methods. Because of the similarity of relaxation to IST/IHT schemes, parallel improvements seem worth pursuing in the sparsity setting. A more sophisticated scheme – Two Stage Thresholding (TST) – combines exact solution of small linear systems combined with thresholding before and after this solution. In stage one, we screen for 'significant' nonzeros just as in IST and IHT:

$$v_i = \eta_{t_{1i}}^{(1)}(x_i + \kappa A' r_i); \qquad r_i = y - Ax_i;$$

We let I_i denote the combined support of v_i and x_i and we solve

$$w_i = (A'_{I_i} A_{I_i})^{-1} A'_{I_i} y.$$

We then threshold a second time,

$$x_{i+1} = \eta_{t_{2i}}^{(2)}(w_i),$$

producing a sparse vector. Here the threshold t and even the nonlinearity $\eta^{(i)}$ might be chosen differently in stages 1 and 2 and might depend on the iteration and on measured signal properties. CoSaMP [20] and subspace pursuit [24] may be considered as special cases of TST. This will be clearer when we explain the threshold choice in the next section.

It seems that the use of explicit solutions to the smaller systems might yield improved performance, although at the cost of potentially much more expense per iteration. An important problem is user reticence. In the case of TST there are even more choices to be made than with IST/IHT. This scheme again presents the 'recipe ingredients without recipe amounts' obstacle: users may be turned off by the requirement to specify many such tunable parameters.

C. Threshold Choice

Effective choice of thresholds is a heavily-developed topic in statistical signal processing. We have focused on two families of tunable alternatives.

Interference heuristic. We pretend that the marginal histogram of $A'r$ at sites in the coefficient vector where $x_0(i) = 0$ is Gaussian, with common standard deviation σ. We robustly estimate the marginal standard deviation σ of the entries in $A'r$ at a given iteration and set the threshold t as a fixed multiple of that standard deviation – $t = \lambda \cdot \sigma$, where λ is our chosen threshold control parameter, typically in the range $2 < \lambda < 4$. The underlying rationale for this approach is explained in [10] where its correctness was heavily tested. Under this heuristic, we control the threshold λ as in standard detection theory using the False Alarm Rate (FAR); thus $FAR = 2 \cdot \Phi(-\lambda)$ where Φ denotes the standard normal distribution function.

Oracle heuristic. In the TST scheme, imagine that an oracle tells us the true underlying sparsity level k, and we scale the threshold adaptively at each iteration so that at stage 1 we yield $\alpha \cdot k$ nonzeros and at stage two $\beta \cdot k$ nonzeros. The method CoSaMP [20] corresponds to $\beta = 2$, $\alpha = 1$, while subspace pursuit [24] corresponds to $\beta = \alpha = 1$.

A problem with the oracle heuristic is that, in interesting applications, there is no such oracle, meaning that we wouldn't in practice ever know what k to use. A problem with the interference heuristic is that the Gaussian model may not work when the matrix is not really 'random'.

III. Phase Transitions

In the case of ℓ_1 minimization with A a random matrix, there is a well-defined 'breakdown point': ℓ_1 can successfully recover the

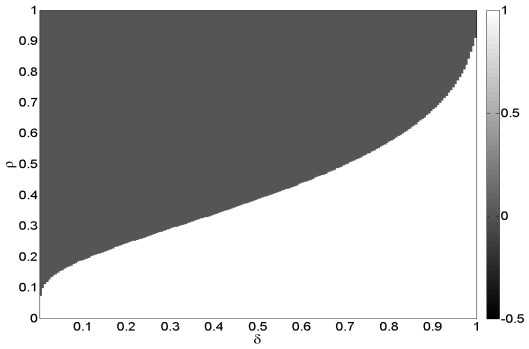

Fig. 1. *Phase Diagram for ℓ_1 minimization*. horizontal axis: indeterminacy $\delta = n/N$. vertical axis: sparsity $\rho = k/n$. Shaded attribute depicts limiting probability that ℓ_1 succesfully reconstructs the sparsest solution, as a function of the state variables (ρ, δ). Dark region: limiting probability 0. Light region: limiting probability is 1.

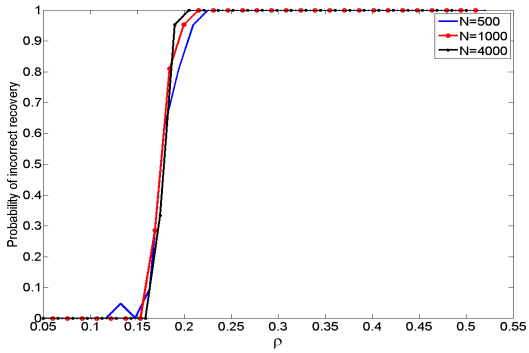

Fig. 2. *Finite-N Phase Transitions.* Fraction of unsuccessful recovery attempts by IHT, as a function of sparsity level ρ. Here $\delta = .5$ and $\rho = k/n$ is varying. Results are shown at 3 values of N: 500, 1000, 4000; note steepening of the transition as N increases, in accord with theory. FAR parameter $= 10^{-3}$. Relaxation parameter = 1. At each unique parameter combination, twenty random problem instances were tried.

sparsest solution provided k is smaller than a certain definite fraction of n.

Let $\delta = n/N$ be a normalized measure of problem indeterminacy and let $\rho = k/n$ be a normalized measure of the sparsity. We get a two-dimensional phase space $(\delta, \rho) \in [0,1]^2$ describing the difficulty of a problem instance – problems are intrinsically harder as one moves up and to the left. Displays indicating success and failure of ℓ_1 minimization as a function of position in phase space often have an interesting two-phase structure (as shown in figure 1), with phases separated by the curve $(\delta, \rho_{\ell_1}(\delta))$, for a specific function ρ_{ℓ_1}.

Let A be a random matrix with iid Gaussian entries and let $y = Ax_0$ with x_0 k-sparse. In [37], [38] one can find explicit formulas for a function ρ definable with the aid of polytope theory and having the following property. *Fix $\epsilon > 0$. The probability that (P_1) recovers the sparsest solution to $y = Ax$ tends to 0 or 1 with increasing system size according as $k \sim n \cdot (\rho_{\ell_1}(n/N) \pm \epsilon)$.* Informally, all that matters is whether $(n/N, k/n)$ lies above or below the curve $(\delta, \rho_{\ell_1}(\delta))$. This is the conclusion of a rigorously proven theorem that describes asymptotic properties as $N \to \infty$; it also describes what actually happens at finite problem sizes [39]. The empirically observed fraction of successful recoveries decays from one to zero as the problem sparsity $\rho = k/n$ varies from just below the critical level $\rho_{\ell_1}(\delta)$ specified in theory to just above it. This transition zone is observed to get increasingly narrow as N increases, matching the theorem, which says that in the large N limit, the zone has vanishing width.

Such sharp phase transitions have also been rigorously proven [40], [41] or empirically observed [10], [6], [42] for other algorithms and/or problem suites. Figure 2 displays behavior of IHT with FAR threshold selection, at a single fixed choice n/N with varying underlying number k of nonzeros. Below a certain threshold, the algorithm works well and above that threshold it fails; the transition zone is narrow, and gets better defined at large problem sizes N.

Incidentally, some readers may be unfamiliar with the notion of phase transitions because popular theoretical tools such as coherence [2], [3] and Restricted Isometry Property (RIP) [43] do not really give information about them. It has been shown by Tanner and co-authors [44] that bounds derived from RIP ensure the existence of a region with high success probability in the δ-ρ phase space; however, the actual region is much larger than what RIP bounds provide.

IV. Estimating the Empirical Phase Transition

For a given algorithm with a fully specified parameter vector θ, we conduct one phase transition measurement experiment as follows. We fix a *problem suite*, i.e. a matrix ensemble and a coefficient distribution for generating problem instances (A, x_0) and a measure of success; see below. For a fixed $N = 800$, we varied $n = \lceil \delta N \rceil$ and $k = \lceil \rho n \rceil$ through a grid of 900 δ and ρ combinations, with δ varying from .05 to 1 in 30 steps and ρ varying from .05 up to a ceiling value $\rho_{max} < 1$ in as many as 30 steps. We then have a grid of δ, ρ values in parameter space $[0,1]^2$. At each (δ, ρ) combination, we will take M problem instances and obtain M algorithm outputs $\hat{x}_i$; in our case $M = 100$. For each problem instance we declare success if

$$\frac{\|x_0 - \hat{x}_i\|_2}{\|x_0\|_2} \leq \text{tol},$$

where tol is a given parameter, in our case 10^{-2}; the variable S_i indicates success on the i^{th} Monte Carlo realization. We summarize the M Monte Carlo repetitions by the total number of successes $S = \sum_i S_i$ in those M trials. S is distributed binomial $\text{Bin}(\pi, M)$ where π denotes the success probability $\pi \in [0,1]$; This probability depends on k, n, N so we write $\pi = \pi(\rho|\delta; N)$.

We *define* the location of the phase transition using logistic regression similarly to [6], [45]. The *finite-N phase transition* is the value of ρ at which success probability crosses 50%:

$$\pi(\rho|\delta; N) = \frac{1}{2} \quad \text{at} \quad \rho = \rho^*(\delta; \theta).$$

This notion is well-known in biometrics where the 50% point of the dose-response is called the LD50. (Actually there is a dependence on the tolerance tol so $\rho^*(\delta; \theta) \equiv \rho^*(\delta; \theta|N,$ tol); but this dependence is found to be weak.)

To *estimate* the phase transition from data, we collect triples $(k, M, S(k, n, N))$ all at one value of (n, N), and model $S(k, n, N) \sim \text{Bin}(\pi_k; M)$ using a generalized linear model with logistic link

$$\text{logit}(\pi) = a + b\rho,$$

where $\rho = k/n$ and $\text{logit}(\pi) = \log(\frac{\pi}{1-\pi})$; in biometric language, we assume that the dose-response probability follows a logistic curve.

The fitted parameters $\hat{a}$,$\hat{b}$, give the estimated phase transition from

$$\hat{\rho}^*(\delta;\theta) = -\hat{a}/\hat{b}.$$

We denote this estimated value by $\rho^*(\delta;\theta)$ in the rest of the paper.

V. Tuning Procedure

We conducted extensive computational experiments to evaluate the phase transitions of various algorithms. In all, we performed more than $90,000,000$ reconstructions, using 38 servers at a commercial dedicated server facility for one month. These calculations would have run more than 3 years on a single desktop computer.

For a fixed iterative scheme and a fixed tuning parameter θ, we considered in turn each of several problem suites $\mathcal{S} = (E, C)$, i.e. several combinations of random matrix ensemble E and coefficient amplitude distributions C. At each such combination, we measured the phase transitions as a function of δ.

In the tuning stage of our project we worked only with the *standard suite* $\mathcal{S}_0$; here the matrix ensemble is the Uniform Spherical Ensemble (USE) [1] matrix and the coefficient ensemble has all nonzeros randomly $\pm$ equiprobable and independent. Below we call this the *CARS* ensemble, short for Constant Amplitude, with Random Signs.

For a fixed $N = 800$ we made simulations at the standard suite, and measured the empirical phase transition $\rho^*(\delta;\theta)$ as described above. We denote the optimal parameter choice via

$$\theta^*(\delta) = \arg\max_{\theta} \rho^*(\delta;\theta). \tag{1}$$

In the later evaluation stage, other problem suites were used to test the robustness of the tuning. As it turns out, the standard suite is approximately the least favorable case and consequently our tuning typically works even better at other problem suites than it does at the standard suite. See Section VIII.

VI. Tuning Results

Figure 3 illustrates tuning results for IST on the standard suite $\mathcal{S}_0$. Here $\theta = (RelaxationParameter, FARParameter)$. Panel (a) shows the different optimized phase transitions available by tuning FAR to depend on δ while the relaxation parameter is fixed. Panel (b) shows the optimally tuned FAR parameters at each given δ and choice of relaxation parameter. Figure 4 offers the same information for IHT.

Optimum performance of IST occurs at higher values of the false alarm rate than for IHT. Decreasing the relaxation parameter beyond the range shown here does not improve the results for IST and IHT.

Figure 5 illustrates performance of TST for different values of $\theta = (\alpha, \beta)$. Panel (a) shows the different optimized phase transitions available by tuning β at fixed $\alpha = 1$ and Panel (b) shows optimal phase transitions with $\alpha = \beta$ varying. Both displays point to the conclusion that $\alpha = \beta = 1$ dominates other choices. Hence subspace pursuit $(\alpha = 1, \beta = 1)$ dominates CoSaMP $(\alpha = 1, \beta = 2)$.

VII. Recommended Choices

We provide three versions of iterative algorithms based on our optimal tuning exercise: recommended-IST, recommended-IHT and recommended-TST. They are implemented in Matlab and published at URL `sparselab.stanford.edu/OptimalTuning/main.htm`.

In our recommended versions, there are no free parameters. The user specifies only the matrix A and the left-hand side y. In particular

[1]The columns of these matrices are iid samples from the uniform distribution on the unit sphere in $\mathbb{R}^n$.

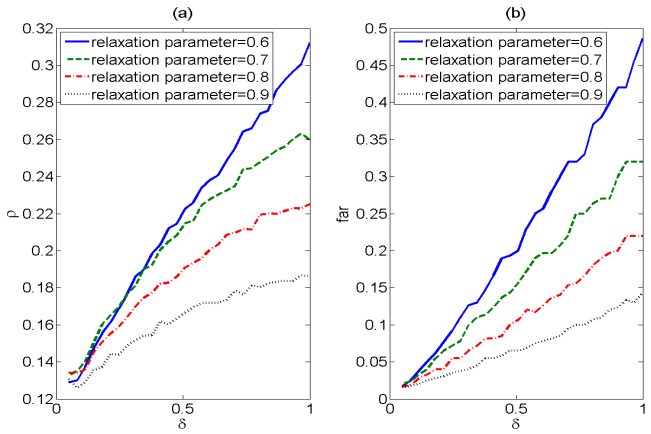

Fig. 3. (a) Optimized phase transitions for IST at various choices of relaxation parameter κ. (b) FAR parameter choices yielding those optima. The value $\kappa = 0.6$ outperforms the other choices.

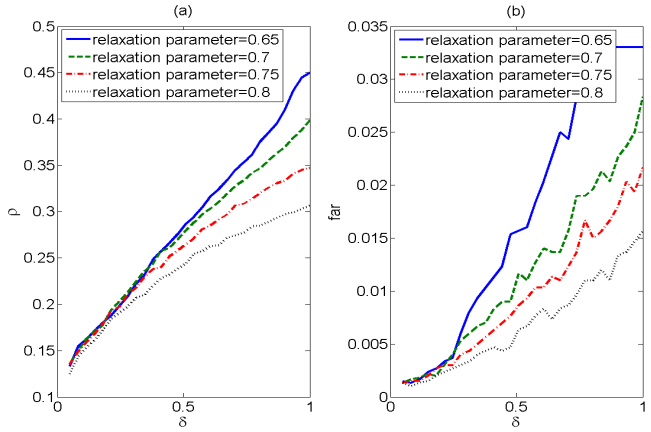

Fig. 4. (a) Optimized phase transitions for IHT at various choices of relaxation parameter κ. (b) FAR parameter choices yielding those optima. The value $\kappa = 0.6$ outperforms the other choices.

the user does not specify the expected sparsity level, which in most applications cannot be considered known.

These recommended algorithms are not the same as previously published algorithms. For example, recommended TST has parameters $\alpha = 1$ and $\beta = 1$, so it initially seems identical to subspace pursuit [24]. However, subspace pursuit demands an *oracle* to inform the user of the true underlying sparsity of the vector. Recommended-TST de facto assumes a specific value for the assumed sparsity level at each given δ (see Table III). If the actual sparsity in x_0 is better than the assumed value, the algorithm still works. If the sparsity is actually worse than what Rec-TST assumes, no other tuning of the algorithm will work either. The user does not need to know the assumed sparsity level – it is hard-coded. In effect, we have removed the oracle dependence of the subspace pursuit method.

We remind the reader that these algorithms dominate other implementations in the same class. Thus, recommended TST dominates CoSaMP; this is particularly evident for $\delta > .5$ (see Figure 5).

A companion set of algorithms – described later – is available

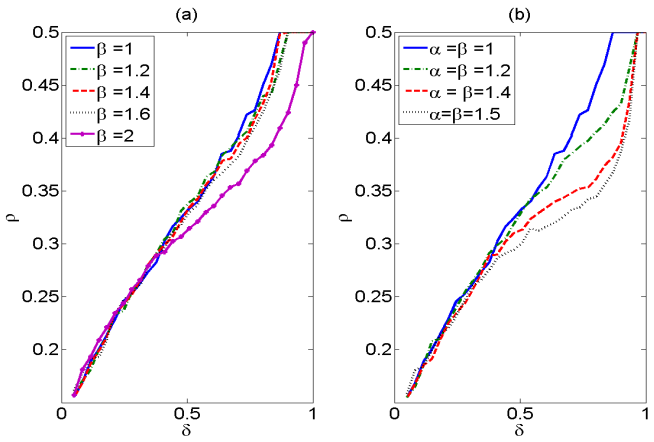

Fig. 5. (a) Empirical phase transitions of TST-(α, β) for $\alpha = 1$ and different values of β; $\beta = 1$ performs best; (b) Empirical phase transitions when $\alpha = \beta$; again $\beta = 1$ performs best. In both panels, differences are small for $\delta < .5$, i.e. undersampling factors greater than 2.

for the case where A is not an explicit matrix but instead a linear operator for which Av and $A'w$ can be computed without storing A as a matrix. Some differences in tuning for that case have been found to be valuable.

We record in the following tables a selection of the optimally tuned parameter values.

TABLE I

RECOMMENDED IST: OPTIMALLY TUNED FAR PARAMETER AND RESULTING OPTIMIZED PHASE TRANSITION, BOTH AS A FUNCTION OF δ. RECOMMENDED VALUE OF RELAXATION PARAMETER $\kappa = 0.6$.

δ	.05	.11	.21	.31	.41	.5	.6	.7	.8	.93
ρ	.124	.13	.16	.18	.2	.22	.23	.25	.27	.29
FAR	.02	.037	.07	.12	.16	.2	.25	.32	.37	.42

TABLE II

RECOMMENDED IHT: OPTIMALLY TUNED FAR PARAMETER AND RESULTING OPTIMIZED PHASE TRANSITION, BOTH AS A FUNCTION OF δ. RECOMMENDED VALUE OF RELAXATION PARAMETER $\kappa = 0.65$.

δ	.05	.11	.21	.41	.5	.6	.7	.8	.93
ρ	.12	.16	.18	.25	.28	.31	.34	.38	.41
$100 \cdot FAR$	.15	.2	.4	1.1	1.5	2	2.7	3.5	4.3

Figure 6 compares our recommended implementations with each other and with LARS [4] and OMP [8], as well as, the theoretical phase transition curve for ℓ_1. The Figure depicts empirical phase transitions at the Standard Suite. These transitions obey the following ordering:

$$\ell_1 > \text{LARS} > \text{Rec-TST} > \text{Rec-IHT} > \text{Rec-IST},$$

where ℓ_1 refers to the phase transition of the limiting probability for exact reconstruction by ℓ_1 minimization (Figure 1), and the other symbols denote empirical transitions of specific algorithms.

TABLE III

RECOMMENDED TST: OPTIMAL TUNING PARAMETERS ARE $\alpha = \beta = 1$.

δ	.05	.11	.21	.31	.41	.5	.6	.7	.8	.93
ρ	.124	.17	.22	.26	.30	.33	.368	.4	.44	.48

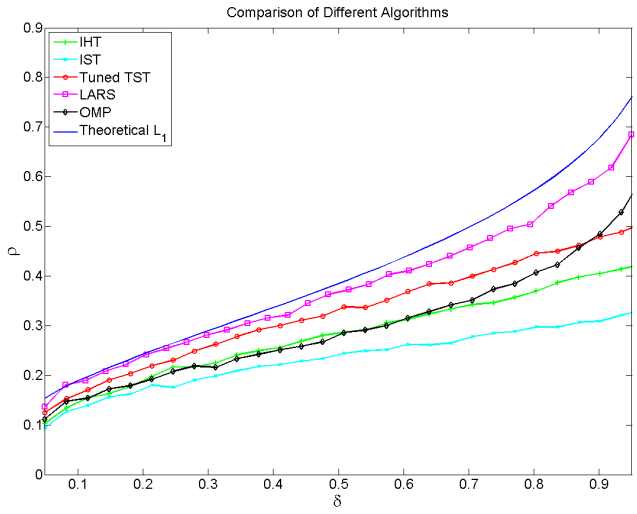

Fig. 6. Phase Transitions of several algorithms at the standard suite. Upper curve: theoretical phase transition, ℓ_1 minimization; lower curves: Observed transitions of algorithms recommended here.

One might have expected this result based on qualitative grounds; however, it is striking to see how close the curves actually are. For example, OMP performance is very similar to tuned IHT for $\delta < 0.7$. These simple iterative algorithms are dramatically easier to program and also dramatically cheaper to run on a per iteration basis, than usual optimization-based approaches. It seems that at moderate to extreme incompleteness levels one would be satisfied with IHT or IST; particularly so for very large problem sizes.

VIII. ROBUSTNESS

A *robust* choice of parameters offers a guaranteed level of performance across all situations. Such a choice can be made by solving the maximin problem

$$\theta^r(\delta) = \arg\max_{\theta} \min_{\mathcal{S} \in \mathcal{U}} \rho^*(\delta; \theta; \mathcal{S}).$$

In words, we tune θ to give the highest possible performance guarantee valid across a universe $\mathcal{U}$ of suites $\mathcal{S}$. The maximin is achieved at the worst case or *least-favorable* suite; this depends on the given algorithm, the tuning, and the universe of suites. Our universe of problem suites explored combinations of matrix ensemble E, and the coefficient ensemble C as follows. Matrix ensembles included the USE defined above, as well as matrices with random $\pm$ entries [Random Sign Ensemble (RSE)] and partial Fourier matrices (to be explained later). We also considered four coefficient ensembles C: in addition to the CARS ensemble defined above, we considered coefficients from the double exponential distribution, the Cauchy, and the uniform distribution on $[-1, 1]$.

As it turns out, our recommended tuning in effect has the maximin property. As described earlier, we tuned at the standard suite $\mathcal{S}_0$, with

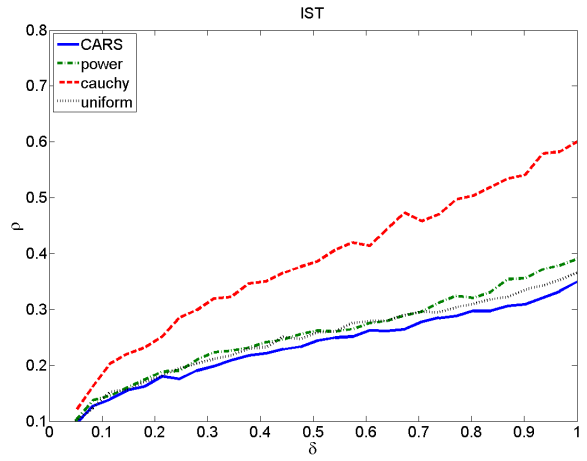

Fig. 7. Observed phase transitions of recommended IST at different coefficient ensembles.

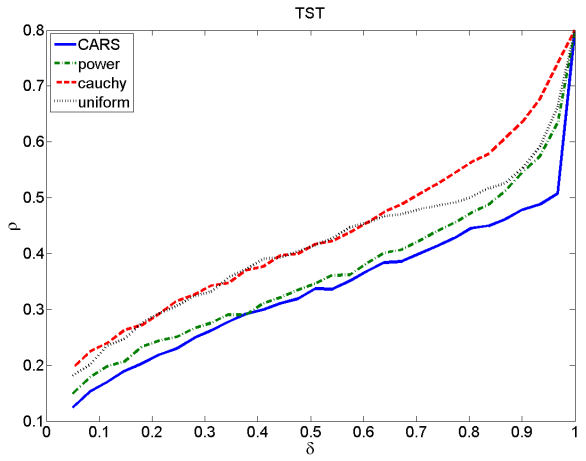

Fig. 9. Observed phase transition of recommended TST at different coefficient ensembles.

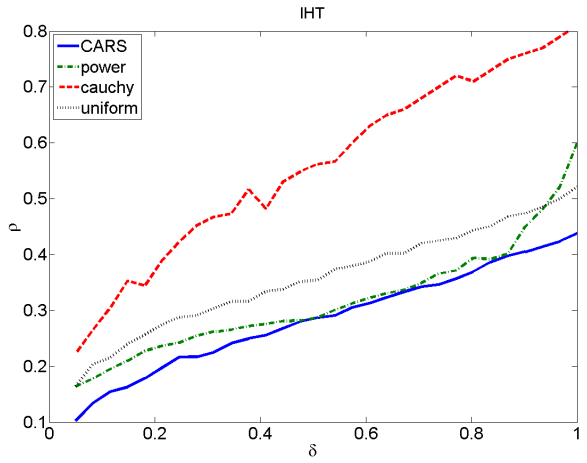

Fig. 8. Observed phase transition of recommended IHT at different coefficient ensembles.

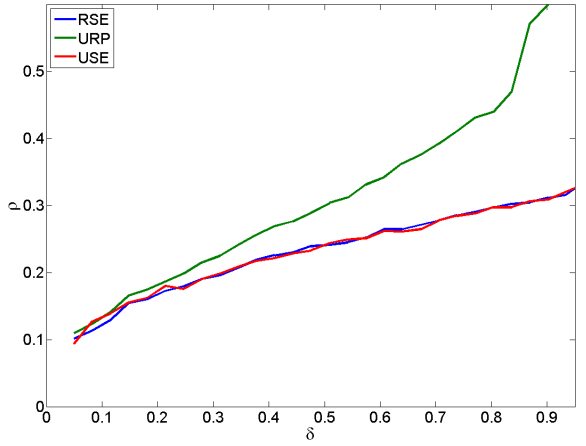

Fig. 10. Observed phase transition of recommended IST for different matrix ensembles.

constant amplitude, random-sign (CARS) coefficients and matrices from USE.

Figures 7-8-9 display results for Rec-IST, Rec-IHT, and Rec-TST at a range of problem suites. For all three algorithms, the CARS ensemble is approximately the least favorable coefficient ensemble. Since we have tuned at that ensemble, our choice of tuning parameters can be said to be robust. In other words, if the problem suite is different from the standard suite, the phase transition of the algorithm will be even better than the phase transitions tabulated in the tables above.

Figures 10-11-12 study Rec-IST, Rec-IHT, and Rec-TST at three matrix ensembles: USE, Random Sign Ensemble (RSE) where the elements of the matrix are chosen iid from ± 1, and Uniform Random Projection (URP) ensemble. Results are similar for the RSE and USE ensembles and usually better for URP. A surprising exception to the above pattern is described below in Section X.

IX. Running Times

Algorithm running times are given in Table IV. They were measured on an Intel 2 Core Processor (2.13GHz, GBytes RAM). All implementations are in Matlab. In order to give a fair comparison between algorithms with very different internal logic, we ran each iterative algorithm until a convergence criterion was met: $\frac{\|y-A\hat{x}\|_2}{\|y\|_2} \leq .001$.

X. Ensembles based on Fast Operators

The matrix ensembles discussed so far all used dense matrices with random elements. However, many applications of sparsity-seeking decompositions use linear operators which are never stored as matrices. Typically such operators can be applied rapidly so we call the resulting measurement operators *FastOps* ensembles. The partial Fourier ensemble [34] provides an example. Here the $n \times N$ matrix A has for its rows a random subset of the N rows in the standard Fourier transform matrix. Av and $A'w$ can both be computed in order $N \log(N)$ time; the comparable dense matrix vector products

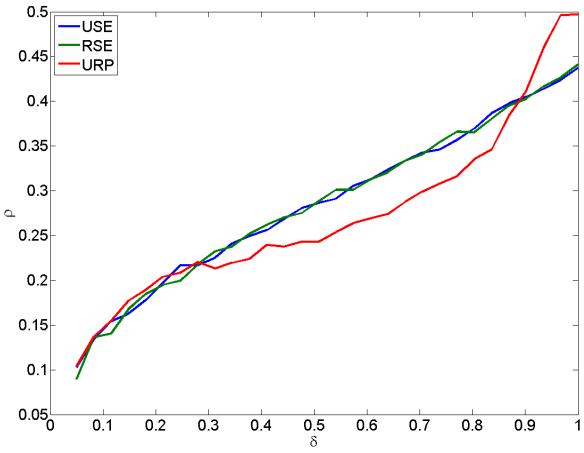

Fig. 11. Observed phase transition of recommended IHT for different matrix ensembles. Note that the red curve (URP matrix ensemble) is significantly below the blue curve (USE matrix ensemble) for $0.3 < \delta < .85$. This was the only substantial exception observed to the maximin property in our study.

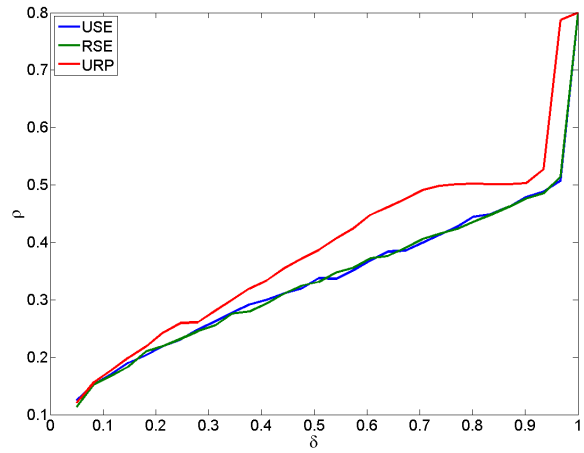

Fig. 12. Observed phase transition of recommended TST for different matrix ensembles.

would cost order N^2 flops. The partial Hadamard ensemble is defined similarly, using the Hadamard matrix of order N in place of the Fourier matrix.

The simple iterative algorithms IHT and IST are particularly suited for use with FastOps ensemble since they require only repetitive application of Av and $A'w$ interleaved with thresholding, and this is exactly how FastOps ensembles are set up to be used.

We considered two FastOps suites: the 1D partial Fourier ensemble and the 1D partial Hadamard ensembles. The *Standard* FastOps suite uses the partial Fourier matrix ensemble and CARS coefficients. Figure 13 compares the optimally-tuned performance of IHT, IST and TST at the standard FastOps suite.

We found that

- it is beneficial to tune IHT and IST specially for FastOps ensembles, because the previous tuning (aka maximin tuning) was driven by least favorable cases occurring at non-FastOps ensembles. Here such cases are ruled out, and maximin tuning only considers a narrower range of relevant cases; the achieved maximin phase transition improves.
- For TST, $\alpha = \beta = 1$ is still optimal, but for the maximin tuning restricted to FastOps, ρ^*, improves.
- the relaxation parameter in IHT/IST makes essentially no contribution to performance in this setting.
- 1D partial Hadamard and 1D partial Fourier gave very similar results.
- the performance of IHT is very much in line with earlier results for the random matrix ensembles.
- IST behaves dramatically better at partial Fourier ensembles than for the random matrix ensembles (Figure 14) and even

TABLE IV

ALGORITHM TIMINGS AT THE STANDARD SUITE. AVERAGE RUNNING TIME (SEC) UNTIL $\frac{\|y-A\hat{x}\|_2}{\|y\|_2} \leq .001$. AVERAGES COVER 10 INDEPENDENT RUNS. PROBLEM SIZES AS INDICATED.

N	δ	ρ	IHT	TST	OMP	LARS
2000	0.9	0.17	10.6	12	20	28
4000	0.9	0.17	44.8	91.2	157	216
6000	0.9	0.17	90	286	537	798
2000	0.7	0.28	7.2	3.3	7.6	11.5
4000	0.7	0.28	28.4	24.5	57.8	98.4
6000	0.7	0.28	64.5	118	188	987
2000	0.5	0.2	5.8	0.91	1.5	2.7
4000	0.5	0.2	23	7	12	20
6000	0.5	0.2	52	23	38	65
8000	0.5	0.2	91	52	97	164
10000	0.5	0.2	130	100	168	270
2000	0.3	0.12	2	0.08	0.25	0.4
4000	0.3	0.12	9	0.65	1.8	2.6
8000	0.3	0.12	34	5	15	22
10000	0.3	0.12	54	13	28.5	38.5

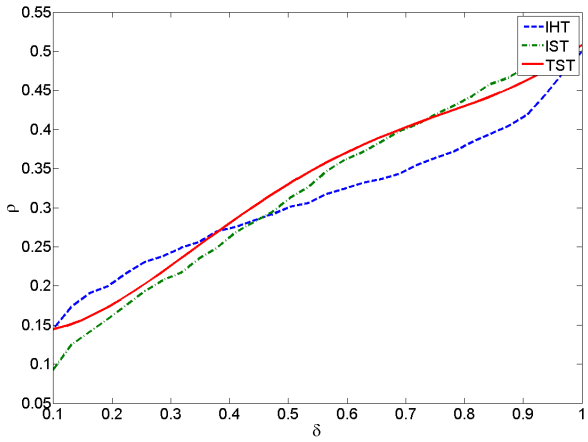

Fig. 13. Comparison of the performance of recommended IHT, IST and TST for partial fourier ensemble.

outperforms IHT for $\delta > .5$ (Figure 13).

Recommended parameters are shown in Tables V-VI. Running times are studied in Table VII. The execution times of both the fast IHT and fast TST scale favorably with the problem size N. In most of our studies TST is faster than IHT and they are both much faster than LARS. The favorable timing results of TST on large problem sizes surprised us.

TABLE V
RECOMMENDED IST, STANDARD FASTOPS SUITE. OPTIMALLY TUNED FAR AND OPTIMIZED PHASE TRANSITION ρ^* OF THE RECOMMENDED IST ALGORITHM. RECOMMENDED RELAXATION PARAMETER $\kappa = 1$.

δ	.11	.21	.31	.41	.5	.6	.7	.8	.9
ρ	.092	.16	.21	.26	.31	.37	.41	.44	.48
FAR	.0209	.0736	.13	.19	.26	.32	.32	.32	.32

TABLE VI
RECOMMENDED IHT, STANDARD FASTOPS SUITE. OPTIMALLY TUNED FAR AND OPTIMIZED PHASE TRANSITION ρ^* OF THE RECOMMENDED IHT ALGORITHM. RECOMMENDED RELAXATION PARAMETER $\kappa = 1$.

δ	.05	.11	.21	.31	.41	.5	.6	.7	.8
ρ	.056	.14	.2	.24	.27	.3	.32	.34	.38
1000FAR	.3	.4	1.8	2.9	3.8	5	5	5	5

TABLE VII
ALGORITHM TIMINGS AT THE STANDARD FASTOPS SUITE. AVERAGE RUNNING TIME (SEC) UNTIL $\frac{\|y-A\hat{x}\|_2}{\|y\|_2} \leq .001$. AVERAGES COVER 10 INDEPENDENT RUNS. PROBLEM SIZES AS INDICATED.

N	δ	ρ	IHT	TST	LARS
8192	.1	.1	.5	.1	.6
16384	.1	.1	1.2	.25	2.5
32768	.1	.1	2.56	.48	10.8
65536	.1	.1	8	2.3	65
131072	.1	.1	18	5.6	> 900
262144	.1	.1	39	13	> 900
524288	.1	.1	85	27	> 900
16384	.3	.18	.5	.4	25
8192	.3	.18	.25	.21	5.2
8192	.5	.21	.18	.19	13.5
16384	.5	.21	.38	.4	81

XI. DISCUSSION

A. Before Using These Results

Readers may find the following reminders helpful:

- Our software already has embedded within it the appropriate values from the tables presented here, so you may not need to copy information from the tables and apply it. However, if you need to code your own implementation, remember that the parameter $\rho = \rho^*$ in our tables specifies the largest workable k^* via $k^* = \lfloor \rho^* \cdot n \rfloor = \lfloor \rho^* \cdot \delta \cdot N \rfloor$.
- Your A matrix must be normalized so that all columns have unit Euclidean norm; for a badly-scaled matrix the algorithms may diverge rapidly.
- The software assumes the sparsity level k is unknown *a priori*, and uses, for each level of the indeterminacy ratio $\delta = n/N$, the largest workable sparsity level k^*. If your application provides an oracle that makes k known in advance. you may wish to customize the code to use this information – but this is not necessary.

B. The Computational Effort-Phase Transition Tradeoff

This project adopted the goal of squeezing the best phase transition performance out of some simple, computationally feasible algorithms. Staring us in the face is the fact that ℓ_1 minimization generally offers better performance (higher phase transitions) than any of our tuned algorithms.

The referees observed, and we agree, that one should keep in mind the tradeoff between computational effort and phase transition. At one extreme, explicit combinatorial enumeration, though extravagantly expensive for large problems, will obtain optimal phase transition performance. At the other extreme fall the iterative heuristic algorithms studied here, which can scale to very large problem sizes, but offer more modest phase transition performance. In between, we have convex optimization solvers, which truly solve ℓ_1 minimization and therefore achieve the phase transition curve ρ_{ℓ_1} which, it turns out, dominates the maximin phase transitions of all the iterative methods discussed here. Such solvers run in (pseudo-)polynomial time, but demand considerably more computational effort than the iterative algorithms considered here.

The tradeoff of computation for phase transition performance is a developing frontier; in particular, there is rapid progress in improving ℓ_1 solvers. Seven years ago, when the work reported in [13], [14], [15] was conducted, heuristic iterative methods were really the only way to get a start on realistic sized problems. Apparently the popularity of research in Compressed Sensing has led to vast improvements in the performance of available ℓ_1 solvers. In this project we did not attempt to document what is possible today using such solvers; an interesting follow-up project might involve the tuning of purported ℓ_1 solvers like Bregman iteration [26] to see if they really can achieve the ℓ_1 transition; another interesting follow-up project would be to carefully document CPU times of available ℓ_1 solvers such as L1LS [5], GPSR [46], SPGL1 [47], and SpaRSA [48]. Because we follow the paradigm of reproducible computational research it will be possible for others to easily compare such methods with our recommended methods on the exact same range of problem sizes studied in this project.

C. Contributions

We mention two larger contributions from our work:

1) The State of the Art: Our work may be useful to active researchers in the field of sparsity-promoting methods. Our study effectively defines the *current state of the art* (CSA), a precisely-specified set of quantitative performance standards which are the best we currently know how to do (within a certain class of algorithms). Once this is defined, any newly proposed algorithm can be evaluated with reference to the CSA, and other researchers can use this comparison to understand the relative improvement, if any, offered by the proposal. Over time, as genuine improvements emerge, the CSA will evolve, by definition always offering the best known current performance.

More broadly, a researcher with a new method *not* intended for comparison with the CSA on a standard suite may expand the set of suites beyond those studied here, adding to such a suite a new matrix or coefficient ensemble, or may provide a new measure of success. This allows the researcher to clearly demonstrate for colleagues the arena where the method is intended to contribute.

The effort to create standard performance metrics and define CSA performance has been valuable in many fields of image and signal processing. Indeed, one can argue that in fingerprint recognition and face recognition, the moment when those fields really started to make progress is precisely the moment when defined databases and success metrics were made available for community use, allowing systematic comparison of algorithms [49], [50]. Similarly, the regular publication of standardized challenge problems in arenas like protein structure prediction is said to have utterly transformed the field. Defining a CSA for sparsity-promoting methods can likewise be expected to lead to much faster and more reliable progress.

Standardization has another predictable effect: it may *improve communication among researchers in the field* reducing the impact of marketing, branding and prestige and increasing the focus on objective measures performance. The example of protein structure prediction (CASP) bears this out [51]; before CASP, certain approaches to protein structure prediction were considered more likely to work than others, but it has been reported that CASP upset expectations, reversing the order of preference for certain algorithms [52]. In fact our work may already be showing this effect: one of our findings is that the more prominent algorithm CoSaMP is dominated by the less well-known algorithm Subspace Pursuit at the suites we have studied and for the success measure we have used.

2) Promoting Reproducible Computational Research: An implicit but still important contribution of our work is adherence to the paradigm of Reproducible Computational Research [53]. The data and code required to reproduce our results are freely available – and not just the conclusions. A researcher or potential user can study our implementation and tuning of an algorithm or our definitions of performance metric, or our collection of problem suites. This speeds up progress in developing and validating new sparsity-seeking algorithms.

- Researchers who feel our study ignores important aspects of performance can define new metrics of success which better reflect their views of what is important, and then conduct parallel studies with that new metric.
- Potential users interested in a specific problem suite we haven't studied, but which is of direct interest in their application, may easily extend our software to add a new suite to the available collection and then run a robustness study or even a tuning study focused on that suite.

By sharing the code underlying our study, we promote further developments of new applications and new methods which outperform current ones.

XII. Conclusions

We defined a set of problem suites and two algorithmic schemes that cover several distinctly branded methods in the literature. We defined algorithm performance using the notion of empirical phase transition and made millions of reconstruction attempts, while systematically varying problem specifications. We identified specific parameter choices which are optimal at so-called standard suites, for specific sparsity-indeterminacy combinations. This produced recommended- IST, IHT and TST algorithms, coded in Matlab and are freely available at URL `sparselab.stanford.edu/OptimalTuning/main.htm`. They can be used 'out of the box' on problem instances of the type we have studied; the user need not specify any parameters whatever in order to run them; simply providing the matrix A and the left-hand side y of the system $y = Ax$ (but keep in mind section XI-A).

Our studies included extensive computations at other suites besides the standard one, verifying the robustness of our parameter choices and checking that at those other suites the recommended algorithms generally behave better than they do at the standard suite.

The standard suite involves random matrices, but many applications of sparsity-seeking algorithms, particularly in compressed sensing, use structured matrices. We did consider an important class of structured matrices based on fast transforms such as 1D and 2D Fourier transforms and fast Hadamard transforms. Such matrices admit of rapid computations with very large problem sizes and in some cases are actually demanded by the application, for example in MR imaging and NMR spectroscopy. We studied some very large problem sizes with problem suites based on these fast transforms and found results largely matching those we found on the random ensembles. In one case – IST with 2D partial fourier – we found tunings which generate unexpectedly high phase transitions markedly better than what we saw for the standard suite. We published recommended choices of IHT, IST for use with such ensembles defined by those fast operators.

We reached the following empirical findings at the 'random' matrix ensembles:

- Phase transitions for optimally-tuned algorithms obey the ordering Rec-TST $>$ Rec-IHT $>$ Rec-IST.
- Setting the relaxation parameter to 0.6 for IST and 0.65 for IHT improves performance of those algorithms significantly. Relaxation has no noticeable effect on performance of TST.
- Performance of the matrix ensemble USE (start with iid Gaussian entries then normalize column lengths) is very similar to RSE (random ± 1 entries).
- The distribution of coefficient amplitudes in the solution x_0 matters very much to these algorithms. The worst case is when all nonzeros have the same amplitude.
- For a given problem suite, the number of iterations to reach a given accuracy of recovery does not seem to depend on problem size, except perhaps near to phase transition.
- Subspace pursuit works better than CoSaMP on the standard suite. Our recommended TST algorithm is essentially subspace pursuit, without need of an oracle.

Conclusions for the special case where the sensing matrix is implicitly defined using a fast linear operator were listed in Section X.

Acknowledgements

Thanks to Jared Tanner for valuable suggestions on an early draft, and the Special Issue Editors, especially Rick Chartrand and Mario Figuereido, for their service to the IEEE. This work was partially supported by NSF DMS 05-05303.

References

[1] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[2] D. L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions on Information Theory*, vol. 52, pp. 6–18, January 2006.

[3] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 1030–1051, 2006.

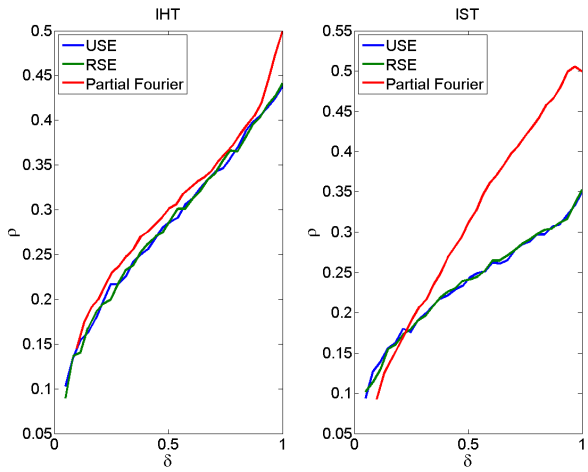

Fig. 14. (a) Phase transitions of recommended IHT for different matrix ensembles (b) Phase transitions of recommended IST for different matrix ensembles. Generally speaking, the partial Fourier ensemble is not the worst case ensemble; note especially the case IST with $\delta > 0.4$.

[4] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, pp. 407–499, 2004.

[5] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "A method for large-scale l1-regularized least squares," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, pp. 606–617, December 2007.

[6] D. L. Donoho and Y. Tsaig, "Fast solution of ℓ_1-norm minimization problems when the solution may be sparse," *IEEE Transactions on Information Theory*, vol. 54, pp. 4789–4812, November 2008.

[7] S. Mallat and S. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, December 1993.

[8] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, 1993.

[9] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[10] D. L. Donoho, I. Drori, Y. Tsaig, and J. L. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," *Stanford Statistics Department Technical Report*, 2006.

[11] S. Sardy, A. G. Bruce, and P. Tseng, "Block coordinate relaxation methods for nonparametric wavelet denoising," *Journal of Computational and Graphical Statistics*, vol. 9, pp. 361–379, 2000.

[12] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 75, pp. 1412–1457, 2004.

[13] J. L. Starck, M. Elad, and D. L. Donoho, "Redundant multiscale transforms and their application for morphological component analysis," *Journal of Advances in Imaging and Electron Physics*, vol. 132, pp. 287–348, 2004.

[14] J. L. Starck, M. Elad, and D. L. Donoho, "Image decomposition via the combination of sparse representations and a variational approach," *IEEE Transactions on Image Processing*, vol. 14, pp. 1570–1582, October 2005.

[15] M. Elad, J. L. Starck, P. Querre, and D. L. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *Applied and Computational Harmonic Analysis*, vol. 19, pp. 340–358, November 2005.

[16] K. K. Herrity, A. C. Gilbert, and J. A. Tropp, "Sparse approximation via iterative thresholding," *Proc. ICASSP*, vol. 3, pp. 624–627, May 2006.

[17] M. Elad, B. Matalon, J. Shtok, and M. Zibulevsky, "A wide-angle view at iterated shrinkage algorithms," *Proc. SPIE (Wavelet XII)*, August 2007.

[18] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *Journal of Fourier Analysis and Applications, special issue on sparsity*, vol. 14, no. 5, pp. 629–654, 2008.

[19] M. Fornasier and H. Rauhut, "Iterative thresholding algorithms," *Applied and Computational Harmonic Analysis*, vol. 25, no. 2, pp. 187–208, 2008.

[20] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2008.

[21] T. Blumensath and M. E. Davies, "How to use the iterative hard thresholding algorithm," *Proc. SPARS*, 2009.

[22] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[23] D. Needell and R. Vershynin, "Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit," *IEEE Journal of Selected Topics in Signal Processing*, 2009.

[24] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[25] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling and Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[26] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing," *SIAM Journal on Imaging Sciences*, vol. 1, no. 1, pp. 143–168, 2008.

[27] I. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm," *IEEE Transactions on Signal Processing*, vol. 45, pp. 600–616, March 1997.

[28] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of American Statistical Association.*, vol. 96, no. 456, pp. 1348–1360, 2001.

[29] R. Chartrand, "Exact reconstructions of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, pp. 707–710, 2007.

[30] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, "One sketch for all: fast algorithms for compressed sensing," *Proc. STOC*, pp. 237–246, 2007.

[31] M. Lustig, D. L. Donoho, and J. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, pp. 1182–1195, December 2007.

[32] J. Bobin, J. L. Starck, and R. Ottensamer, "Compressed sensing in astronomy," *IEEE Journal of Selected Topics in Signal Processing*, vol. 2, no. 5, pp. 718–726, 2008.

[33] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, pp. 489–509, April 2006.

[34] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, pp. 489–509, February 2006.

[35] J. Bobin, J. L. Starck, J. Fadili, Y. Moudden, and D. L. Donoho, "Morphological component analysis: An adaptive thresholding strategy," *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2675–2681, 2007.

[36] K. Schnass and P. Vandergheynst, "Average performance analysis for thresholding," *IEEE Signal Processing Letters*, vol. 14, no. 11, pp. 828–831, 2007.

[37] D. L. Donoho, "High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension," *Discrete and Computational Geometry*, vol. 35, no. 4, pp. 617–652, 2006.

[38] D. L. Donoho and J. Tanner, "Counting faces of randomly projected polytopes when the projection radically lowers dimension," *Journal of American Mathematical Society*, vol. 22, pp. 1–53, 2009.

[39] D. L. Donoho and J. Tanner, "Precise undersampling theorems," *IEEE Transactions on Information Theory*. submitted for publication.

[40] D. L. Donoho and J. Tanner, "Neighborliness of randomly-projected simplices in high dimensions," *Proceedings of the National Academy of Sciences*, vol. 102, no. 27, pp. 9452–9457, 2005.

[41] D. L. Donoho and J. Tanner, "Counting the faces of randomly projected hypercubes and orthants with applications," *Discrete and Computational Geometry*. to appear.

[42] R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss, "Combining geometry with combinatorics: a unified approach to sparse signal recovery," 2008. submitted.

[43] E. Candès, "The restricted isometry property and its implications for compressed sensing," *Compte Rendus de l'Academie des Sciences*, vol. 346, pp. 589–592, 2008.
[44] J. D. Blanchard, C. Cartis, and J. Tanner, "The restricted isometry property and l_q-regularization: Phase transitions for sparse approximation," submitted.
[45] D. L. Donoho and J. Tanner, "Observed universality of phase transitions in high-dimensional geometry, with applications in modern signal processing and data analysis," *Philosophical Transactions of the Royal Society A*, vol. 367, no. 1906, pp. 4273–4293, 2009.
[46] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics of Signal Processing*, vol. 1, no. 4, pp. 586–598, 2007.
[47] E. van den Berg and M. P. Friedlander, "Probing the pareto frontier for basis pursuit solutions," *SIAM Journal on Scientific Computing*, vol. 31, no. 2, pp. 890–912, 2008.
[48] S. Wright, R. Nowak, and M. Figueiredo, "Sparse reconstruction by separable approximation," *Proc. of ICASSP*, 2009.
[49] *http://www.frvt.org/FRGC/*.
[50] *http://www.itl.nist.gov/iad/894.03/fing/fing.html/*.
[51] *http://predictioncenter.org/*.
[52] *Personal Communication, David Haussler*.
[53] D. L. Donoho, A. Maleki, I. U. Rahman, M. Shahram, and V. Stodden, "Reproducible research in computational harmonic analysis," *Computing in Science and Engineering*, vol. 11, pp. 8–18, January/February 2009.