Sudocodes – Fast Measurement and Reconstruction of Sparse Signals

Shriram Sarvotham, Dror Baron, Richard G. Baraniuk
Department of Electrical and Computer Engineering
Rice University, Houston, TX 77005, USA

***Abstract*— Sudocodes are a new scheme for lossless compressive sampling and reconstruction of sparse signals. Consider a sparse signal $x \in \mathbb{R}^N$ containing only $K \ll N$ non-zero values. Sudo-encoding computes the codeword $y \in \mathbb{R}^M$ via the linear matrix-vector multiplication $y = \Phi x$, with $K < M \ll N$. We propose a non-adaptive construction of a sparse Φ comprising only the values 0 and 1; hence the computation of y involves only sums of subsets of the elements of x. An accompanying sudo-decoding strategy efficiently recovers x given y. Sudocodes require only $M = O(K \log(N))$ measurements for exact reconstruction with worst-case computational complexity $O(K \log(K) \log(N))$. Sudocodes can be used as erasure codes for real-valued data and have potential applications in peer-to-peer networks and distributed data storage systems. They are also easily extended to signals that are sparse in arbitrary bases.**

I. Introduction

In this paper, we introduce *sudocodes*, an efficient lossless measurement and reconstruction scheme for compressive sampling (CS) of sparse signals. Consider a discrete-time signal $x \in \mathbb{R}^N$ that contains only K non-zero values. The core tenet of CS is that it is unnecessary to measure all the N values of x; rather, we can recover x from a small number of projections onto an *incoherent* basis [1, 2]. To measure x, we compute the measurement vector $y \in \mathbb{R}^M$ as $M \ll N$ linear projections of x via the matrix-vector multiplication $y = \Phi x$. Candès et al. [1] and Donoho [2] proposed a measurement strategy using independent and identically distributed (iid) Gaussian entries for Φ and a tractable reconstruction procedure based on linear programming. The reconstruction based on linear programming requires $M = cK$ projections for signal reconstruction, where $c \approx \log_2(1 + N/K)$ [3], and has computational complexity at least $O(N^3)$ [2].

While linear programming techniques figure prominently in the design of practical CS reconstruction schemes, their $O(N^3)$ complexity renders them impractical for many applications. We often encounter sparse signals with large N; for example, current digital cameras acquire images with the number of pixels N of the order of 10^6 or more. For such applications, the need for faster reconstruction algorithms is critical. Furthermore, the measurement process also has high complexity; encoding with a full Gaussian or Bernoulli Φ requires $O(MN)$ computations.

In this paper, we develop a framework we term *sudocoding* to measure (or encode) the signal x so that the reconstruction (or decoding) of x given y and Φ has low computational complexity. Sudocodes are applicable to strictly sparse signals whose non-zero values satisfy certain simple conditions and can be extended to signals that are sparse in arbitrary bases.

The key idea behind sudocoding is to construct measurements by summing subsets of the coefficient values $x(i)$ akin to group testing [4]. We consider subsets comprising L coefficients of x at a time. In other words, the entries of Φ are either 0 or 1, with 1 appearing L times in each row of Φ (sudo-decoding is robust to a relatively large range of L). The sparse structure of Φ lends itself to fast measurement and reconstruction algorithms. The sudo-decoder receives a stream of measurements and the corresponding rows of Φ and performs reconstruction.[1]

The name "sudocode" is inspired by Sudoku puzzles, where the challenge is to determine unknown values of cells in a grid [5];[2] our sudo-decoder works in a similar manner. Decoding is based on the key insight that both zero measurements and matching measurements can be used to infer coefficient values. As in group testing [4], if the decoder receives a zero measurement, then it knows that all coefficients that were involved in generating the measurement are zero. Sudocodes extend group testing by also resolving coefficient values when the decoder locates matching measurements. We use a Binary Search Tree (BST) data structure to store the

This work was supported by NSF-CCF, NSF-NeTS, ONR, AFOSR, DARPA, and the Texas Instruments Leadership University Program.
Email: {shri, drorb, richb}@rice.edu; Web: dsp.rice.edu/cs.

[1] Pseudo-random construction of Φ obviates the need to transmit the rows of Φ; the decoder can easily reproduce Φ using the same seed.

[2] Thanks to Ingrid Daubechies for pointing out the connection.

TABLE I

Comparison of Sudocodes with Chaining Pursuit [6].[3]

	Chaining Pursuit	Sudocodes
$N = 10000$ $K = 10$	$M = 5915$ $T = 0.16s$	$M = 461$ $T = 0.14s$
$N = 10000$ $K = 100$	$M = 90013$ $T = 2.43s$	$M = 803$ $T = 0.37s$
$N = 100000$ $K = 10$	$M = 17398$ $T = 1.13s$	$M = 931$ $T = 1.09s$
$N = 100000$ $K = 1000$	$M > 10^6$ $T > 30s$	$M = 5132$ $T = 5.47s$

measurements to perform search operations in logarithmic time. We show that sudocodes require $M = O(K\log(N))$ measurements for perfect reconstruction with high probability and that the worst-case decoding complexity is $O(K\log(K)\log(N))$. Sudocodes thus substantially expand the horizon for CS applications. We present simulation results for problem sizes of practical interest and show that the number of measurements and decoding times outperform other recently proposed schemes (Table I).

One of the highlights of sudocoding is that it employs an *avalanche* strategy to accelerate the decoding process. The inference of a coefficient value triggers an avalanche of coefficient revelations and thus enables the signal to be recovered using far fewer measurements.

Sudocodes are applicable beyond data acquisition systems. They can be applied as a sort of *erasure code* [7] for data in $\mathbb{R}^N$. We can use this property in distributed settings where multiple encoders send measurements to a single reconstruction center. Once enough measurements have been received at the reconstruction center, the original signal can be recovered.

While we present sudocodes for sparse signals in the canonical "spike" basis, they can be generalized to the case where x is sparse in an arbitrary basis Ψ. In this case, we multiply the sudocode matrix Φ with the sparsifying matrix Ψ and use $\Phi\Psi$ to encode x as $y = (\Phi\Psi)x$. The decoder uses y to determine the signal represented in the sparsifying basis (namely Ψx), and x is then recovered by multiplying by Ψ^T. The encoding matrix $\Phi\Psi$ may not be sparse. Therefore, the encoding and decoding must perform extra computations; this extra cost is $O(N^2)$ in general. Fortunately, in many practical situations Ψ is structured (e.g., Fourier or wavelet bases) and thus amenable to fast computation. Therefore, extending sudocodes to such bases is feasible.

[3]The decoding time T is based on simulations in Matlab 7.1.0 running on a Linux platform with a 2.2GHz AMD Athlon 64X2 Dual Core Processor 4400+ and 2GB memory. The value of M shown for sudocodes includes measurements in both decoding phases.

II. Related work

Encoding and decoding of sparse signals in $\mathbb{R}^N$ has been addressed in the compressed sensing (CS) literature. The practical revelation that supports CS is that the much easier problem of ℓ_1 minimization $\widehat{x} = \arg\min \|x\|_1$ s.t. $y = \Phi x$ can be used to reconstruct x [1, 2]. This optimization problem can be solved with linear programming techniques. Yet, the $O(N^3)$ complexity of these techniques is impractical for many applications.

At the expense of slightly more measurements, iterative greedy algorithms have been developed. Examples include the iterative Orthogonal Matching Pursuit (OMP) [8], Matching Pursuit (MP), and Tree Matching Pursuit (TMP) [9] algorithms. In CS applications, OMP requires $c \approx 2\ln(N)$ [8] to succeed with high probability; decoding complexity is $O(NK^2)$. Unfortunately, $O(NK^2)$ is cubic in N and K, and therefore OMP is also impractical for large N.

Cormode and Muthukrishnan recently proposed a fast CS algorithm based on group testing [10]. Their scheme considers subsets of the signal coefficients in which we expect at most one large coefficient to lie. Within this set, they locate the position and value of the large coefficient using a Hamming code construction. Their decoding scheme has low $O(K\log^2(N))$ complexity, but they require $M = O(K\log^2(N))$ measurements.

Gilbert et al. [6] propose the Chaining Pursuit (CP) algorithm for reconstructing K-sparse signals. Unlike sudocodes, CP can be applied to approximately sparse and compressible signals as well.[4] CP reconstructs with $O(K\log^2(N))$ non-adaptive linear measurements in $O(K\log^2(N)\log^2(K))$ time. CP works best for "super-sparse" signals, where the ratio K/N is very small (see Table I). Define the *sparsity rate* as the fraction of non-zero coefficients in x: $S \triangleq K/N$. As we increase S, CP requires an enormous number of measurements; for some ranges of S the number of measurements $M > N$, which is undesirable. In contrast, sudocodes require only $M = O(K\log(N))$ measurements and offer $O(K\log(K)\log(N))$ worst-case computational complexity for a wide range of sparsity rates S.

III. Sudocodes

A. Setup

The sudocode scheme presented in this paper is for strictly sparse signals. Without loss of generality, we make the following mild assumption. Let $\Omega \triangleq \{i : i \in [1,\ldots,N], x_i \neq 0\}$ be the support set for the non-zero entries of x and denote its cardinality $|\Omega|$

[4]A signal is compressible if its coefficients sorted by magnitude obey a power law; it can be approximated well by a sparse signal.

by K. We assume that for any two subsets $\Omega_1, \Omega_2 \subseteq \Omega$ with $\Omega_1 \neq \Omega_2$, the following condition holds: $\left|\sum_{i\in\Omega_1} x_i - \sum_{i\in\Omega_2} x_i\right| > \epsilon$; that is, we assume that the sum of $x(i)$'s corresponding to each subset of Ω is unique up to a given precision ϵ. This assumption is valid with high probability if the non-zero signal coefficients are drawn from a continuous random distribution such as uniform or Gaussian. For signals that do not conform to the above condition, we add pseudo-random perturbations to the non-zero coefficients to make it so. The decoder can then regenerate the pseudo-random perturbations using the same seed and subtract them off after reconstruction. Extensions of sudocodes to other signal classes such as sparse signals in noise and compressible signals are part of our ongoing work (see Section VI).

B. Sudo-encoding

A sudo-encoding matrix Φ is sparse with entries 0 or 1. Each row of Φ contains exactly L ones placed randomly, where L is a constant chosen based on the values of N and K. We show below that L proportional to N/K yields good performance (Section IV). The encoder constructs the measurement vector y via the matrix-vector multiplication $y = \Phi x$, which merely sums different sets of L coefficients of x.

C. Sudo-decoding

The decoding scheme is iterative; we process each measurement $y(j)$ in succession, $j = 1, 2, \ldots, M$. If we have identified some of the values of the L coefficients that make up measurement $y(j)$ from previous iterations, then we account for those values by subtracting them from $y(j)$ and then considering the modified measurement, which we call $y^*(j)$. Effectively, we are decreasing the size of the problem. The modified measurement captures the unexplained portion of the original measurement $y(j)$. Now, $y^*(j)$ is a sum of $L^*(j)$ coefficients of x, where $L^*(j) \leq L$. The decoding process proceeds based on whether $y^*(j)$ is zero or non-zero.

Case 1: $y^*(j) = 0$: We conclude that the corresponding values of the $L^*(j)$ coefficients of x are all zero. This follows from our uniqueness assumption on the non-zero coefficients of x.

Case 2: $y^*(j) \neq 0$: We compare $y^*(j)$ to the past measurements (i.e., $y^*(k)$, $k < j$). If there is a perfect match (within the precision ϵ) between $y^*(j)$ and a past measurement then we infer that the two measurements originate from the same set of non-zero coefficients of x. Therefore the indices present in one set but not in the other set have zero coefficient values. Furthermore, we also determine the coefficient value of a non-zero coefficient if the intersection of the two index sets contains only one element; this occurs with overwhelmingly high probability as we increase N.

Accelerated decoding: In order to search the past measurements for matches efficiently, we store the past measurements in a Binary Search Tree (BST) data structure. For a BST containing n elements, the search, insert, delete, and update operations require $O(\log(n))$ time on average. We also introduce the *history* matrix H of size $R \times N$, where $R = O(ML/N)$. The i'th column of H stores the list of row numbers of Φ that measured $x(i)$ up to the current value of j. The history matrix makes it easier to update the measurements as the coefficients of x are recovered. When $x(i) \neq 0$ is resolved, we subtract off its value from the previous measurements to which it contributed.

Avalanche: An avalanche strategy further accelerates the decoding process and also reduces the number of measurements. Whenever a coefficient is resolved, we search the past measurements that measured the coefficient value (using the history matrix) to induce further coefficient revelations. This triggers an avalanche of coefficient revelations, and thus the signal can be recovered using fewer measurements.

Two phase coding: It is useful to break the encoding and decoding processes into two phases. First consider encoding. Phase 1 encoding proceeds as above by computing sums of sets of L coefficients of x. Phase 2 measurements are encoded differently: we compute $\widetilde{K}$ measurements of x using a *non-sparse* $\widetilde{K} \times N$ matrix $\widetilde{\Phi}$. The only requirement for $\widetilde{\Phi}$ is that every submatrix formed by taking $\widetilde{K}$ columns from $\widetilde{\Phi}$ is invertible (the value of $\widetilde{K}$ will be described soon). Now consider decoding. In Phase 1 decoding, we proceed as above until we have reconstructed most of the coefficients in x. When at least $N - \widetilde{K}$ coefficient values have been determined, we transition to Phase 2 decoding. In Phase 2 decoding, we have at most $\widetilde{K}$ remaining unknown coefficients. We extract the columns of $\widetilde{\Phi}$ corresponding to the unresolved coefficients to form a $\widetilde{K} \times \widetilde{K}$ matrix $\widetilde{\Phi}_{\widetilde{K}}$. The decoder solves for the unknown coefficients by computing the matrix inverse of $\widetilde{\Phi}_{\widetilde{K}}$ and multiplying it with the Phase 2 measurements. Since matrix inversion has cubic complexity, we select $\widetilde{K}$ proportional to $K^{1/3}$, thereby requiring $O(K)$ complexity for Phase 2 decoding.

The reason for introducing Phase 2 is as follows. As we decode more coefficients of x in Phase 1, we see more frequent occurrences of measurements that provide no new information; this occurs when all of the indices corresponding to a row of Φ are already known. Phase 2 en/decoding is introduced before the number of such measurement explodes.

IV. Analysis of Sudo-Decoding

A. Choice of L

The performance of sudo-decoding (the number of measurements M as well as the encoding and decoding time) depends on the choice of L. The optimal L depends on the probability that we capture all zero coefficients in a set of size L. For large N, this probability depends only on the sparsity rate $S = K/N$. Figure 1 shows the number of measurements needed (M) as a function of L. This plot is based on simulations for $N = 16000$, 64000, and 256000 and sparsity rate $S = 0.02$. Figure 2 plots the optimal L (which requires the least number of measurements) for a variety of sparsity rates.

B. Number of measurements M

The following theorem characterizes the trade-off between M and the fraction of coefficients recovered.

Theorem 1: Using $L = O(N/K)$, Phase 1 sudo-decoding requires $M = O(K \log(1/\epsilon))$ measurements for exact reconstruction of $N(1-\epsilon)$ coefficients.

Proof: Let $L = c_1/S = c_1 N/K$. Consider a single measurement. We have $\Pr(\text{zero measurement}) = f_1(c_1) + o(1)$, where the $o(1)$ term vanishes as N increases and the binomial distribution is approximated as Poisson. Similarly, the probability that a single coefficient was measured satisfies $\Pr(\text{single coefficient}) = f_2(c_1) + o(1)$.

Let Phase 1 take $M = c_2 K$ measurements in total. In the remainder of the proof we ignore the $o(1)$ terms, because they are easily accounted for by modifying M slightly. Phase 1 encounters a zero measurement $f_1(c_1)c_2 K$ times; each time $L = c_1 N/K$ zero coefficients are determined, for a total of $c_1 c_2 f_1(c_1) N$ zero coefficients. However, the same zero coefficient may have been determined multiple times. Because there are $N - K$ zero coefficients in total, each was determined $c_1 c_2 f_1(c_1) N/(N-K) > c_1 c_2 f_1(c_1)$ times. Therefore, the probability that any specific zero coefficient was determined is greater than $f_3(c_1 c_2 f_1(c_1))$ where the Poisson approximation gives $f_3(\theta) = 1 - e^{-\theta} + o(1)$.

We now analyze the total number of non-zero coefficients recovered in Phase 1. The total number of times that a single coefficient was measured is $c_2 f_2(c_1) K$. If the same non-zero coefficient is measured multiple times without other coefficients interfering, then that coefficient is recovered by Phase 1. This happens with probability $f_4(c_2 f_2(c_1))$, where the Poisson approximation gives $f_4(\theta) = 1 - e^{-\theta}(1+\theta) + o(1)$. We complete the proof by noting that c_1, $f_1(c_1)$, and $f_2(c_1)$ are constants, and so the proportion of coefficients that are not recovered decays as $e^{-O(c_2)}$. □

Phase 1 must cover all but the last $O(K^{1/3})$ coefficients, which corresponds to $\epsilon = O(K^{1/3}/N)$. Applying Theorem 1, we have $M = O(K \log(N/K^{1/3})) = O(K \log(N))$. We express this in the following.

Corollary 1: Using $L = O(N/K)$, Phase 1 sudo-decoding requires $M = O(K \log(N))$ measurements for exact reconstruction of $N - \widetilde{K}$ coefficients, where $\widetilde{K} = O(K^{1/3})$.

Finally, Phase 2 requires $O(K^{1/3})$ measurements, which is much less than the $O(K \log(N))$ measurements required for Phase 1. Therefore, the total number of measurements required for exact reconstruction by sudo-decoding is $M = O(K \log(N))$.

C. Computational complexity

Encoding complexity: Each measurement requires $O(L)$ additions. Therefore, sudo-encoding requires $O(ML) = O(N \log(N))$ computations in Phase 1. In Phase 2, sudo-encoding requires $O(\widetilde{K} N)$ complexity in general; however special measurements such as DCT or wavelets can be used to reduce the complexity to $O(N \log(N))$ or $O(N)$.

Decoding complexity: Sudo-decoding can be implemented efficiently using the following data structures. The measurement matrix Φ must be stored sparsely. By storing the indices of the L coefficients measured by each row, we can efficiently perform operations such as comparing the support sets of two rows when two non-zero measurements are found to be equal. Using the BST data structure to store past measurements, we can search for matching measurements with $O(\log(M))$ complexity. Using the history matrix H makes it easier to update the measurements as the coefficients of x are recovered, as described in Section III-C. Every search for past identical measurements using the BST costs $O(\log(M))$. Because the aggregate number of accesses to the heap is $O(M)$, the aggregate decoding complexity is $O(M \log(M)) = O(K \log(N) \log(K \log(N))) = O(K \log(K) \log(N))$.[5]

Phase 2 inverts a matrix of size $\widetilde{K} \times \widetilde{K}$, and so its complexity is $O(\widetilde{K}^{1/3}) = O(K)$, which is smaller than the computation in Phase 1. We summarize the discussion with the following theorem.

Theorem 2: The computation complexity of the sudo-encoder is $O(N \log(N))$. The computation complexity of the sudo-decoder is $O(K \log(K) \log(N))$.

V. Numerical results

The decoding performance of sudocodes (number of measurements M and decoding time T) is presented

[5]We assume that K grows faster than $\log(N)$. Otherwise, the decoding complexity is $O(K \log(N) \log\log(N))$.

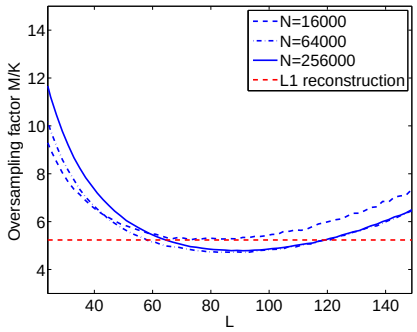

Fig. 1. Over sampling factor vs. L for sparsity rate $S = 0.02$. There is an optimal choice of L that minimizes M.

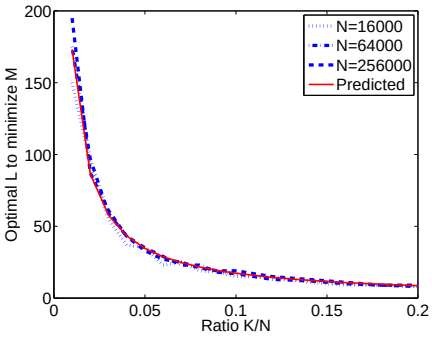

Fig. 2. Optimal of L for different sparsity rates. The optimal L is independent of N and is $O(1/S)$, where $S = K/N$.

in Table I for a range of N and K. For comparison, the corresponding performance of Chaining Pursuit (CP) [6] is also shown. Sudocodes perform very well in all sparsity rate regimes shown in the Table. CP is well suited for "super-sparse" (very small $S = K/N$) signals. For moderately sparse signals, CP requires a very large number of measurements; in particular, $M > N$ in some simulations.

VI. Discussion

Sudocodes could lend themselves to a plethora of applications. Since sudocodes resolve several signal coefficients with each measurement they can be used in applications where partial reconstruction of a signal is sufficient (e.g., in detection). Sudocodes can be used as *erasure codes* for signals in p2p and distributed file storage applications. For example, multiple p2p sources can stream compressed digital content such as video, audio, images, etc. When an individual user wishes to download the content, the sources take sudocode measurements and transmit them toward the user. Once the user has received enough measurements, the original signal can be reconstructed. The thresholded DCT/wavelet coefficients used in JPEG/JPEG2000 compression are excellent candidates for sudocoding.

Sudocodes can be enhanced in a variety of ways. Sudocodes can be made to leverage the *statistical dependencies* between the non-zero coefficients in the signal. For example, natural images that are sparsely represented in a wavelet basis exhibit strong correlation between the parent and the child nodes [9]. Sudocodes can be enhanced to provide resiliency against *unknown sparsity* order K. If the encoder does not have have the knowledge of K, then it can begin with a small estimate for K and then refine the estimate as it transmits measurements. The number of ones L in each row can be set based on the best estimate of K. *Feedback* from the decoder to the encoder can allow the measurements to be adapted to the coefficients that remain unresolved at the decoder.

Finally, modifications can be made to make sudocodes robust to the presence of noise in the signal or the measurements. We are investigating the use of message passing algorithms like belief propagation [11] in this scenario.

Acknowledgments: Thanks to Ron DeVore, Marco Duarte, Michael Wakin, Mark Davenport, Jason Laska, and Matthew Moravec for informative and inspiring conversations.

References

[1] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[2] D. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[3] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed compressed sensing," Nov. 2005, Submitted.

[4] D-Z Du and F. K. Hwang, *Combinatorial Group Testing and its Applications*, Vol 3, Series on Applied Mathematics, World Scientific, 1993.

[5] "Sudoku website," http://www.sudoku.com.

[6] A. C. Gilbert, M. J. Strauss, J. Tropp, and R. Vershynin, "Algorithmic linear dimension reduction in the ℓ_1 norm for sparse vectors," Apr. 2006, Submitted.

[7] J. Bloemer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, "An XOR based erasure resilient coding scheme," *ICSI TR-95-0498*, August 1995, Technical Report.

[8] J. Tropp and A. C. Gilbert, "Signal recovery from partial information via orthogonal matching pursuit," Apr. 2005, Preprint.

[9] M. F. Duarte, M. B. Wakin, and R. G. Baraniuk, "Fast reconstruction of piecewise smooth signals from random projections," in *Proc. SPARS05*, Rennes, France, Nov. 2005.

[10] G. Cormode and S. Muthukrishnan, "Combinatorial algorithms for compressed sensing," *DIMACS Technical Report TR 2005-40*, 2005.

[11] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding Belief Propagation and its Generalizations," *Mitsubishi Tech. Rep. TR2001-022*, Jan. 2002.