A robust parallel algorithm for combinatorial compressed sensing

Rodrigo Mendoza-Smith*†, Jared Tanner*†, and Florian Wechsung*

* Mathematical Institute, University of Oxford, Oxford, OX2 6GG.
† Alan Turing Institute, British Library, London, NW1 2DB.

***Abstract*—It was shown in [1] that a vector $\mathbf{x} \in \mathbb{R}^n$ with at most $k < n$ nonzeros can be recovered from an expander sketch $\mathbf{Ax}$ in $\mathcal{O}(\mathrm{nnz}(\mathbf{A}) \log k)$ operations via the Parallel-ℓ_0 decoding algorithm, where $\mathrm{nnz}(\mathbf{A})$ denotes the number of nonzero entries in $\mathbf{A} \in \mathbb{R}^{m\times n}$. In this paper we present the Robust-ℓ_0 decoding algorithm, which robustifies Parallel-ℓ_0 when the sketch $\mathbf{Ax}$ is corrupted by additive noise. This robustness is achieved by approximating the asymptotic posterior distribution of values in the sketch given its corrupted measurements. We provide analytic expressions that approximate these posteriors under the assumptions that the nonzero entries in the signal and the noise are drawn from continuous distributions. Numerical experiments presented show that Robust-ℓ_0 is superior to existing greedy and combinatorial compressed sensing algorithms in the presence of small to moderate signal-to-noise ratios in the setting of Gaussian signals and Gaussian additive noise.**

***Index Terms*—compressed sensing, expander graphs, dissociated signals, robust algorithms.**

I. Introduction

COMPRESSED sensing is a well studied method by which a sparse or compressible vector can be acquired by a number of measurements proportional to the number of its dominant entries [2], [3]. To fix notation, let χ_k^n be the set of vectors in $\mathbb{R}^n$ that have at most k non-zero entries, let $\mathbf{x} \in \chi_k^n$ and let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be a matrix with $m < n$. We will refer to $\mathbf{A}$ as the measurement matrix, $\mathbf{x}$ as the signal and $\mathbf{y} = \mathbf{Ax}$ as the measurements. The goal of compressed sensing is to recover the sparsest, most parsimonious, $\mathbf{x} \in \mathbb{R}^n$ from the measurements $\mathbf{y}$ and the matrix $\mathbf{A}$. Letting $\|\cdot\|_0$ denote the number of non-zeros in $\mathbf{x}$, the problem of finding $\mathbf{x}$ can be written as

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{x}\|_0 \text{ s.t. } \mathbf{Ax} = \mathbf{y}.$$

Many algorithms have been developed to solve this problem or equivalent formulations and there are good theoretical results on when and how fast recovery of a signal is possible given certain types of measurement matrix $\mathbf{A}$ and signal $\mathbf{x}$. These algorithms can be broadly categorised into convex optimization based algorithms like those implemented in [4]–[7] and greedy algorithms [8]–[14], and were designed and analysed for the setting of dense sensing matrices; e.g. independent (sub-)Gaussian entries or randomly subsampled Fourier matrices. Alternatively, algorithms can be categorised in terms of the ensemble where the measurement matrix is sampled from. In [15] two main approaches were identified: the *geometric* approach in which the measurement matrices satisfy $\|\mathbf{Ax}\|_2 \approx \|\mathbf{x}\|_2$ for all $\mathbf{x} \in \chi_k^n$; and the *combinatorial* approach in which the measurement matrices satisfy $\|\mathbf{Ax}\|_1 \approx \|\mathbf{x}\|_1$ for all $\mathbf{x} \in \chi_k^n$. The geometric approach is mainly represented by (sub-)Gaussian and partial Fourier ensembles, while the combinatorial approach is represented by *expander matrices* or adjacency matrices of so-called unbalanced expander graphs. To introduce a precise definition, we borrow notation from combinatorics and use the shorthands $[n] := \{1, \dots, n\}$, $[n]^{(k)} := \{S \subset [n] : |S| = k\}$ where $|S|$ denotes the cardinality of the set S, and $[n]^{(\leq k)} := \cup_{\ell\leq k}[n]^{(\ell)}$ for $n, k \in \mathbb{N}$ and $k < n$. Additionally, we let $\mathbb{1}_E$ be an indicator function that evaluates to 1 when E is true and to 0 otherwise.

Definition I.1 (Expander matrices [1])**.** The matrix $\mathbf{A} \in \{0,1\}^{m\times n}$ is a (k, ε, d)-expander matrix if $\sum_{i=1}^m \mathbb{1}_{|\mathbf{A}_{i,j}|>0} = d$ for all $j \in [n]$ and

$$\left|\left\{i \in [m] : \sum_{j\in S} \mathbb{1}_{|\mathbf{A}_{i,j}|>0}\right\}\right| > (1-\varepsilon)d|S|$$

for all $S \in [n]^{(\leq k)}$. We denote by $\mathbb{E}_{k,\varepsilon,d}^{m\times n}$ the set of (k, ε, d)-expander matrices of dimension $m \times n$.

Intuitively, $\mathbf{A} \in \{0,1\}^{m\times n}$ is an expander matrix if each column has $d \ll m$ non-zeros and every subset of at most k columns has at least a fixed fraction of rows with exactly one non-zero. Expander matrices are important in compressed-sensing applications because they are highly-structured, which makes them highly efficient for storage, generation and computation of matrix-vector products. Though Fourier ensembles are also structured enough to yield competitive efficiency guarantees, these are not known to achieve the optimal sampling rate of $\mathcal{O}(k\log(n/k))$. Instead, they are only known to achieve a sampling rate of $m = \mathcal{O}\left(k\log^2 k \log n\right)$ [16]. This sub-sampling guarantee is of special relevance in the context of *linear sketching* applications in which high-dimensional sparse datasets need to be reduced to subspaces of lower dimensionality without distorting the relative geometry of the dataset [17]. Additionally, this is of crucial importance for *data-stream computing* which studies algorithms that compute on data-streams or sequences of data that can only be read

RMS acknowledges the support of CONACyT

This work was supported by The Alan Turing Institute under the EPSRC grant EP/N510129/1

Furthermore, it is based on work partially supported by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with PA Consulting Group

once by the algorithm [18]. When the dimensionality of the data $\mathbf{x} \in \chi_k^n$ is very large, it is sought to sketch or summarise $\mathbf{x}$ via $\mathbf{Ax} \in \mathbb{R}^m$ via a low-complexity mapping $\mathbf{A}$ that allows for efficient updating of the sketch whenever an entry of $\mathbf{x}$ is modified [17]. Expander matrices are also fit for traditional compressed-sensing imaging applications when the image is naturally sparse.

Ever since the seminal work of [15] a number of combinatorial compressed-sensing algorithms have been proposed [19]–[24], see [1] for a review. Of particular interest is [23] which shows that if $\mathbf{y} = \mathbf{Ax}$ is an expander sketch and $\mathbf{x} \in \chi_k^n$, then there always exists an index $\{j\} \subset [n]$ such that, $|\{i \in [m] : \mathbf{y}_i = \mathbf{x}_j\}|$ is bounded below by a positive constant depending on d and ε; that is, if $\varepsilon < 1/4$ there at least one entry in $\mathbf{x}$ whose value will be repeated at last $d/2$ times in $\mathbf{y}$. This is further shown to imply that $j \in \mathrm{supp}(\mathbf{x})$ by invoking a property of $\mathbf{A}$ that can be derived from Definition I.1. In [1] this algorithm was extended assuming a *dissociated* model on the measurements, i.e. measurements that satisfy that disjoint sums of signal entries are never equal, see Definition A.1 in the appendix.

Note that any signal with non-zero entries drawn independently from a continuous distribution is dissociated. The use of dissociated signals for noiselss decoding was first proposed by [19] in their *Sudocodes* algorithm, while an extension for noisy measurements was given in [24]. A consequence of the dissociated model for nonzeros in $\mathbf{x}$ is that for $\mathbf{A}$ having binary entries, the sum $\sum_j \mathbf{A}_{i,j}\mathbf{x}_j \neq 0$ for every $i \in [m]$. This observation is used in distinct and complementary ways to motivate Sudocodes and peeling algorithms. In particular Sudocodes uses that if $\mathbf{y}_j$ is believed to be non-zero only due to the additive noise, then any column of $\mathbf{A}$ containing a nonzero in its j^{th} entry is excluded and this reduced problem is then passed to any standard algorithm for compressed sensing (or least squares). In contrast other peeling algorithms, such as [1], [20], [23], primarily exploit that if $\mathbf{A}$ is also an expander graph then one will often observe many entires in $\mathbf{y}$ which, up to the additive noise, are equal. In particular, [1] showed that if $\mathbf{y} = \mathbf{Ax}$ is an expander sketch and either $\mathbf{x} \in \chi_k^n$ is dissociated or each column of $\mathbf{A}$ is scaled by an entry from a dissociated vector $\mathbf{s} \in \mathbb{R}^n$ with $\|\mathbf{s}\|_0 = n$, then there exists *a set* $T \subset [n]$ of cardinality strictly greater than one such that $|\{i \in [m] : \mathbf{y}_i = \mathbf{x}_j\}|$ is bounded below by a positive constant depending on d and ε. This guarantees that if $|\{i \in \mathrm{supp}(a_j) : y_i = y_\ell\}| > d/2$ then $\mathbf{x}_j = \mathbf{y}_\ell$ and provides a framework for a parallel updating mechanism, which is implemented as the Parallel-ℓ_0 algorithm (Algorithm 1) [1].

Lemma I.2. Let $\mathbf{y} = \mathbf{Ax}$, $\mathbf{x}$ dissociated, $\mathbf{A} \in \mathbb{E}_{k,\varepsilon,d}^{m\times n}$ with $\varepsilon < \frac{1}{4}$. Then there exists a nonempty set $T \subset [n] \times \mathbb{R}$ such that

$$|\{i \in \mathcal{N}(j) : \mathbf{y}_i = \omega\}| \geq (1-2\varepsilon)d \;\; \forall (j,\omega) \in T$$

and for every tuple in T that satisfies this property, we have $w = \mathbf{x}_j$.

This means that at each iteration, if the residual $\mathbf{r} \in \mathbb{R}^m$ is non-zero, i.e. if we have not yet found the correct $\mathbf{x}$, then there is a set of entries in $\mathbf{x}$ that we can change so that we reduce the number of non-zeros in $\mathbf{r}$ by at least $|T|(1-2\varepsilon)d$. More precisely, Parallel-ℓ_0 implements these observations by letting $\hat{\mathbf{x}} = 0$ be an initial estimation and by iteratively estimating the decrease in $\|\mathbf{y}\|_0$ when performing the update $\hat{\mathbf{x}}_j \leftarrow \hat{\mathbf{x}}_j + \mathbf{y}_\ell$. For a more complete description we let $\mathcal{N}(j) := \{i \in [m] : |\mathbf{A}_{ij}| > 0\}$ be the *neighbours* of an entry $j \in [n]$. To estimate the decrease in $\|\mathbf{y}\|_0$, Parallel-ℓ_0 computes

$$n_e \leftarrow |\{\ell \in \mathcal{N}(j) : \mathbf{y}_i = \mathbf{y}_\ell\}|, \tag{1}$$

$$n_z \leftarrow |\{\ell \in \mathcal{N}(j) : \mathbf{y}_\ell = 0\}|. \tag{2}$$

If $n_e > n_z$ then updating entry j in $\mathbf{x}$ with $\mathbf{r}_i$ will lead to a decrease of the residual in the $\|\cdot\|_0$ norm.

Algorithm 1: Parallel-ℓ_0 [1]

Data: $\mathbf{A} \in \mathbb{E}_{k,\varepsilon,d}^{m\times n}$; $\mathbf{y} = \mathbf{Ax} \in \mathbb{R}^m$ for $\mathbf{x} \in \chi_k^n$; $\alpha \in (1, d]$
Result: $\hat{\mathbf{x}} \in \mathbb{R}^n$ s.t. $\hat{\mathbf{x}} = \mathbf{x}$
$\hat{\mathbf{x}} \leftarrow \mathbf{0}$, $\mathbf{r} \leftarrow \mathbf{y}$;
while *not converged* **do**
  for $j \in [n]$ **do**
    $\mathbf{u} \leftarrow \mathbf{0}$;
    for $i \in \mathcal{N}(j)$ **do**
      if $\mathbf{r}_i \neq 0$ **then**
        $n_e \leftarrow |\{\ell \in \mathcal{N}(j) : \mathbf{r}_i = \mathbf{r}_\ell\}|$;
        $n_z \leftarrow |\{\ell \in \mathcal{N}(j) : \mathbf{r}_\ell = 0\}|$;
        if $n_e - n_z \geq \alpha$ **then**
          $\mathbf{u}_j \leftarrow \mathbf{r}_i$;
        end
      end
    end
  end
  $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \mathbf{u}$;
  $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{A}\hat{\mathbf{x}}$;
end

It is shown in [1] that Parallel-ℓ_0 provably converges to the true signal in $\mathcal{O}(\mathrm{nnz}(\mathbf{A})\log(k))$ operations.

Theorem I.3 (Convergence of Algorithm 1 [1])**.** Let $\mathbf{A} \in \mathbb{E}_{k,\varepsilon,d}^{m\times n}$ and let $\varepsilon \leq \frac{1}{4}$, and $\mathbf{x} \in \chi_k^n$ be dissociated. Then, Parallel-ℓ_0 with $\alpha = (1-2\varepsilon)d$ can recover $\mathbf{x}$ from $\mathbf{y} = \mathbf{Ax}$ in $\mathcal{O}(\log k)$ iterations of complexity $\mathcal{O}(dn)$.

To put this result into context and show its applicability, we refer the reader to the remark after Definition II.1, stating that random matrices as considered in this work are indeed expander matrices with high probability [25], [26, Theorem 13.6]. We furthermore emphasize the fact that the algorithm is designed in a way that allows for massively parallel implementations.

A. Main contributions and outline of the paper

The main contribution of this work is the extension of the Parallel-ℓ_0 algorithm to a case in which the measurement vector is corrupted by noise. We do this by replacing n_e and n_z in (14)-(15) with probability functions that estimate the likelihood of entries in the residual being equal to zero

or equal to another entry in the residual respectively. We state the main result necessary for this as well as the resulting algorithm in Section II. In Section III we derive the probability functions and remark upon some practical considerations like their computationally efficient calculation in the Gaussian case. In Section IV we present numerical experiments which demonstrate Robust-ℓ_0 to perform superior to a number of leading greedy and combinatorial compressed sensing algorithms. Further technical details are reserved for the Appendix.

II. Robust-ℓ_0 decoding

We now consider the case where the measurements $\mathbf{y}$ are subject to additive noise, i.e. instead of $\mathbf{y} = \mathbf{A}\mathbf{x}$, we measure $\hat{\mathbf{y}} = \mathbf{y} + \boldsymbol{\eta}$ where $\boldsymbol{\eta}$ is a realization of a random vector with $\boldsymbol{\eta}_i \sim \nu$.

Parallel-ℓ_0 is not able to cope with additive noise since, when computing n_e and n_z, it needs to determine whether a value in $\hat{\mathbf{y}}$ is zero and whether two values in $\hat{\mathbf{y}}$ are equal to each other. While for very small noise levels we could consider two values as equal if they are within a certain number of standard deviations, for larger noise levels the decision becomes more challenging. The approach that we propose is to instead calculate the probabilities of entries in $\mathbf{y}$ being zero/equal to each other *given that we observe noisy measurements* $\hat{\mathbf{y}}$. One can then calculate a score from these probabilities which estimates the likelihood that an update would correspond to decreasing the number of nonzeros in $\mathbf{r}$ if there were no additive noise. The goal is hence to estimate:

1) The probability of $\mathbf{y}_i = 0$ given that we observe $\hat{\mathbf{y}}_i$.
$$p_z(\omega) := \mathbb{P}(\mathbf{y}_i = 0 \mid \hat{\mathbf{y}}_i = \omega). \tag{3}$$
2) The probability of $\mathbf{y}_{i_1} = \mathbf{y}_{i_2}$ given that we observe $\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2}$.
$$p_e(\omega) := \mathbb{P}(\mathbf{y}_{i_1} = \mathbf{y}_{i_2} \mid \hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega) \tag{4}$$

Among our contributions are a series of approximations for (3)-(4) when the signals and measurements are generated according to the generating model given in Definition II.1. In what follows, we let $\mathbb{D}(\mathbb{R})$ be the set of distributions supported on $\mathbb{R}$. If $\mu \in \mathbb{D}(\mathbb{R})$, we write $z \sim \mu$ to denote that z was drawn according to the distribution μ. We also use the notation $\mathbf{v}_i \overset{\text{i.i.d.}}{\sim} \mu$ to denote that each $\mathbf{v}_i$ is drawn independently at random from μ. Finally, we use $U(S)$ to denote the uniform distribution over a set S.

Definition II.1 (Generating model $\mathrm{GM}(n, m, k, d, \mu, \nu)$)**.** Let $n, m, k, d \in \mathbb{N}$ be such that $k < m < n$ and $d \ll m$. Let $\mu, \nu \in \mathbb{D}(\mathbb{R})$. Then, the problem $(\mathbf{A}, \hat{\mathbf{y}})$ is drawn from the model $\mathrm{GM}(n, m, k, d, \mu, \nu)$ if $\mathbf{A} \in \{0,1\}^{m \times n}$ and $\hat{\mathbf{y}} \in \mathbb{R}^m$ are such that

1) each column of $\mathbf{A}$ has a support drawn uniformly at random from $[m]^{(d)}$;
2) $\mathrm{supp}(\mathbf{x})$ is drawn uniformly at random from $[n]^{(k)}$;
3) $\mathbf{x}_j \overset{\text{i.i.d.}}{\sim} \mu$ for each $j \in \mathrm{supp}(\mathbf{x})$;
4) $\boldsymbol{\eta}_i \overset{\text{i.i.d.}}{\sim} \nu$ for each $i \in [m]$;
5) $\boldsymbol{\eta}_i$ is independent of $\mathbf{x}_j$ for all $i \in [m], j \in \mathrm{supp}(\mathbf{x})$;
6) $\mathbf{y} = \mathbf{A}\mathbf{x}$ and $\hat{\mathbf{y}} = \mathbf{y} + \boldsymbol{\eta}$.

We write $(\mathbf{A}, \hat{\mathbf{y}}) \sim \mathrm{GM}(n, m, k, d, \mu, \nu)$ to denote problem instances drawn from this signal model.

Remark.
- It is important to note that a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ generated under the model presented in Definition II.1, is a (k, ε, d)-expander matrix with high probability, see [25], [26, Theorem 13.6].
- The first property in the above Definition implies that, when looking at a row in $\mathbf{A}$, the entries are Bernoulli(d/m) distributed. This will be used to characterise the distribution of entries in the product $\mathbf{A}\mathbf{x}$ later on.

Moreover, the generating model in Definition II.1 also allows us to define robust estimates for (3)-(4) for general noise and signal distributions and to any degree of accuracy under the assumption that these probability measures are available. From there we can define noisy analogues to the values n_e and n_z in (1)-(2) used in Parallel-ℓ_0 but which are robust to additive noise. The contributions of this work are two-fold: (i) to present principled ways to compute (3)-(4) in the case where the nonzeros in $\boldsymbol{\eta}$ and $\mathbf{x}$ are drawn from continuous probability distributions; (ii) to provide a variation of Parallel-ℓ_0 that is robust to noise. While other similar generating models can be considered using the techniques presented here, for ease of exposition and clarity, we restrict our description to this model.

Theorem II.2 (Probabilities for general signal and noise distributions)**.** Let $\delta, \rho \in (0,1)$. For each $n > 1$, let $m = \delta n$, $k = \rho m$ and $d \ll m$. If $\mu, \nu \in \mathbb{D}(\mathbb{R})$ and $(\mathbf{A}, \hat{\mathbf{y}}) \sim \mathrm{GM}(n, m, k, d, \mu, \nu)$. Then as $n \to \infty$,
$$p_z(\omega) \to \frac{\nu(\omega)}{\sum_{q \geq 0} \frac{(d\rho)^q}{q!} (\nu * \mu_q)(\omega)}, \tag{5}$$
$$p_e(\omega) \to \frac{\tilde{\nu}(\omega)}{\sum_{q \geq 0} \frac{(2d\rho)^q}{q!} (\tilde{\nu} * \bar{\mu}_q)(\omega)}, \tag{6}$$
where $\mu_q, \bar{\mu}_q, \tilde{\nu}$ are probability measures constructed as in Definition III.1.

Equations (5) and (6) allow us to quantify the uncertainty associated with computing the score for Parallel-ℓ_0 under the presence of additive noise. Note that equations (5) and (6) can be easily adapted to alternative generative models, such as where the expected density of nonzeros per row varies, but for expository clarity we restrict our discussion to this somewhat generic model. It will be discussed in Section III-C that suitably scalled equations (5) and (6) are used as proxies for (1) and (2), respectively, and may be used in their *direct* or *quantised* forms as defined by Table I. Robust-ℓ_0 (Algorithm 2) then works in the same way as Parallel-ℓ_0 but with the scores n_z and n_e replaced by q_e and q_z. Furthermore, two additional steps are added: an update is only accepted if it leads to a decrease of the residual in the $\|\cdot\|_1$ norm and after each loop hard thresholding on $\mathbf{x}$ is performed.

III. Derivation of the probability functions

In the previous section we discussed how to extend the

Algorithm 2: Robust-ℓ_0

Data: $\mathbf{A} \in \mathbb{E}^{m\times n}_{k,\varepsilon,d}$; $\mathbf{y} = \mathbf{A}\mathbf{x} \in \mathbb{R}^m$ for $\mathbf{x} \in \chi^n_k$; $\alpha \in (1, d]$; $\mu, \nu \in \mathbb{D}(\mathbb{R})$; $c \in (0, 1)$
Result: $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ s.t. $\hat{\boldsymbol{x}} \approx \mathbf{x}$
Estimate $\breve{p}_z = \breve{p}_z(d, k, m, n, \mu, \nu)$ as in (13);
Estimate $\breve{p}_e = \breve{p}_e(d, k, m, n, \mu, \nu)$ as in (13);
if `quantised` **then**
  Use *quantised* scores given in Table I.
else
  Use *continuous* scores given in Table I.
end
$\hat{\boldsymbol{x}} \leftarrow \mathbf{0}$;
$\hat{\boldsymbol{r}} \leftarrow \mathbf{y}$;
$t \leftarrow 1$;
while *not converged and* $t > 0$ **do**
  $\mathbf{x}' \leftarrow \hat{\boldsymbol{x}}$;
  $\mathbf{r} \leftarrow \hat{\boldsymbol{r}}$;
  for $j \in [n]$ **do**
    for $i \in \mathcal{N}(j)$ **do**
      if $1 - \breve{p}_z(\mathbf{r}_i) \geq t$ **then**
        $n_e \leftarrow \sum_{\ell\in\mathcal{N}(j)} q_e(\mathbf{r}_i - \mathbf{r}_\ell \mid t)$;
        $n_z \leftarrow \sum_{\ell\in\mathcal{N}(j)} q_z(\mathbf{r}_\ell \mid t)$;
        $\omega \leftarrow \frac{1}{n_e}\sum_{\ell\in\mathcal{N}(j)} \mathbf{r}_\ell q_e(\mathbf{r}_i - \mathbf{r}_\ell \mid t)$;
        if $\|\mathbf{r} - \omega\mathbf{e}_i\|_1 \leq \|\mathbf{r}\|_1$ *and* $n_e - n_z \geq \alpha$ **then**
          $\mathbf{x}'_j \leftarrow \mathbf{x}'_j + \omega$;
        end
      end
    end
  end
  $\mathbf{x}' \leftarrow \mathcal{H}_k(\mathbf{x}')$;
  $\mathbf{r} \leftarrow \mathbf{y} - \mathbf{A}\mathbf{x}'$;
  $t \leftarrow t - c$;
  if $\|\mathbf{r}\|_1 < \|\hat{\boldsymbol{r}}\|_1$ **then**
    $\hat{\boldsymbol{x}} \leftarrow \mathbf{x}'$;
    $\hat{\boldsymbol{r}} \leftarrow \mathbf{r}$;
    if `adaptive_k` **then**
      $k_0 \leftarrow k - \sum_{j\in[n]} \breve{p}_z(\hat{\boldsymbol{x}}_j)$;
      $k_0 \leftarrow \max(k_0, \lfloor \frac{m}{100} \rfloor)$;
      Recompute $\breve{p}_z = \breve{p}_z(k_0, m, n, \mu, \nu)$;
      Recompute $\breve{p}_e = \breve{p}_e(k_0, m, n, \mu, \nu)$;
    end
  end
end

Parallel-ℓ_0 algorithm to the case where the measurements $\mathbf{y}$ are subject to additive noise. Key to this extension are the functions $p_z(\hat{\mathbf{y}}_i)$ and $p_e(\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2})$ which correspond, respectively, to the probability of $\mathbf{y}_i = 0$ given that we observe $\hat{\mathbf{y}}_i$ and the probability of $\mathbf{y}_{i_1} = \mathbf{y}_{i_2}$ given that we observe $\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2}$. The functions p_z and p_e depend on the parameters fed into the generative model in Definition II.1 and in particular on the distribution of $\mathbf{y}$ and $\hat{\mathbf{y}}$. We include Definition III.1 in order to remind the reader of some actions on measures used in this manuscript in order to make it relatively self-contained.

Definition III.1 (Measures [27])**.** Let $\mathcal{B}(\mathbb{R})$ denote the Borel σ-algebra over $\mathbb{R}$. If $E \in \mathcal{B}(\mathbb{R})$ let $-E := \{-x : x \in E\}$. Let $\mu \in \mathbb{D}(\mathbb{R})$, we define the following measures.

1) The q-convolution,
$$\mu_0(E) = \delta_0(E) = \begin{cases} 1 & 0 \in E \\ 0 & 0 \notin E \end{cases}, \forall E \in \mathcal{B}(\mathbb{R})$$
$$\mu_q(E) = (\mu_{q-1} * \mu)(E), \forall E \in \mathcal{B}(\mathbb{R}), q \in \mathbb{N}$$
2) The negative measure,
$$\mu^-(E) = \mu(-E), \ \ \forall E \in \mathcal{B}(\mathbb{R})$$
3) The symmetrized measure,
$$\bar{\mu}(E) = \frac{\mu(E) + \mu(-E)}{2}, \ \ \forall E \in \mathcal{B}(\mathbb{R})$$
4) The measure associated with the difference of two random variables,
$$\tilde{\mu}(E) = (\mu * \mu^-)(E), \ \ \forall E \in \mathcal{B}(\mathbb{R}).$$

We can now express the limiting distribution of the residual as $n \to \infty$ in the terms of the previously defined measures.

Theorem III.2 (Distribution of $\hat{\mathbf{y}}_i$ and $\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2}$)**.** Fix $\delta, \rho \in (0, 1)$ and for $n \in \mathbb{N}$ let $m = \delta n$, $k = \rho m$ and $d \ll m$. Furthermore, let μ and ν be measures and assume that $\hat{\mathbf{y}} = \mathbf{y}+\boldsymbol{\eta} = \mathbf{A}\mathbf{x}+\boldsymbol{\eta}$ is drawn from the model $\mathrm{GM}(n, m, k, d, \mu, \nu)$. Then as $n \to \infty$
$$\hat{\mathbf{y}}_i \overset{(d)}{\to} \hat{\mathbf{y}}^*_i \quad \text{and} \quad \hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} \overset{(d)}{\to} \hat{g}^*$$
where
$$\hat{\mathbf{y}}^*_i \sim \exp(-d\rho) \sum_{q\geq 0} \frac{(d\rho)^q}{q!} \nu * \mu_q, \tag{7}$$
$$\hat{g}^* \sim \exp(-2d\rho) \sum_{q\geq 0} \frac{(2d\rho)^q}{q!} \tilde{\nu} * \bar{\mu}_q. \tag{8}$$

Proof. See the appendix. □

Theorem III.2 follows from entries in $\mathbf{y}$ being the inner product of rows of $\mathbf{A}$ and the signal $\mathbf{x}$, both of which are sparse.

A. Explicit formulas for the centred Gaussian case

Let $\mathcal{N}(\mu, \sigma^2)$ denote the Gaussian distribution with mean μ and variance σ^2. We further elucidate (5)-(6) from Theorem II.2 in the case when μ and ν are Gaussian with mean zero, which are the distributions considered in Section IV. To this end, let $\mu = \mathcal{N}(0, \sigma_s^2)$ and $\nu = \mathcal{N}(0, \sigma_n^2)$. We observe that in this case $\bar{\mu} = \mu$, $\mu_q = \bar{\mu}_q = \mathcal{N}(0, q\sigma_s^2)$ and $\tilde{\nu} = \mathcal{N}(0, 2\sigma_n^2)$. Hence, denoting by $\varphi(\cdot \mid \sigma^2)$ the probability density function of a centred Gaussian random variable with variance σ^2, we obtain
$$p_z(\omega) \to \frac{\varphi(\omega \mid \sigma_n^2)}{\sum_{q\geq 0} \frac{(d\rho)^q}{q!} \varphi(\omega \mid q\sigma_s^2 + \sigma_n^2)} \tag{9}$$
$$p_e(\omega) \to \frac{\varphi(\omega \mid 2\sigma_n^2)}{\sum_{q\geq 0} \frac{(2d\rho)^q}{q!} \varphi(\omega \mid q\sigma_s^2 + 2\sigma_n^2)}. \tag{10}$$

1) Estimating the tails in the Gaussian case: We approximate the infinite sum in in the denominators of (9) and (10) by doing an approximation to the tail of this summation.

Lemma III.3 (Sums of centred density functions)**.** Let μ_i be the density function of a random variable with mean zero and variance σ_i^2 and let $\alpha_i > 0$ be such that $\sum_i \alpha_i = 1$. Then, $\mu = \sum_i \alpha_i \mu_i$ is the density function of a random variable with mean zero and variance $\sum_i \alpha_i \sigma_i^2$.

Proof. See the appendix. □

To simplify notation let $\sigma_q^2 = q\sigma_s^2 + \sigma_n$ for $q \in \mathbb{N} \cup \{0\}$. Consider the series in the denominator of (30)

$$S(\omega) := \sum_{q=0}^{\ell} \frac{(d\rho)^q}{q!} \varphi(\omega \mid \sigma_q^2) + \sum_{q=\ell+1}^{\infty} \frac{(d\rho)^q}{q!} \varphi(\omega \mid \sigma_q^2).$$

Let,

$$R_z(\ell) := \exp(d\rho) - \sum_{q=0}^{\ell} \frac{(d\rho)^q}{q!}$$

and write

$$S_a(\omega) := \sum_{q=0}^{\ell} \frac{(d\rho)^q}{q!} \varphi(\omega \mid \sigma_q^2),$$
$$S_b(\omega) := \sum_{q=\ell+1}^{\infty} \frac{(d\rho)^q}{q!} \varphi(\omega \mid \sigma_q^2).$$

Note that S_b/R_z satisfies the conditions of Lemma III.3 so it corresponds to the density function of a centred random variable with variance

$$\sigma_z^2 = \frac{1}{R_z} \sum_{q=\ell+1}^{\ell} \frac{(d\rho)^q}{q!} (q\sigma_s^2 + \sigma_n^2)$$
$$= \frac{\sigma_s^2 (d\rho) R_z(\ell-1) + \sigma_n^2 R_z(\ell)}{R_z(\ell)}$$

Therefore,

$$p_z(\omega) \approx \frac{\varphi(\omega \mid \sigma_n^2)}{\sum_{q=0}^{\ell} \frac{(d\rho)^q}{q!} \varphi(\omega \mid \sigma_q^2) + R_z(\ell) \varphi(\omega \mid \sigma_z^2)}. \quad (11)$$

A similar argument shows that

$$p_e(\omega) \approx \frac{\varphi(\omega \mid 2\sigma_n^2)}{\sum_{q=0}^{\ell} \frac{(2d\rho)^q}{q!} \varphi(\omega \mid \sigma_{q,e}^2) + R_e(\ell) \varphi(\omega \mid \sigma_e^2)}, \quad (12)$$

where $\sigma_{q,e}^2 = q\sigma_s^2 + \sigma_n^2$, and

$$R_e(\ell) := \exp(2d\rho) - \sum_{q=0}^{\ell} \frac{(2d\rho)^q}{q!},$$
$$\sigma_e^2 = \frac{\sigma_s^2 (2d\rho) R_e(\ell-1) + 2\sigma_n^2 R_e(\ell)}{R_e(\ell)}.$$

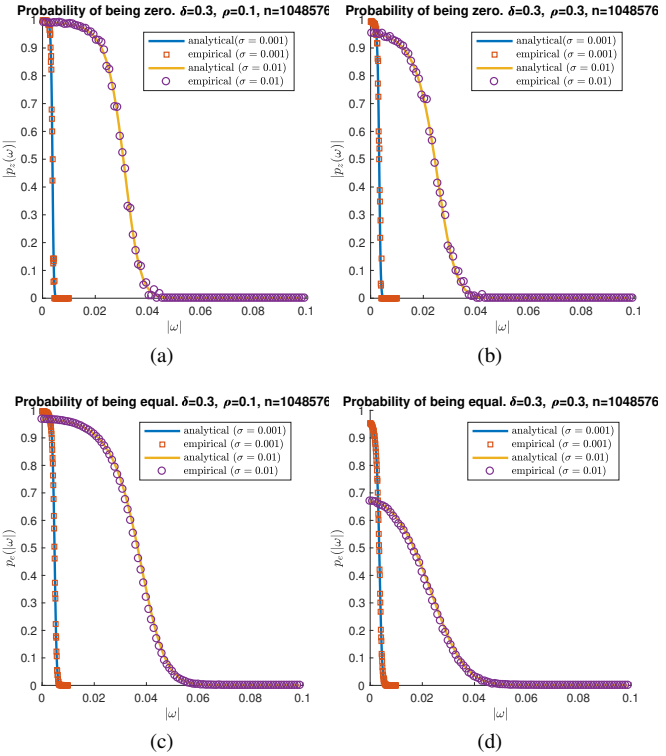


Fig. 1: Comparison of analytical and empirical probabilities of a value in the residual being zero or two values in the residual being equal for $\rho \in \{0.1, 0.3\}$ and $\sigma_n \in \{10^{-3}, 10^{-2}\}$.

B. Comparison with empirical probabilities

We test the approximations given in (11) and (12) by randomly generating $\hat{\mathbf{y}}$ and $\mathbf{y}$ according the generating models $\mathrm{GM}(n, \delta n, \rho\delta n, 7, \mathcal{N}(0,1), \mathcal{N}(0,\sigma^2))$, for $\delta = 0.3$, $\rho \in \{0.1, 0.3\}$ and $\sigma \in \{10^{-3}, 10^{-2}\}$. The results can be seen in Figure 1.

Overall the analytical expressions fit the empirical probabilities very well, indicating that the approximations we made in the calculations above are justified. However, we observe that as ρ and especially σ increase, both functions drop significantly. This means that for large values of these parameters, the noise eventually dominates and it is difficult to decided whether values are zero or equal.

C. Scaled probabilities $\breve{p}_z$ and $\breve{p}_e$

We mentioned in Section I that our algorithms do not implement the functions p_z and p_e exactly, but a scaled version of these. As observed in Figure 1, the value of $\max_s p_e(s)$ and $\max_s p_z(s)$ varies significantly as σ and ρ change, so to account for these variations we work with *normalised* functions $\breve{p}_e$ and $\breve{p}_z$ defined as,

$$\breve{p}_e(\omega) = \frac{p_e(\omega)}{\max_s p_e(s)}, \quad \breve{p}_z(\omega) = \frac{p_z(\omega)}{\max_s p_z(s)}. \quad (13)$$

In the most general case n_e and n_z can be written as the sum of individual scores q_e and q_z as follows

$$n_e \leftarrow \sum_{\ell \in \mathcal{N}(j)} q_e\left(\mathbf{r}_{i_1} - \mathbf{r}_{i_2} \mid t\right) \tag{14}$$

$$n_z \leftarrow \sum_{\ell \in \mathcal{N}(j)} q_z\left(\mathbf{r}_i \mid t\right) \tag{15}$$

A confidence threshold $t > 0$ needs to be given for some variants of our algorithm, so to simplify the exposition we include this parameter in the scores $q_e(\cdot \mid t)$ and $q_z(\cdot \mid t)$ regardless of whether it is used or not. A summary of the score functions are given in Table I.

	Robust-ℓ_0 Continuous	Robust-ℓ_0 Quantised	Parallel-ℓ_0
$q_e(\mathbf{r}_{i_1} - \mathbf{r}_{i_2} \mid t)$	$\breve{p}_e(\mathbf{r}_{i_1} - \mathbf{r}_{i_2})$	$\mathbb{1}_{\{\breve{p}_e(\mathbf{r}_{i_1} - \mathbf{r}_{i_2}) \geq t\}}$	$\mathbb{1}_{\{\mathbf{r}_{i_1} = \mathbf{r}_{i_2}\}}$
$q_z(\mathbf{r}_i \mid t)$	$\breve{p}_z(\mathbf{r}_i)$	$\mathbb{1}_{\{\breve{p}_z(\mathbf{r}_i) \geq 1 - t\}}$	$\mathbb{1}_{\{\mathbf{r}_i = 0\}}$

TABLE I: Extensions of scores used in Expander ℓ_0-decoding to identify candidate updates to the sparse signal $\hat{\mathbf{x}}$.

Algorithm 2 evaluates whether a given score is large or not by implementing a sweeping parameter t that is set to one at the beginning of the algorithm and decreased by a constant c after every iteration. In order to use a fixed initial t we consider the scaled probabilities $\breve{p}_e$ and $\breve{p}_z$ in (13); otherwise the initial value of t would depend on σ and ρ.

D. Adaptive k

Algorithm 2 can optionally account for the sparsity of the current estimate via the flag `adaptive_k`. In the noiseless model of $\mathbf{r} = \mathbf{A}\hat{\mathbf{x}}$, an update of the form $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \omega \mathbf{e}_j$ with Parallel-ℓ_0 guarantees that $j \in \operatorname{supp}(\mathbf{x})$ so that at the next iteration the problem with residual $\mathbf{r} - a_j\omega$ and $(k-1)$-sparse signal is considered. The `adaptive_k` flag updates the sparsity prior in $\hat{\mathbf{x}}$ after every update in hope of having more reliable estimates of p_e and p_z. We will see in the numerical experiments that this parameter does not bring substantial benefits when used with the data generating model that we tested this strategy under. However, we do not rule out the possibility that there are other signal and noise distributions for which this flag becomes especially useful, so we retain it for the benefit of practitioners.

IV. Numerical experiments

In this section we present numerical experiments which validate the efficacy of Robust-ℓ_0 decoding. In particular, we contrast Robust-ℓ_0 with other state-of-the-art greedy algorithms for compressed sensing in terms of their ability to recover the measured signal for varying problem sizes (k, m, n) as well as their computational complexity. To facilitate reproducibility, we begin by describing the stopping conditions and measures used to denote successful recovery in the presence of noise in Section IV-A, along with how the parameter c is varied in Section IV-B. We then present the main numerical results in Section IV-C where the algorithms phase transitions and runtime are presented, along with Sections IV-D and IV-E which show further details on Robust-ℓ_0 decoding's performance as a function of noise variance and for extreme subsampling respectively.

A. Stopping conditions

We are interested in the signal model $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 I_{m\times m})$. If $\hat{\mathbf{x}}$ is an approximation to $\mathbf{x}$, the residual is $\mathbf{r} = \mathbf{y} - \mathbf{A}\hat{\mathbf{x}}$. Note that if $\hat{\mathbf{x}} = \mathbf{x}$, then

$$\begin{aligned} \|\mathbf{r}\|_1 &= \|\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}\|_1 \\ &= \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_1 \\ &= \|\boldsymbol{\eta}\|_1, \end{aligned}$$

so we should not seek reductions in the residual below $\|\boldsymbol{\eta}\|_1$ since these would result in over-fitting. We further account for the variance of $\|\boldsymbol{\eta}\|_1$ and denote the algorithm to have successfully recovered $\mathbf{x}$ if $\hat{\mathbf{x}}$ satisfies

$$\frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_1}{\|\mathbf{x}\|_1} \leq \frac{\mathbb{E}\left[\|\boldsymbol{\eta}\|_1\right] + C_1\sqrt{\operatorname{Var}\left[\|\boldsymbol{\eta}\|_1\right]}}{\|\mathbf{x}\|_1} \tag{16}$$

for some $C_1 \geq 0$. We should be aware that the right hand side of (16) might be greater than 1 for some choices of k, m and σ. When this happens, the stopping condition (16) becomes invalid since we expect $\hat{\mathbf{x}}$ to have captured a proportion of the ℓ_1-energy of $\|\mathbf{x}\|_1$. Hence, if the right hand side of (16) is greater than $\frac{1}{10}$ we clip the upper bound at this value and use the stopping condition

$$\frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_1}{\|\mathbf{x}\|_1} \leq \min\left(\frac{\mathbb{E}\left[\|\boldsymbol{\eta}\|_1\right] + C_1\sqrt{\operatorname{Var}\left[\|\boldsymbol{\eta}\|_1\right]}}{\|\mathbf{x}\|_1}, \frac{1}{10}\right). \tag{17}$$

For the numerical experiments conducted in this section we consider nonzero entries in $\mathbf{x}$ drawn as $\mathbf{x}_i \sim \mathcal{N}(0, 1)$, and noise $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2)$ for which $\mathbb{E}\left[\|\mathbf{x}\|_1\right] = k\sqrt{\frac{2}{\pi}}$ and $\mathbb{E}\left[\|\boldsymbol{\eta}\|_1\right] = m\sigma\sqrt{\frac{2}{\pi}}$ and moreover $\operatorname{Var}\left[\|\boldsymbol{\eta}\|_1\right] = m\sigma^2\left(1 - \frac{2}{\pi}\right)$, see e.g. [28].

B. Selection of parameter c in Algorithm 2

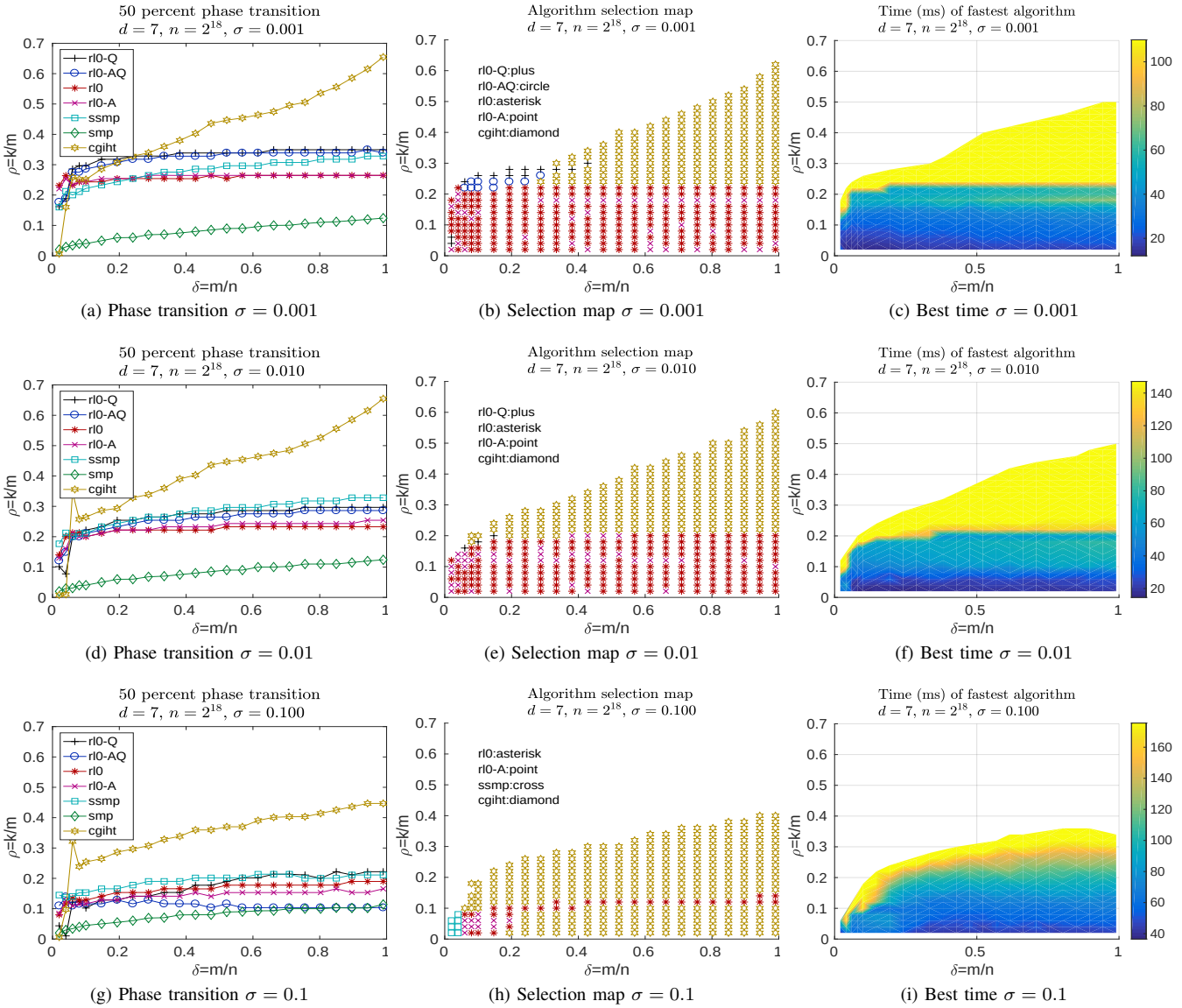
The sweeping parameter t in Algorithm 2 is initialised at 1 and updated by decreasing it by a constant value c. We observed in our experiments that the quality of the phase transitions are sensitive on the parameter c especially for low δ and ρ. We do not provide a way to fine-tune c, but we run our phase transitions with the following choices:

1) If the algorithm is `quantised`,

$$c = \begin{cases} 0.01 & \delta \leq 0.05 \\ 0.05 & \delta > 0.05 \text{ and } \rho \in (0, 0.1] \\ 0.075 & \delta > 0.05 \text{ and } \rho \in (0.1, 0.2] \\ 0.1 & \delta > 0.05 \text{ and } \rho \in (0.2, 1) \end{cases}. \tag{18}$$

2) If the algorithm is `continuous`,

$$c = \begin{cases} 0.01 & \delta \leq 0.05 \\ 0.025 & \delta > 0.05 \end{cases}. \tag{19}$$



(a) Phase transition $\sigma = 0.001$ (b) Selection map $\sigma = 0.001$ (c) Best time $\sigma = 0.001$

(d) Phase transition $\sigma = 0.01$ (e) Selection map $\sigma = 0.01$ (f) Best time $\sigma = 0.01$

(g) Phase transition $\sigma = 0.1$ (h) Selection map $\sigma = 0.1$ (i) Best time $\sigma = 0.1$

Fig. 2: Phase transitions, selection maps and timings for $n = 2^{18}$.

Algorithm	Label	`adaptive_k`	`quantised`
Robust-ℓ_0	rl0	No	No
Robust-ℓ_0-adaptive	rl0-A	Yes	No
Robust-ℓ_0-quantised	rl0-Q	No	Yes
Robust-ℓ_0-adaptive-quantised	rl0-AQ	Yes	Yes

TABLE II: Variants of Robust-ℓ_0

The values in (18) and (19) were chosen heuristically for ν and μ Gaussian.

C. Phase transitions and runtime

We benchmark the variants of Robust-ℓ_0 against other greedy algorithms via their *phase-transitions and runtime*. The user can supply two binary flags, `adaptive_k` and `quantised` which yield four different variants of Robust-ℓ_0. We assigned a unique label to each of these variants as described in Table II.

The phase transition of a compressed-sensing algorithm [29] is the largest value of k/m for which the algorithm is typically able recovery all k sparse vectors with sparsity less than k for a fixed m/n. Hence, for a fixed value of $\delta = m/n$ the phase transition of an algorithm is the largest value $\rho^*(\delta)$ for which the algorithm converges for all $\rho(\delta) < \rho^*(\delta)$. The value $\rho^*(m/n)$ often converges to a fixed value as $n \to \infty$, so phase transitions often partition the $\delta \times \rho$ space into two regions: One in which the algorithm converges with high probability and another in which the algorithm doesn't converge with high probability. We benchmark Robust-ℓ_0 against the algorithms presented in [21], [22], [30]. Specifically, our tests include the following algorithms,

$$\{\text{Robust-}\ell_0,\ \text{Robust-}\ell_0\text{-adaptive},\ \text{Robust-}\ell_0\text{-quantised},\ \text{Robust-}\ell_0\text{-adaptive-quantised},\ \text{SSMP},\ \text{SMP},\ \text{CGIHT}\}.$$

In the deterministic case Parallel-ℓ_0 was compared against a range of combinatorial compressed sensing algorithms in

[1]; out of those we have selected SSMP and SMP as these perform best and are similar in nature to Robust-ℓ_0. We also compare with CGIHT, as this algorithm was shown to be the fastest among the greedy algorithms compared in [13], [30]. Figures 2a, 2d and 2g show the phase transition curves for these algorithms with $\sigma = 10^{-3},\ 10^{-2},\ 10^{-1}$ respectively. The curves were computed by setting $n = 2^{18}$, $d = 7$, and using the stopping condition $\|\mathbf{r}\|_1 \leq \mathbb{E}\left[\|\boldsymbol{\eta}\|_1\right] = m\sigma\sqrt{\frac{2}{\pi}}$ and a success condition (16) with $C_1 = 1$. The testing is done at $m = \delta_p n$ for

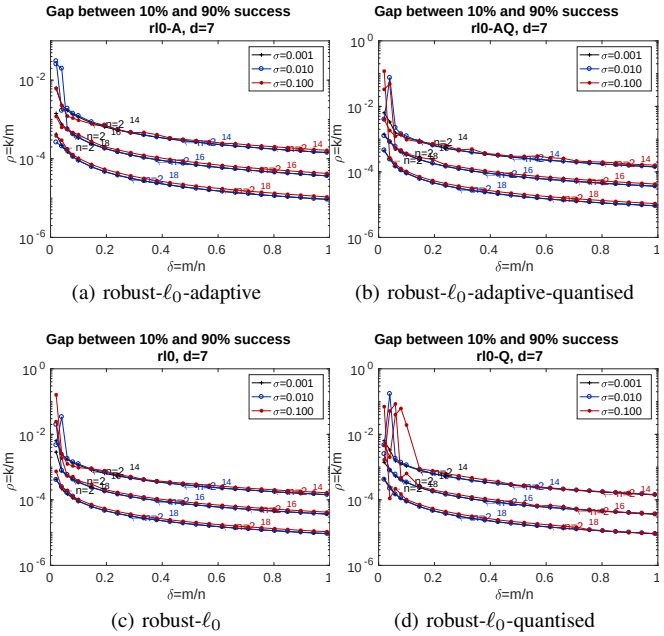
$$\delta_p \in \{0.02p : p \in [4]\} \cup \left\{0.1 + \frac{89}{1900}(p-1) : p \in [20]\right\}.$$

For each δ_p, we set $\rho = 0.01$ and generate 10 synthetic problems to apply the algorithms to, with a problem generated as in GM with the given parameters and μ and ν being normal Gaussian. If at least one such problem was recovered successfully, the sparsity ratio ρ is increased by 0.01 and the experiment is repeated. Following the testing framework in [31], the recovery data is fitted using a logistic function and finally the 50% recovery transition function is computed and presented in the phase transition plots contained herein. Figures 2b, 2e and 2h show a selection map for these algorithms. Namely, these plots indicate which algorithm requires the least computational time[1] at each point in the $\delta \times \rho$ space where the algorithm converges. Finally, Figures 2c, 2f and 2i show the total time for convergence in milliseconds for the fastest algorithm at each point in the $\delta \times \rho$ space.

We can see from Figure 2 that CGIHT [30] dominates the upper region of the phase transition space, while the Robust-ℓ_0 algorithms only converge for $\rho \lessapprox 0.3$ which is consistent with the observed phase transitions for Parallel-ℓ_0 [1]. In terms of speed, Robust-ℓ_0 seems to be the most competitive for $\sigma \in \{10^{-3}, 10^{-2}\}$ and $\rho \lessapprox 0.2$. However, for large noise, $\sigma = 10^{-1}$, CGIHT becomes the fastest algorithm of all. We remark that while CGIHT performs very well in our numerical tests, the current theory developed for it does not hold in the setting considered here, as it requires zero-mean columns in A.

Figure 3 shows the widths for the Robust-ℓ_0 algorithms. The widths measure how sharp the phase transition of an algorithm is; namely, how thin the boundary between the region of recovery with high-probability and the region of recovery where combinatorial search is needed. It has been shown that the widths of a compressed sensing algorithm tend to zero as $n \to \infty$ when decoding with linear programming [33], and we usually expect the same behaviour for other algorithms [30]. Figure 3 show that the widths for the Robust-ℓ_0 algorithms indeed decrease with n and with δ. The observed smoothness of the phase transition widths signal also suggest that the stopping conditions of the algorithm are consistent for the problem under consideration.

[1]All the numerical results presented here were performed using a Linux machine with Intel Xeon E5-2643 CPUs 3.30 GHz, NVIDIA Tesla K10 GPUs and executed from Matlab R2016b. The code was added to the GAGA library available at `http://www.gaga4cs.org/`, and described in [32], so as to facilitate large scale benchmarking against the other algorithms presented here which are also contained in the aforementioned library.

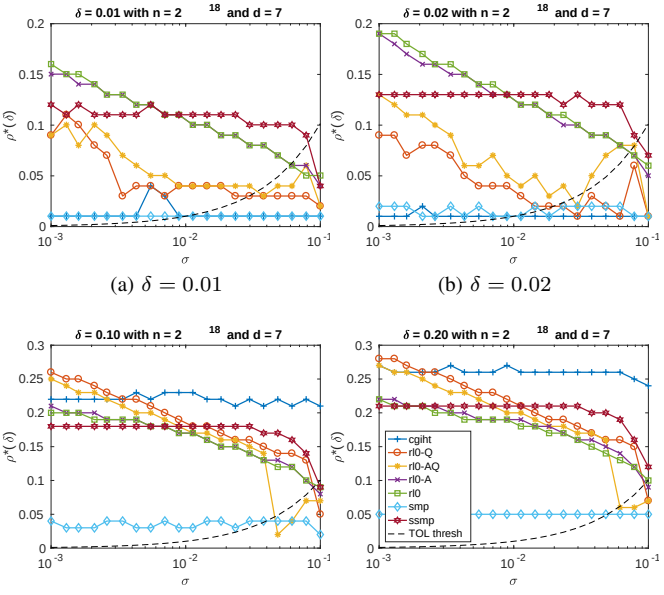


(a) robust-ℓ_0-adaptive (b) robust-ℓ_0-adaptive-quantised

(c) robust-ℓ_0 (d) robust-ℓ_0-quantised

Fig. 3: Widths for Robust-ℓ_0 variants

D. Dependence on noise variance, σ



(a) $\delta = 0.01$ (b) $\delta = 0.02$

(c) $\delta = 0.10$ (d) $\delta = 0.20$

Fig. 4: Decrease in phase transition for varying σ.

We now investigate the extent to which the phase transitions of the algorithm decrease as we increase the noise level σ. To do this, we consider $\delta \in \{0.01, 0.02, 0.1, 0.2\}$ and for each value of δ we compute we define the grid

$$\sigma \in \{10^{-3+\frac{i}{10}} : i \in \{0\} \cup [20]\}.$$

Then at each value of σ we let $\rho = 0.01$ and draw ten problem instances from GM with signal distribution $\mathcal{N}(0,1)$ and noise distribution $\mathcal{N}(0,\sigma^2)$. If at least one of the problems was

recovered successfully, then we set $\rho \leftarrow \rho + 0.01$ and repeat the experiment. We do this procedure for each of the algorithms considered and record the largest ρ having at least 50% success rate. The results are shown in Figure 4. In order to show where the clipping in (17) becomes active, the figures also show a TOL-curve which partitions the space into the region where the right hand side of (17) equals $\frac{1}{10}$ (bottom region) and the region where it equals the right hand side of (16) (top region). We can appreciate from Figure 4 that for small δ both Robust-ℓ_0 and SSMP have the best recovery capabilities, with Robust-ℓ_0 being preferable for noise levels $\sigma \lessapprox 10^{-2}$ and SSMP being better suited for larger noise levels. For larger δ, CGIHT is preferable except for very low noise levels.

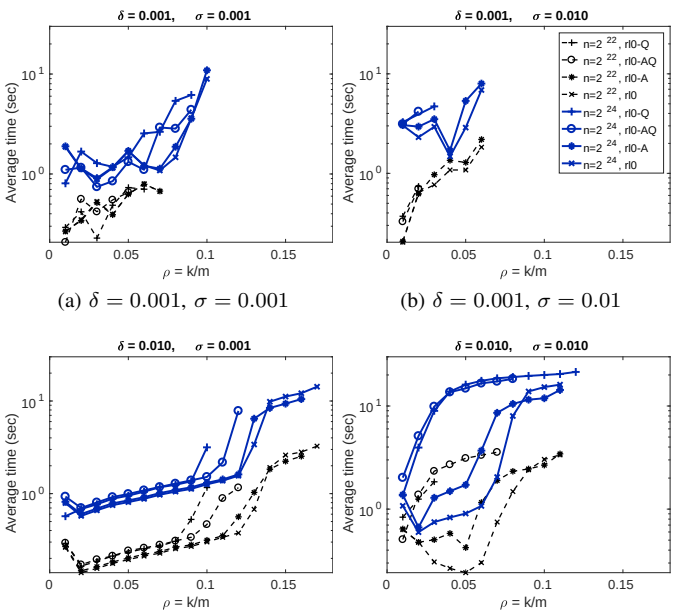
E. Phase transitions for extreme subsampling, $\delta \ll 1$

The numerical experiments of Parallel-ℓ_0 in [1] showed flat phase transitions; that is, it was observed that $\rho^*(\delta)$ remained approximately 0.3 as $\delta \to 0$ provided n was sufficiently large. While Robust-ℓ_0 does not exhibit precisely the same behaviour in the presence of noise, we do observe that $\rho^*(\delta)$ remains nontrivial even for δ as small as 10^{-3}, again provided n is sufficiently large. We provide numerical evidence for this in Figure 5. For each $(\delta, \sigma) \in \{0.001, 0.01\} \times \{0.001, 0.01\}$ we let $\rho = 0.01$ and solve ten problems drawn from GM with nonzero distribution $\mathcal{N}(0,1)$ and noise distribution $\mathcal{N}(0, \sigma^2)$. If at least one problem instance converges, we average the run-time of the problems that converged and repeat the process with $\rho \leftarrow \rho + 0.01$. We plot the timing at each ρ for parameter values for which at which at least 50% of the problems were successfully recovered under the criteria (17).

It can be seen in Figure 5a-5d that the phase transition either remains nearly unchanged or increases as n increases from 2^{22} to 2^{24}. In particular, Figure 5a shows that for $\sigma = 0.001$ and $\delta = 0.001$, the phase transitions for all variants of the Robust-ℓ_0 algorithms in fact increase from $\rho \approx 0.08$ to $\rho \approx 0.1$ as n increases from 2^{22} to 2^{24}. Additionally, contrasting Figures 5a and 5b or 5c and 5d shows that a ten-fold increase in σ has the expected effect of reducing the phase transition and increasing the computational time. Figures 5a and 5b show the phase transition for Robust-ℓ_0 and Robust-ℓ_0-adaptive remain significant even for δ as small as 10^{-3}. Figures 5c and 5d show results for the same set of experiments, but for $\delta = 0.01$ which corresponds to a ten-fold increase in δ over the value used in Figures 5a and 5b. For $\sigma = 0.001$ the phase transition of Robust-ℓ_0 reaches $\rho \approx 0.17$, while for $\sigma = 0.01$ the phase transitions drop to $\rho \approx 0.12$ and there is an increase in the computational time.

V. Conclusions

We have shown that the decoding framework presented in [1] can be extended to the case where the measurements are corrupted by additive noise. This framework is extended by deriving the posterior distribution of an entry in the residual being zero or being equal to another residual entry given the corrupted measurements. This Bayesian approach to decoding was implemented in Robust-ℓ_0 and its four variants. We show that the resulting algorithms inherits some desirable properties



(a) $\delta = 0.001$, $\sigma = 0.001$ (b) $\delta = 0.001$, $\sigma = 0.01$

(c) $\delta = 0.01$, $\sigma = 0.001$ (d) $\delta = 0.01$, $\sigma = 0.01$

Fig. 5: Phase transitions and timing Robust-ℓ_0 for $\delta \ll 1$.

from Parallel-ℓ_0 like high phase transitions for low δ and large n and low-latency. However, these qualities are weakened by the corruption of the measurements. Our numerical experiments show that Robust-ℓ_0 should be considered in cases of moderate noise and $\rho \lessapprox 0.3$.

Acknowledgments

The authors would like to thank William Carson for many helpful discussions on imaging applications of this work. RMS acknowledges the support of CONACyT. This work was supported by The Alan Turing Institute under the EPSRC grant EP/N510129/1.

Appendix

Definition A.1 (Dissociated signals [19])**.** A signal $\mathbf{x} \in \mathbb{R}^n$ is said to be dissociated if

$$\sum_{j \in S_1} \mathbf{x}_j \neq \sum_{j \in S_2} \mathbf{x}_j \qquad \forall S_1,\ S_2 \subset \operatorname{supp}(\mathbf{x}) \text{ s.t. } S_1 \neq S_2.$$

A. Limiting distribution for sparse sums of random variables

Lemma A.2. Let $p > 0$, let $\mu \in \mathbb{D}(\mathbb{R})$ and let $\mu_q \in \mathbb{D}(\mathbb{R})$ be its the q-fold convolution. For each $n \geq 1$, let

$$s_n := \sum_{j=1}^{n} b_j x_j \tag{20}$$

be such that,

1) $x_j \overset{\text{i.i.d.}}{\sim} \mu$ for each $j \in [n]$,
2) $b_j \overset{\text{i.i.d.}}{\sim} \text{Ber}(\frac{p_n}{n})$ for each $j \in [n]$ with $p_n \to p \in \mathbb{R}$ as $n \to \infty$.

Then, as $n \to \infty$ it holds that $s_n \overset{(d)}{\to} s$ where

$$s \sim \exp(-p) \sum_{q \geq 0} \frac{p^q}{q!} \mu_q. \tag{21}$$

Remark. The setting in this Lemma is different from the classical central limit theorem as we consider a in some sense *sparse* sum of independent random variables. This also means that no normalization as in the CLT is necessary, as the sparsity behaves like p/n for $n \to \infty$ and hence the sum is taken over only finitely many non-zero values. Moreover, Lem. A.2 considers *rows* with entries drawn $b_j \overset{\text{i.i.d.}}{\sim} \mathrm{Ber}(\frac{p_n}{n})$ which follows from the generative model Def. II.1 where *columns* have support drawn uniform from $[n]^{(k)}$; note this is particular to the generative model considered here and not true for arbitrary expander matrices.

Proof of Lemma A.2. Let $\psi_{s_n}(t)$ be the characteristic function of s_n. Let $\mathbf{x} \sim \mu$, then

$$\begin{aligned}\psi_{s_n}(t) &= \mathbb{E}\left[\exp(its_n)\right] \\ &= \prod_{j=1}^{n} \mathbb{E}\left[\exp(itb_j\mathbf{x}_j)\right] \\ &= \left(\left(1-\frac{p_n}{n}\right) + \left(\frac{p_n}{n}\right)\psi_{\mathbf{x}}(t)\right)^n \\ &= \left(1 + \frac{p_n(\psi_{\mathbf{x}}(t)-1)}{n}\right)^n.\end{aligned}$$

Taking the limit $n \to \infty$ we see that

$$\lim_{n\to\infty} \psi_{s_n}(t) = \exp(-p)\exp(p\psi_{\mathbf{x}}(t)). \tag{22}$$

Letting $w_q = \sum_{j=1}^{q} \mathbf{x}_j$ it holds by the independence of $\{\mathbf{x}_1, \dots, \mathbf{x}_q\}$ that $w_q \sim \mu_q$ and

$$[\psi_{\mathbf{x}}(t)]^q = \psi_{w_q}(t). \tag{23}$$

Now, consider a random variable z distributed according to

$$z \sim \exp(-p)\sum_{q\geq 0} \frac{p^q}{q!}\mu_q. \tag{24}$$

The characteristic function of z is given by

$$\begin{aligned}\psi_z(t) &= \mathbb{E}\left[\exp(itz)\right] \\ &= \exp(-p)\sum_{q\geq 0}\frac{p^q}{q!}\psi_{w_q}(t) \\ &= \exp(-p)\sum_{q\geq 0}\frac{p^q}{q!}\left(\psi_{\mathbf{x}}(t)\right)^q \\ &= \exp(-p)\exp(p\psi_{\mathbf{x}}(t)).\end{aligned} \tag{25}$$

Therefore (25) equals (22). By Lévy's continuity Theroem pointwise convergence of the characteristic functions implies weak convergence of the random variables (cf. [27, Theorem 15.23]) and hence the statement follows. □

Proof. To show (7), let $i \in [m]$ and

$$\mathbf{y}_i = \sum_{j=1}^{n} \mathbf{A}_{i,j}\mathbf{x}_j.$$

By our assumptions on $\mathbf{A}$ and $\mathbf{x}$,

$$\begin{aligned}\mathbb{P}\left(\mathbf{A}_{i,j}\mathbf{x}_j \neq 0\right) &= \mathbb{P}\left(\mathbf{A}_{i,j} \neq 0 \wedge \mathbf{x}_j \neq 0\right) \\ &= \mathbb{P}\left(\mathbf{A}_{i,j} \neq 0\right)\mathbb{P}\left(\mathbf{x}_j \neq 0\right) \\ &= \frac{d}{m}\frac{k}{n} = \frac{d\rho}{n}\end{aligned}$$

where for two events E_1 and E_2 we let $E_1 \wedge E_2$ be the conjunction of the events. Note that if $\mathbf{A}_{i,j}\mathbf{x}_j \neq 0$, then $j \in \mathrm{supp}(\mathbf{x})$ so $\mathbf{A}_{i,j}\mathbf{x}_j = \mathbf{x}_j \sim \mu$. Hence, letting $b_j \sim \mathrm{Ber}\left(\frac{d\rho}{n}\right)$,

$$\mathbf{y}_i \overset{(d)}{=} \sum_{j=1}^{n} b_j\mathbf{x}_j.$$

Invoking Lemma A.2 with $p_n = p = d\rho$ we obtain that $\mathbf{y}_i \to \mathbf{y}_i^*$ as $n \to \infty$ where

$$\mathbf{y}_i^* \sim \exp(-d\rho)\sum_{q\geq 0}\frac{(d\rho)^q}{q!}\mu_q.$$

By the independence of $\mathbf{y}_i^*$ and $\boldsymbol{\eta}_i$, since $\hat{\mathbf{y}}_i = \mathbf{y}_i + \boldsymbol{\eta}_i$, the distribution of $\hat{\mathbf{y}}_i^*$ is given by

$$\hat{\mathbf{y}}_i^* \sim \left(\exp(-d\rho)\sum_{q\geq 0}\frac{(d\rho)^q}{q!}\mu_q\right) * \nu. \tag{26}$$

Equation (7) follows from (26) and the distributivity of the convolution operator.

To show (8), let $i_1, i_2 \in [m]$ be such that $i_1 \neq i_2$ and let

$$\mathbf{y}_{i_1} - \mathbf{y}_{i_2} = \sum_{j=1}^{n}\left(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j}\right)\mathbf{x}_j.$$

Similarly to the previous case, we compute

$$\begin{aligned}\mathbb{P}\left(\left(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j}\right)\mathbf{x}_j \neq 0\right) &= \mathbb{P}\left(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j} \neq 0 \wedge \mathbf{x}_j \neq 0\right) \\ &= \mathbb{P}\left(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j} \neq 0\right)\mathbb{P}\left(\mathbf{x}_j \neq 0\right) \\ &= 2\frac{d}{m}\frac{(m-1)-(d-1)}{m-1}\frac{k}{n} \\ &= \frac{2d\rho}{n}\left(1-o(1)\right)\end{aligned}$$

Note that if $(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j})\mathbf{x}_j \neq 0$, then $j \in \mathrm{supp}(\mathbf{x})$ and $(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j})$ is either $+1$ with probability $\frac{1}{2}$ or -1 with probability $\frac{1}{2}$. Then, Hence,

$$(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j})\mathbf{x}_j \sim \begin{cases} \mu & \text{with probability } \frac{1}{2}, \\ \mu^- & \text{with probability } \frac{1}{2}, \end{cases}$$

then,

$$(\mathbf{A}_{i_1,j} - \mathbf{A}_{i_2,j})\mathbf{x}_j \sim \bar{\mu}.$$

Letting $b'_j \sim \mathrm{Ber}\left(\frac{2d\rho}{n}(1-o(1))\right)$,

$$\mathbf{y}_{i_1} - \mathbf{y}_{i_2} \overset{(d)}{=} \sum_{j=1}^{n} b'_j\mathbf{x}_j.$$

Again, invoking Lemma A.2 with $p_n = 2d\rho(1-o(1))$ we obtain that $p = 2d\rho$ and also that as $n \to \infty$, $\mathbf{y}_{i_1} - \mathbf{y}_{i_2} \to g^*$ with

$$\mathbf{y}_{i_1}^* - \mathbf{y}_{i_2}^* \sim \exp(-2d\rho)\sum_{q\geq 0}\frac{(2d\rho)^q}{q!}\bar{\mu}_q. \tag{27}$$

Therefore, Given that $\boldsymbol{\eta}_{i_1} - \boldsymbol{\eta}_{i_2} \sim \nu * \nu^-$ and that

$$\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \left(\mathbf{y}_{i_1} - \mathbf{y}_{i_2}\right) + \left(\boldsymbol{\eta}_{i_1} - \boldsymbol{\eta}_{i_2}\right),$$

we convolve (27) with $\nu * \nu^-$ to recover (8). □

Proof. Using Bayes rule we write

$$\mathbb{P}(\mathbf{y}_i = 0|\hat{\mathbf{y}}_i = \omega) = \frac{\mathbb{P}(\hat{\mathbf{y}}_i = \omega \wedge \mathbf{y}_i = 0)}{\mathbb{P}(\hat{\mathbf{y}}_i = \omega)}. \quad (28)$$

From (7) we can deduce that

$$\mathbb{P}(\hat{\mathbf{y}}_i = \omega \wedge \mathbf{y}_i = 0) = \exp(-d\rho)\nu(\omega) \quad (29)$$

and using equation (7) from Theorem III.2 we obtain that as $n \to \infty$,

$$\mathbb{P}(\hat{\mathbf{y}}_i = \omega) \to \exp(-d\rho) \sum_{q \geq 0} \frac{(d\rho)^q}{q!} (\nu * \mu_q)(\omega). \quad (30)$$

Coupling (28), (30) and (29) yields (5).

Again, by Bayes rule,

$$\mathbb{P}\left(\mathbf{y}_{i_1} = \mathbf{y}_{i_2} | \hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega\right) = \frac{\mathbb{P}\left(\mathbf{y}_{i_1} = \mathbf{y}_{i_2} \wedge \hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega\right)}{\mathbb{P}\left(\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega\right)}. \quad (31)$$

Noting that $\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = (\mathbf{y}_{i_1} - \mathbf{y}_{i_2}) + (\boldsymbol{\eta}_{i_1} - \boldsymbol{\eta}_{i_2})$,

$$\mathbb{P}\left(\mathbf{y}_{i_1} = \mathbf{y}_{i_2} \wedge \hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega\right) = \tilde{\nu}(\omega) \quad (32)$$

By (7) from Theorem III.2 we obtain that as $n \to \infty$,

$$\mathbb{P}(\hat{\mathbf{y}}_{i_1} - \hat{\mathbf{y}}_{i_2} = \omega) \to \exp(-2d\rho) \sum_{q \geq 0} \frac{(2d\rho)^q}{q!} \tilde{\nu} * \bar{\mu}_q(\omega) \quad (33)$$

Coupling (31), (32), and (33) yields (6).

□

Proof of Lemma III.3. Let $\mathbf{x} \sim \mu$ and $\mathbf{x}_i \sim \mu_i$ be such that $\mathbb{E}[\mathbf{x}_i] = 0$ and $\mathbb{V}ar[\mathbf{x}_i] = \sigma_i^2$. Note that since $\mathbb{E}[\mathbf{x}] = 0$, $\mathbb{V}ar[\mathbf{x}] = \int \omega^2 \mu(\omega) d\omega = \int \omega^2 \left(\sum_i \alpha_i \mu_i(\omega)\right) d\omega$. Therefore, $\mathbb{V}ar[\mathbf{x}] = \sum_i \alpha_i \int \omega^2 \mu_i(\omega) d\omega = \sum_i \alpha_i \sigma_i^2$. □

REFERENCES

[1] R. Mendoza-Smith and J. Tanner, "Expander l0-decoding," *Applied and Computational Harmonic Analysis*, pp. –, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520317300210

[2] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, Dec 2005.

[3] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[4] D. Donoho, V. Stodden, Y. Tsaig *et al.*, "Sparselab," *SparseLab: Seeking Sparse Solutions to Linear Systems of Equations, SparseLab toolbox shared online, http://sparselab. stanford. edu/, 24th August*, 2007.

[5] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale ℓ_1-regularized least squares," *IEEE journal of selected topics in signal processing*, vol. 1, no. 4, pp. 606–617, 2007.

[6] M. A. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of selected topics in signal processing*, vol. 1, no. 4, pp. 586–597, 2007.

[7] E. Candes and J. Romberg, "l1-magic: Recovery of sparse signals via convex programming," *URL: www. acm. caltech. edu/l1magic/downloads/l1magic. pdf*, vol. 4, p. 14, 2005.

[8] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *CoRR*, vol. abs/0805.0510, 2008.

[9] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, *Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition*, Nov 1993.

[10] 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, 2009.

[11] T. Blumensath and M. E. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE Journal of selected topics in signal processing*, vol. 4, no. 2, pp. 298–309, 2010.

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

[13] J. D. Blanchard, J. Tanner, and K. Wei, "CGIHT: Conjugate gradient iterative hard thresholding for compressed sensing and matrix completion," *Information and Inference*, vol. 4, no. 4, pp. 289–327, 2015.

[14] V. Cevher, "An ALPS view of sparse recovery," in *Acoustics Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, 2011, pp. 5808 –5811.

[15] R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*. IEEE, 2008, pp. 798–805.

[16] I. Haviv and O. Regev, "The restricted isometry property of subsampled fourier matrices," in *Geometric Aspects of Functional Analysis*. Springer, 2017, pp. 163–179.

[17] P. Indyk, "Sketching, streaming and sublinear-space algorithms," *Graduate course notes, available at http://stellar.mit.edu/S/course/6/fa07/6.895/*, 2007.

[18] S. Muthukrishnan *et al.*, "Data streams: Algorithms and applications," *Foundations and Trends® in Theoretical Computer Science*, vol. 1, no. 2, pp. 117–236, 2005.

[19] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Sudocodes — fast measurement and reconstruction of sparse signals," in *2006 IEEE International Symposium on Information Theory*, July 2006, pp. 2804–2808.

[20] W. Xu and B. Hassibi, "Efficient compressive sensing with deterministic guarantees using expander graphs," in *Information Theory Workshop, 2007. ITW '07. IEEE*, Sept 2007, pp. 414–419.

[21] R. Berinde and P. Indyk, "Sequential sparse matching pursuit," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, Sept 2009, pp. 36–43.

[22] R. Berinde, P. Indyk, and M. Ruzic, "Practical near-optimal sparse recovery in the l1 norm," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, Sept 2008, pp. 198–205.

[23] S. Jafarpour, W. Xu, B. Hassibi, and R. Calderbank, "Efficient and robust compressed sensing using optimized expander graphs," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4299–4308, Sept 2009.

[24] Y. Ma, D. Baron, and D. Needell, "Two-part reconstruction with noisy-sudocodes," *IEEE Trans. Signal Processing*, vol. 62, no. 23, pp. 6323–6334, 2014. [Online]. Available: http://dx.doi.org/10.1109/TSP.2014.2362892

[25] B. Bah and J. Tanner, "Vanishingly sparse matrices and expander graphs, with application to compressed sensing," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7491–7508, Nov 2013.

[26] S. Foucart and H. Rauhut, *A mathematical introduction to compressive sensing*. Birkhäuser Basel, 2013, vol. 1, no. 3.

[27] A. Klenke, *Probability theory: a comprehensive course*. Springer Science & Business Media, 2013.

[28] F. Leone, L. Nelson, and R. Nottingham, "The folded normal distribution," *Technometrics*, vol. 3, no. 4, pp. 543–550, 1961.

[29] D. L. Donoho and J. Tanner, "Precise undersampling theorems," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 913–924, 2010.

[30] J. D. Blanchard, J. Tanner, and K. Wei, "Conjugate gradient iterative hard thresholding: Observed noise stability for compressed sensing," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 528–537, Jan 2015.

[31] J. D. Blanchard and J. Tanner, "Performance comparisons of greedy algorithms in compressed sensing," *Numerical Linear Algebra with Applications*, vol. 22, no. 2, pp. 254–282, 2015.

[32] ——, "GPU accelerated greedy algorithms for compressed sensing," *Mathematical Programming Computation*, vol. 5, no. 3, pp. 267–304, 2013.

[33] D. L. Donoho and J. Tanner, "Exponential bounds implying construction of compressed sensing matrices, error-correcting codes and neighborly polytopes by random sampling," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 2002–2016, 2010.