Streaming and Compressed Sensing

Piotr Indyk

MIT

Recap

- Algorithms for estimating coordinates in an n-dimensional vector x
  (from a linear sketch Ax of length m)

- In particular, one algorithm guaranteed w.h.p for all $i=1\ldots n$

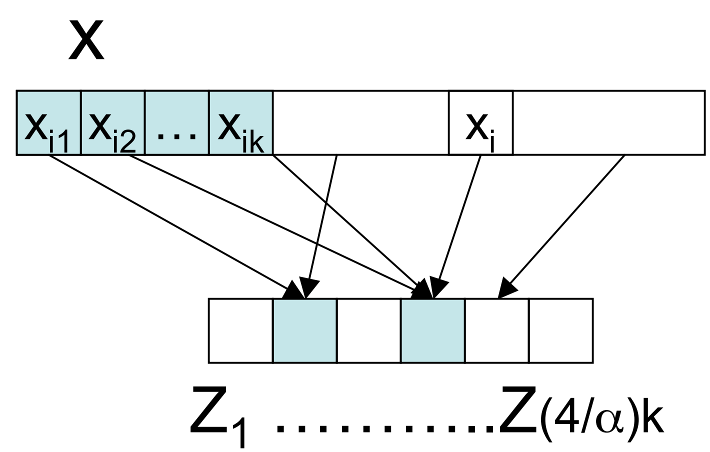
  $$|x^*_i - x_i| < \alpha Err^k_1 / k$$

  using $m=O(k/\alpha \log n)$ sketch length
- In other words we get l_∞ / l_1 guarantee

  $$||x^* - x||_\infty < \alpha Err^k_1 / k$$

- This implies l_1 / l_1 guarantee

  $$||x^* - x||_1 < \alpha Err^k_1$$
- Recovery time: $O(n \log n)$
  - Can improve to $O(k \log^2 n)$ with extra $\log n$ factor in sketch length



Compressive Sensing
[Donoho, Candes-Romberg-Tao,…]

- Concept from the land of engineers
- New ideas:
  - Sensing framework
  - Deterministic matrices A (“for all” signals x, as opposed to “for each”).
    Suffices if A satisfies Restricted Isometry Property (RIP):
    for all k-sparse vectors x

    $$||x||_2 \leq ||Ax||_2 \leq C\ ||x||_2$$

    - Random Gaussian/Bernoulli: $m=O(k \log (n/k))$
    - Random Fourier: $m=O(k \log^{O(1)} n)$

  - L1 minimization, a.k.a. Basis Pursuit

    minimize $||x^*||_1$
    subject to $Ax^*=Ax$

  - L2/L1 guarantee

    $$||x^*-x||_2 < c\ Err^k_1 / k^{1/2}$$

  - Noisy measurements (?!), universality, $O(k \log(n/k))$ sketch length,..

Parameters

- Given: dimension n, sparsity k
- Parameters:
  - Sketch length m
  - Time to compute/update Ax
  - Time to recover x* from Ax
  - Matrix type:
    - Deterministic (one A that works for all x)
    - Randomized (random A that works for a fixed x w.h.p.)
  - Measurement noise, universality, …

Result Table

Paper	Rand. / Det.	Sketch length	Encode time	Sparsity/ Update time	Recovery time	Apprx
[CCF'02], [CM'06]	R	k log n	n log n	log n	n log n	l2 / l2
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l2 / l2
[CM'04]	R	k log n	n log n	log n	n log n	l1 / l1
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1
[CRT'04] [RV'05]	D	k log(n/k)	nk log(n/k)	k log(n/k)	n^c	l2 / l1
	D	k $\log^c$ n	n log n	k $\log^c$ n	n^c	l2 / l1
[GSTV'06] [GSTV'07]	D	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1
	D	k $\log^c$ n	n $\log^c$ n	k $\log^c$ n	k^2 $\log^c$ n	l2 / l1
[BGIKS'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n^c	l1 / l1
[GLR'08]	D	k $\log n^{\log\log\log n}$	kn^{1-a}	n^{1-a}	n^c	l2 / l1
[NV'07], [DM'08], [NT'08,BM'08]	D	k log(n/k)	nk log(n/k)	k log(n/k)	nk log(n/k) * T	l2 / l1
	D	k $\log^c$ n	n log n	k $\log^c$ n	n log n * T	l2 / l1
[IR'08, BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k)	l1 / l1
[BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k) *T	l1 / l1

Legend:

- n=dimension of x
- m=dimension of Ax
- k=sparsity of x*
- T = #iterations

Approx guarantee:

- l2/l2: $||x-x^*||_2 \leq C||x-x'||_2$
- l1/l1: $||x-x^*||_1 \leq C||x-x'||_1$
- l2/l1: $||x-x^*||_2 \leq C||x-x'||_1/k^{1/2}$

Scale: Excellent | Very Good | Good | Fair

Result Table

Paper	Rand. / Det.	Sketch length	Encode time	Sparsity/ Update time	Recovery time	Apprx
[CCF'02], [CM'06]	R	k log n	n log n	log n	n log n	l2 / l2
	R	$k \log^c n$	$n \log^c n$	$\log^c n$	$k \log^c n$	l2 / l2
[CM'04]	R	k log n	n log n	log n	n log n	l1 / l1
	R	$k \log^c n$	$n \log^c n$	$\log^c n$	$k \log^c n$	l1 / l1
[CRT'04] [RV'05]	D	k log(n/k)	nk log(n/k)	k log(n/k)	n^c	l2 / l1
	D	$k \log^c n$	n log n	$k \log^c n$	n^c	l2 / l1
[GSTV'06] [GSTV'07]	D	$k \log^c n$	$n \log^c n$	$\log^c n$	$k \log^c n$	l1 / l1
	D	$k \log^c n$	$n \log^c n$	$k \log^c n$	$k^2 \log^c n$	l2 / l1
[BGIKS'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n^c	l1 / l1
[GLR'08]	D	$k \log n^{\log\log\log n}$	kn^{1-a}	n^{1-a}	n^c	l2 / l1
[NV'07], [DM'08], [NT'08,BM'08]	D	k log(n/k)	nk log(n/k)	k log(n/k)	nk log(n/k) * T	l2 / l1
	D	$k \log^c n$	n log n	$k \log^c n$	n log n * T	l2 / l1
[IR'08, BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k)	l1 / l1
[BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k) *T	l1 / l1
[CDD'07]	D	Ω(n)				l2 / l2

Legend:

- n=dimension of x
- m=dimension of Ax
- k=sparsity of x*
- T = #iterations

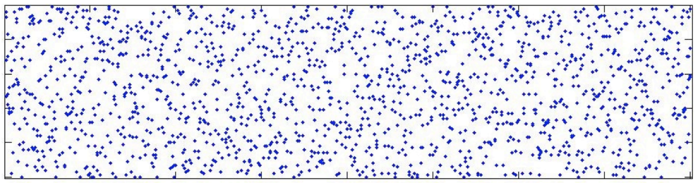
Approx guarantee:

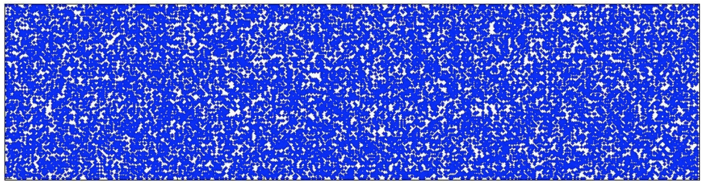
- l2/l2: $\|x-x^*\|_2 \le C\|x-x'\|_2$
- l1/l1: $\|x-x^*\|_1 \le C\|x-x'\|_1$
- l2/l1: $\|x-x^*\|_2 \le C\|x-x'\|_1/k^{1/2}$

Caveats: (1) all bounds up to O() factors; (2) only results for general vectors x are shown; (3) most "dominated" algorithms not shown; (4) specific matrix type often matters (Fourier, sparse, etc); (5) Ignore universality, explicitness, etc

General approach

- Choose encoding matrix A at random
  - Sparse matrices:
    - Data stream algorithms
    - Coding theory (LDPCs)
  - Dense matrices:
    - Compressed sensing
    - Complexity theory (Fourier)





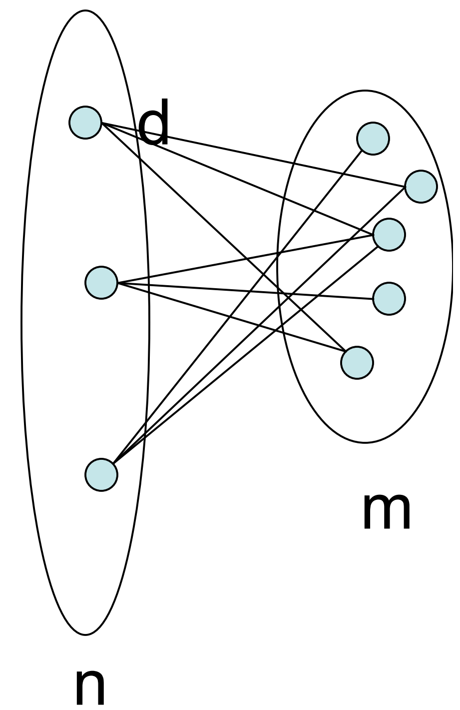
- Tradeoffs:
  - Sparse: computationally more efficient, explicit
  - Dense: shorter sketches
- Best of both worlds ?

Dealing with Sparsity

- Consider "random" m x n adjacency matrices of d-regular bipartite graphs
- Do they satisfy RIP ?
  - No, unless $m=\Omega(k^2)$ [Chandar'07]
- However, they can satisfy the following RIP-1 property: for any k-sparse x

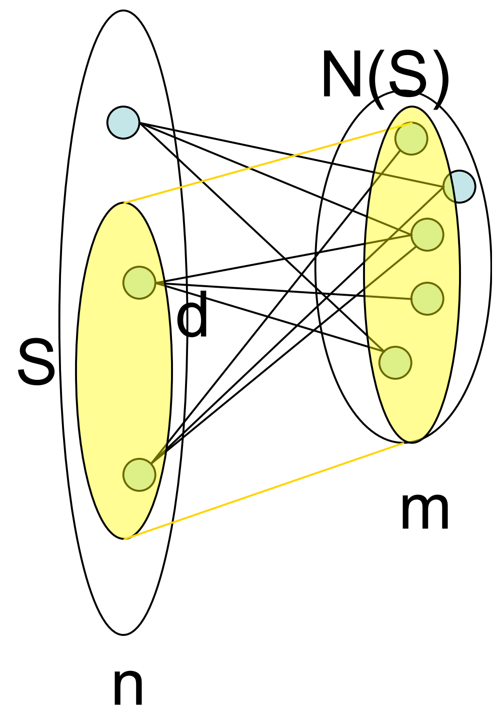
$$d\,(1-2\varepsilon)\ ||x||_1 \le ||Ax||_1 \le d||x||_1$$

  - Sufficient (and necessary) condition: the graph is a $(\ k,\ d(1-\varepsilon)\ )$-expander [Berinde-Gilbert-Indyk-Karloff-Strauss'08]



Expanders

- A bipartite graph is a $(k,d(1-\varepsilon))$-expander if for any left set S, $|S|\leq k$, we have $|N(S)|\geq(1-\varepsilon)d\,|S|$
- Plenty of applications in computer science, coding theory (LDPC) etc
- Constructions:
  - Randomized: $m=O(k \log (n/k))$
  - Explicit: $m=k$ quasipolylog n



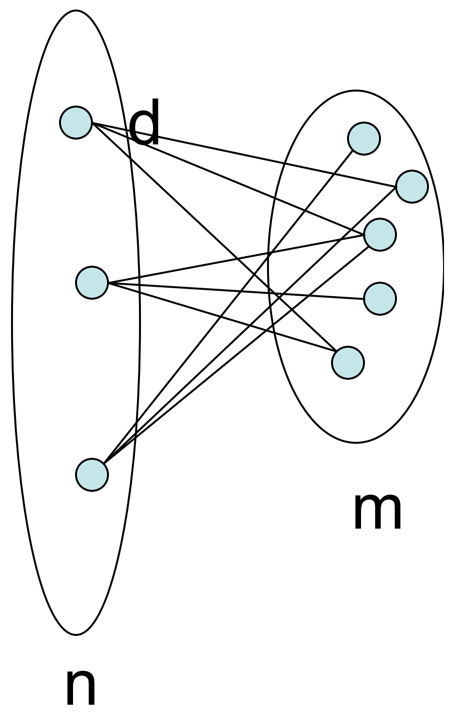
Dealing with Sparsity

- Consider "random" m x n adjacency matrices of d-regular bipartite graphs
- Do they satisfy RIP ?
  - No, unless $m=\Omega(k^2)$ [Chandar'07]
- However, they can satisfy the following RIP-1 property: for any k-sparse x

$$d\,(1-2\varepsilon)\,||x||_1 \le ||Ax||_1 \le d||x||_1$$

  - Sufficient (and necessary) condition: the graph is a $(\, k,\, d(1-\varepsilon)\,)$-expander [Berinde-Gilbert-Indyk-Karloff-Strauss'08]

  (proof of sufficiency in a moment)
- What is the use of RIP-1 ?



A satisfies RIP-1 $\Rightarrow$ LP works

[Berinde-Gilbert-Indyk-Karloff-Strauss'08]

- Compute a vector x^* such that $Ax=Ax^*$ and $||x^*||_1$ minimal
- Then we have, over all k-sparse x'

$$||x-x^*||_1 \leq C \min_{x'} ||x-x'||_1$$

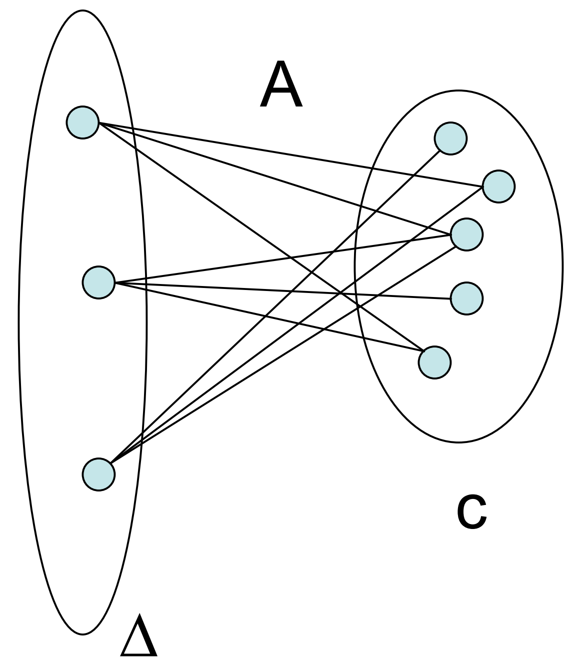
  – $C \rightarrow 2$ as the expansion parameter $\varepsilon \rightarrow 0$
- Can be extended to the case when Ax is noisy

A satisfies RIP-1 $\Rightarrow$ Sparse Matching Pursuit works

[Berinde-Indyk-Ruzic'08]

- Algorithm:
  - $x^*=0$
  - Repeat T times
    - Compute $c=Ax-Ax^* = A(x-x^*)$
    - Compute Δ such that Δ_i is the median of its neighbors in c
    - Sparsify Δ
      (set all but 2k largest entries of Δ to 0)
    - $x^*=x^*+\Delta$
    - Sparsify x^*
      (set all but k largest entries of x^* to 0)
- After T=log() steps we have

$$||x-x^*||_1 \le c\ Err^k_1$$



Proof: d(1-ε)-expansion ⇒ RIP-1

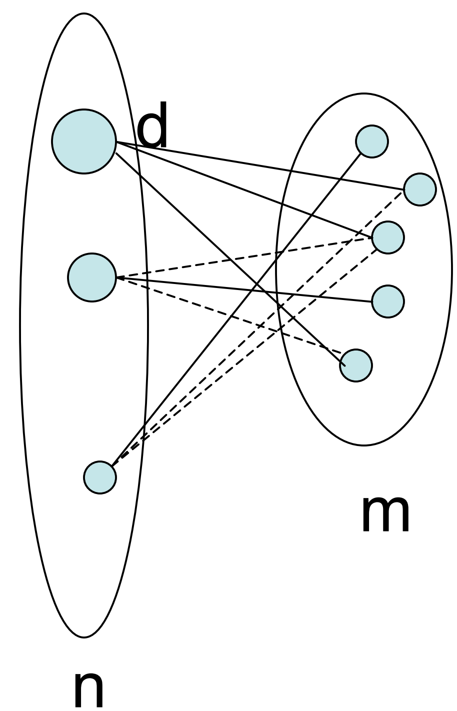
- Want to show that for any k-sparse x we have

  $$d\,(1-2\varepsilon)\,||x||_1 \leq ||Ax||_1 \leq d||x||_1$$

- RHS inequality holds for ***any*** x
- LHS inequality:
  - W.l.o.g. assume

    $$|x_1| \geq \ldots \geq |x_k| \geq |x_{k+1}| = \ldots = |x_n| = 0$$

  - Consider the edges e=(i,j) in a lexicographic order
  - For each edge e=(i,j) define r(e) s.t.
    - r(e)=-1 if there exists an edge (i',j)<(i,j)
    - r(e)=1 if there is no such edge
- Claim: $||Ax||_1 \geq \sum_{e=(i,j)} |x_i| r_e$



Proof: $d(1-\varepsilon)$-expansion $\Rightarrow$ RIP-1 (ctd)

- Need to lower-bound

  $$\sum_e z_e r_e$$

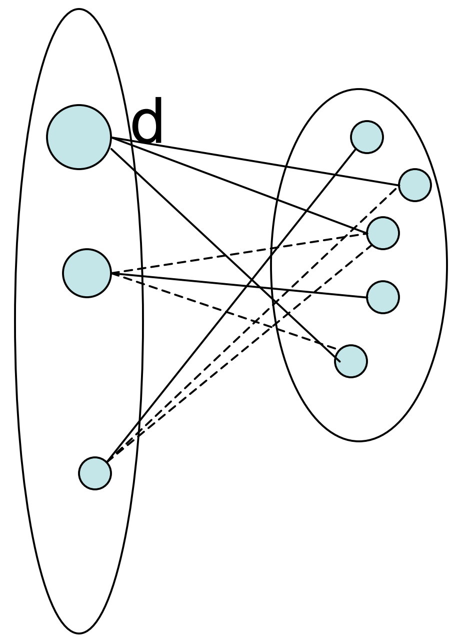
  where $z_{(i,j)}=|x_i|$
- Let R_b= the sequence of the first bd r_e's
- From graph expansion, R_b contains at most εbd -1's

  (for $b=1$, it contains no -1's)
- The sequence of r_e's that minimizes $\sum_e z_e r_e$ is

  $$\underbrace{1,1,\ldots,1}_{d},\underbrace{-1,\ldots,-1}_{\varepsilon d},\underbrace{1,\ldots,1}_{(1-\varepsilon)d},\ldots$$

- Thus

  $$\sum_e z_e r_e \geq (1-2\varepsilon)\sum_e z_e = (1-2\varepsilon)\, d\|x\|_1$$



Scale: Excellent | Very Good | Good | Fair

Result Table

Paper	Rand. / Det.	Sketch length	Encode time	Sparsity/ Update time	Recovery time	Apprx
[CCF'02], [CM'06]	R	k log n	n log n	log n	n log n	l2 / l2
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l2 / l2
[CM'04]	R	k log n	n log n	log n	n log n	l1 / l1
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1
[CRT'04] [RV'05]	D	k log(n/k)	nk log(n/k)	k log(n/k)	n^c	l2 / l1
	D	k $\log^c$ n	n log n	k $\log^c$ n	n^c	l2 / l1
[GSTV'06] [GSTV'07]	D	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1
	D	k $\log^c$ n	n $\log^c$ n	k $\log^c$ n	$k^2 \log^c$ n	l2 / l1
[BGIKS'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n^c	l1 / l1
[GLR'08]	D	k $\log n^{\log\log\log n}$	kn^{1-a}	n^{1-a}	n^c	l2 / l1
[NV'07], [DM'08], [NT'08,BM'08]	D	k log(n/k)	nk log(n/k)	k log(n/k)	nk log(n/k) * T	l2 / l1
	D	k $\log^c$ n	n log n	k $\log^c$ n	n log n * T	l2 / l1
[IR'08, BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k)	l1 / l1
[BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k) *T	l1 / l1
[CDD'07]	D	Ω(n)				l2 / l2

Legend:

- n=dimension of x
- m=dimension of Ax
- k=sparsity of x*
- T = #iterations

Approx guarantee:

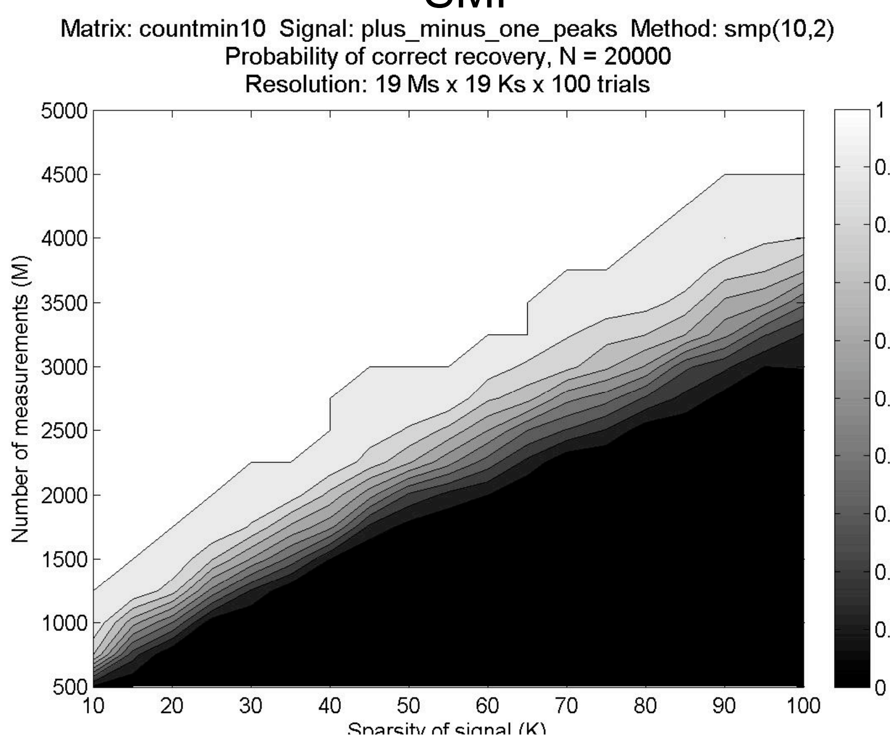
- l2/l2: $||x-x^*||_2 \le C||x-x'||_2$
- l1/l1: $||x-x^*||_1 \le C||x-x'||_1$
- l2/l1: $||x-x^*||_2 \le C||x-x'||_1/k^{1/2}$

Caveats: (1) all bounds up to O() factors; (2) only results for general vectors x are shown; (3) most "dominated" algorithms not shown; (4) specific matrix type often matters (Fourier, sparse, etc); (5) Ignore universality, explicitness, etc

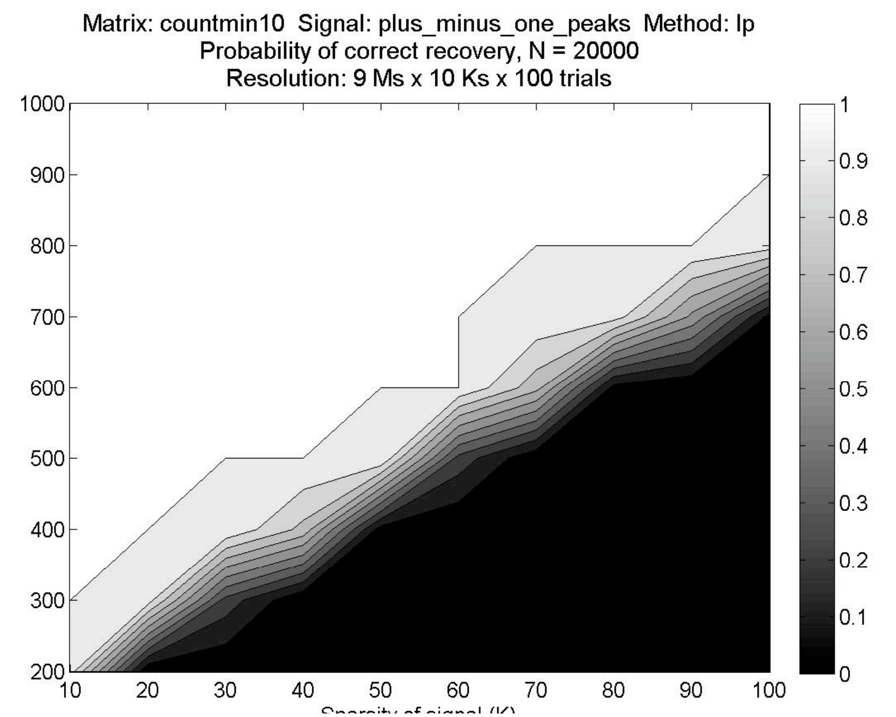
Experiments

- Probability of recovery of random k-sparse +1/-1 signals from m measurements
  - –Sparse matrices with d=10 1s per column
  - –Signal length n=20,000

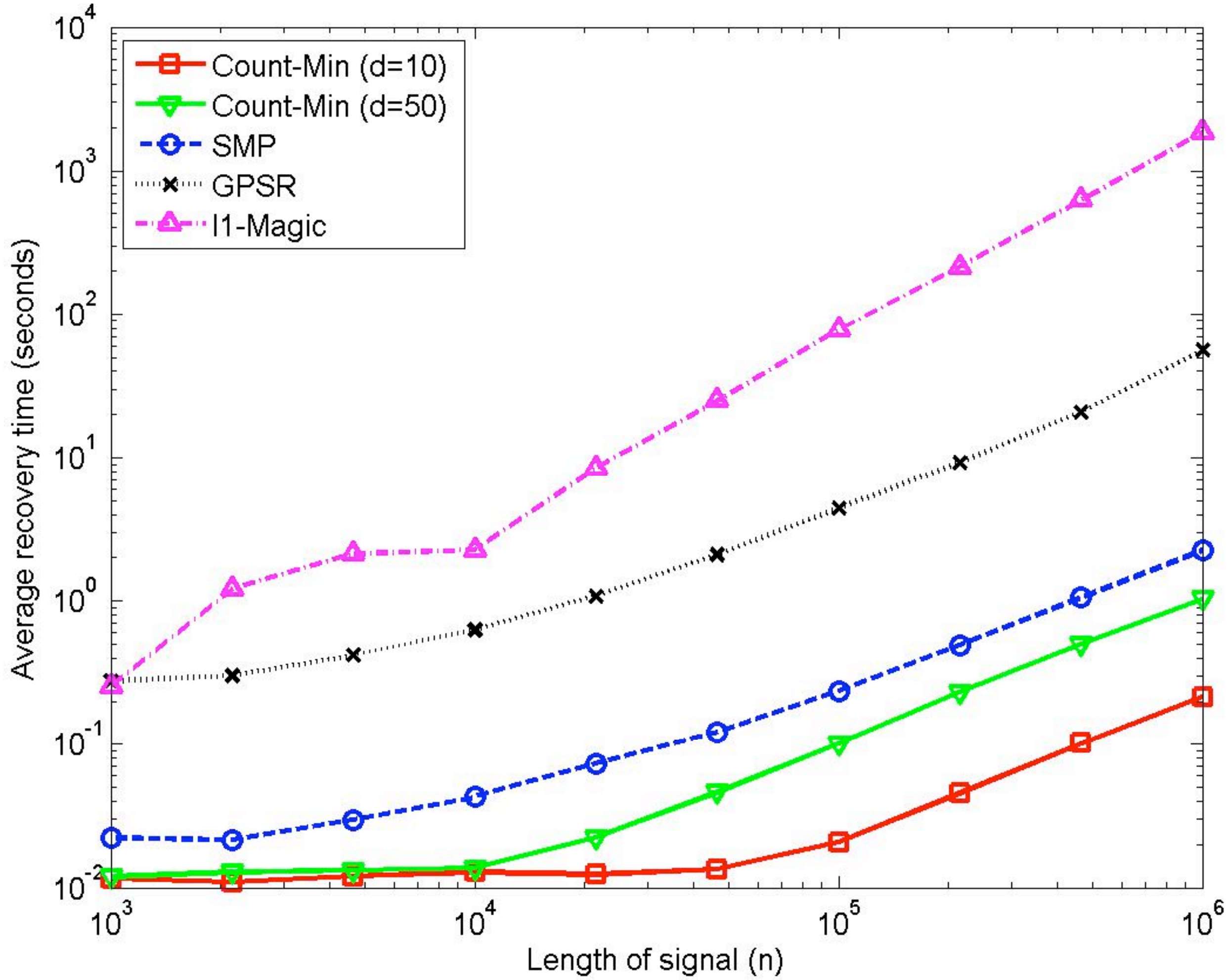
SMP



LP



Running times



Conclusions

- Sparse approximation using sparse matrices
- State of the art: can do 2 out of 3:
  - Near-linear encoding/decoding
  - O(k log (n/k)) measurements
  - Approximation guarantee with respect to L2/L1 norm
- Open problems:
  - 3 out of 3 ?
  - Explicit constructions ?
    - RIP1: via expanders, quasipolylog m extra factor
    - l2 section of l1: quasipolylog m extra factor [GLR]
    - RIP2: extra factor of k [DeVore]

Recovery algorithms

- L1 minimization, a.k.a. Basis Pursuit [Donoho],[Candes-Romberg-Tao]:

  minimize $||x^*||_1$
  subject to $Ax^*=Ax$

  – Solvable in polynomial time using using linear programming

- Matching pursuit: OMP, ROMP, StOMP, CoSaMP, EMP, SMP,…
  – Basic outline:
    - Start from $x^*=0$
    - In each iteration
      – Compute an approximation Δ to $x-x^*$ from $A(x-x^*)=Ax-Ax^*$
      – Sparsify Δ , i.e., set all but t largest (in magnitude) coordinates to 0 (t = parameter)
      – $x^*=x^*+\Delta$
  – Many variations

Result Table (with techniques)

Paper	Rand. / Det.	Sketch length	Encode time	Sparsity	Recovery time	Apprx	Matrix property	Algo
[CCF'02], [CM'06]	R	k log n	n log n	log n	n log n	l2 / l2	sparse +1/-1	"one shot MP" *
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l2 / l2		
[CM'04]	R	k log n	n log n	log n	n log n	l1 / l1	sparse binary	"one shot MP" *
	R	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1		
[CRT'04] [RV'05]	D	k log(n/k)	nk log(n/k)	k log(n/k)	n^c	l2 / l1	RIP2	BP
	D	k $\log^c$ n	n log n	k $\log^c$ n	n^c	l2 / l1		
[GSTV'06] [GSTV'07]	D	k $\log^c$ n	n $\log^c$ n	$\log^c$ n	k $\log^c$ n	l1 / l1	augmented RIP1/RIP2*	MP
	D	k $\log^c$ n	n $\log^c$ n	k $\log^c$ n	k^2 $\log^c$ n	l2 / l1		
[BGIKS'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n^c	l1 / l1	RIP1	BP
[GLR'08]	D	k $\log n^{\log\log\log n}$	kn^{1-a}	n^{1-a}	n^c	l2 / l1	l2 sections of l1	BP
[NV'07], [DM'08], [NT'08,BM'08]	D	k log(n/k)	nk log(n/k)	k log(n/k)	nk log(n/k) * T	l2 / l1	RIP2	MP
	D	k $\log^c$ n	n log n	k $\log^c$ n	n log n * T	l2 / l1		
[IR'08, BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k)	l1 / l1	RIP1/ expansion	MP
[BIR'08]	D	k log(n/k)	n log(n/k)	log(n/k)	n log(n/k) *T	l1 / l1		

l2/l2: $\|x-x^*\|_2 \leq C\|x-x^*\|_2$ l1/l1: $\|x-x^*\|_1 \leq C\|x-x^*\|_1$ l2/l1: $\|x-x^*\|_2 \leq C\|x-x^*\|_1/k^{1/2}$

* In retrospective