Sparse Recovery and Compressive Sensing

CS 264 Guest Lecture
2-21-2017
Mary Wootters

Agenda

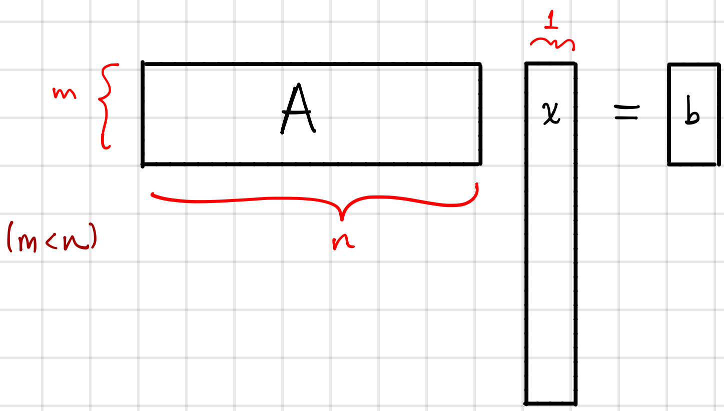
1. Solving Underdetermined Linear Systems
2. What should we hope for?
3. A solution via Linear Programming: Intuition
4. Formal (ish) Proof

① Solving underdetermined linear systems

Recently (I gather) in this course you've been talking about exact solutions: when can you find exact solutions to NP-hard problems?

Today, we'll see another example of this: compressed sensing.

Consider a linear system $Ax=b$ that looks like this:



That is, the system is underdetermined.

PROBLEM: Given A, b, find x.

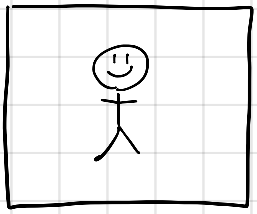
This problem is either impossible or not very interesting, depending on how you ask it:
- It's impossible to find "the" solution, as there may be many solutions.
- It's easy to find "a" solution (some x so that $Ax=b$).

However, this problem becomes interesting if we assume x is SPARSE:

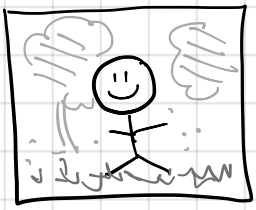
Def A vector $x \in \mathbb{R}^n$ is k-sparse if it has at most k nonzero entries:

$$|\underbrace{\mathrm{supp}(x)}_{\text{support of } x}| \leq k$$

The assumption that x is sparse (or nearly sparse) shows up a lot.



This picture is sparse



This picture is approximately sparse

This picture is also approximately sparse... in a wavelet basis

My voice is approximately sparse (in the frequency domain).

also in the time domain when I'm not talking...

These two relaxations (approximate sparsity and sparsity after a change of basis) make "sparsity" a very natural assumption. For simplicity, for this lecture we will consider only exact sparsity, in the standard basis. But one can extend our discussion to these more general notions.

So, now we have:

PROBLEM: Given A, and $Ax = b$ for some k-sparse x, find x.

There are two ways to interpret this problem, both interesting.

"SPARSE RECOVERY" or "COMPRESSED SENSING"

PROBLEM 1: Let $x \in \mathbb{R}^n$ be k-sparse. Given A and $b = Ax$, find $\hat{x}$ so that $\hat{x} = x$.

If x is approximately sparse, we'd ask for $\hat{x} \approx x$

Here, we should find the original x. In particular, it should be unique.

We will focus on this one in this lecture.

"SPARSE APPROXIMATION"

PROBLEM 2. Given A and b, find any k-sparse x so that $Ax = b$.

Here, we just want to find any sparse solution (assuming it exists).

Sparse approximation is clearly the easier of the two and already this is NP-hard.

Thm SPARSE-APPROXIMATION is NP-hard.

[Natarajan 1995, Davis 1997]

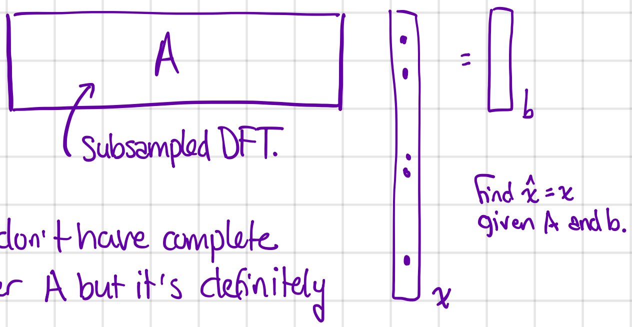
The proof is by reduction from EXACT-COVER-BY-3SETS.

So what hope do we have? The inputs need not be worst case:

In many applications, we get to design the matrix A.

EXAMPLE APPLICATIONS:

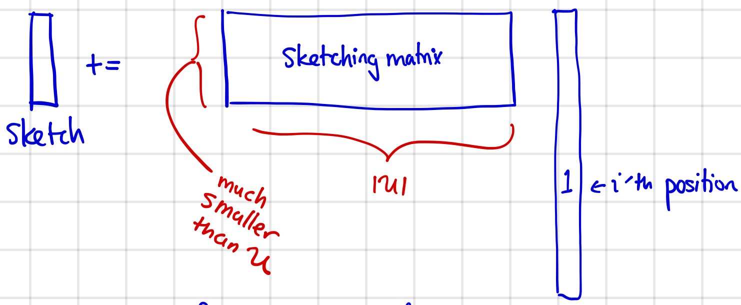
- Sparse recovery for image compression.
  I want to compress data by writing it as a sparse linear combination of a fixed "dictionary" of vectors (these are the columns of A) - I get to choose the dictionary
- "Single pixel camera"
  Instead of acquiring images pixel-by-pixel in the standard basis, instead acquire an image in a compressed form by measuring linear combinations of pixels. Here, we get to choose the linear combinations, aka the matrix A.
- MRI
  Measurements (in a stylized version) are of the form $\langle \varphi, x \rangle$, where x is a sparse vector and φ is a row of the Discrete Fourier Transform.
  So the problem is:



Here, we don't have complete control over A but it's definitely not worst-case.

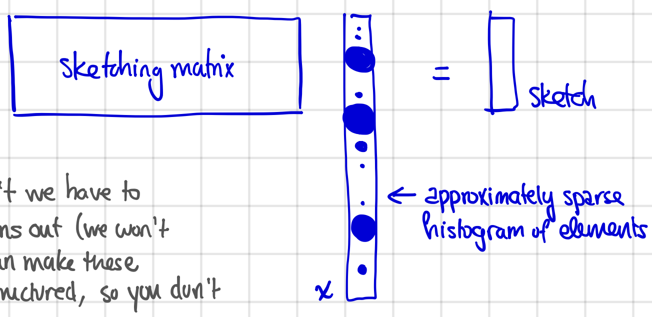
In this example, more samples means more time in the MRI, which is expensive and sometimes medically impossible. (If someone's heart needs to be stopped). So we want to minimize the number of rows.

- Streaming algorithms:
  - Have a data stream of elements in some universe $\mathcal{U}$. and we'd like to estimate the most frequent elements, in very small space.
  - Keep a small "sketch" of the data: when item i arrives, update the sketch by:



In this example, the number of rows of the matrix is the storage* required. So we want to minimize the number of rows.

  - At the end of the day, we have



and we want to identify the support of x.
Here, we have control over the sketching matrix.

*A good question is: but don't we have to store the matrix too? It turns out (we won't talk about it today), you can make these matrices very explicit + structured, so you don't need to store much.

In these applications:

1. We have full or partial control of A, and
2. We would like A to have as few rows as possible.

This leads us to the following questions:

QUESTIONS

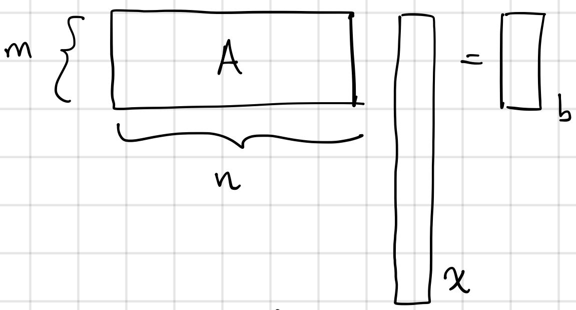
1. For what matrices A is there at most one sparse solution to $Ax=b$?
2. For what matrices A can we find a sparse solution to $Ax=b$ efficiently?
3. And, how can we minimize the number of rows of A in both of the above?

We will tackle all of these questions at the same time, by giving a condition on A so that

1. There is a unique solution
2. We can find it efficiently
3. There exist matrices A with this condition and with very few rows.

(2) WHAT CAN WE HOPE FOR?

Suppose we want there to be a unique k-sparse solution to



How many rows m must A have?

On the back of an envelope, there are $\binom{n}{k}$ possible supports for x, so we need at least $\log\binom{n}{k} \approx k\log(n/k)$ bits in order to distinguish these different possibilities.

So intuitively, we might expect to need $m \gtrsim k\log(n/k)$ rows.

This argument doesn't really make sense, since the elements of b are real numbers, not bits, and in fact for exact sparsity you can actually get away with $m = 2k$.

However, this line of reasoning is pretty truthy and one can show (see, eg, [DoBe, Indyk, Price, Woodruff]) that for approximate sparsity $m = \Omega(k\log(n/k))$ measurements are needed, and this is what we'll shoot for today.

③ A SOLUTION via LINEAR PROGRAMMING. (INTUITION)

(3A) What should we try?

We'd like to solve:

Find the sparsest vector x so that $Ax=b$

aka

$$\boxed{\text{minimize} \quad \underbrace{\|x\|_0}_{} \quad \text{s.t.} \quad Ax=b.}\ ,$$

this is the "ℓ_0 norm", which is the sparsity of x. (It's not really a norm).

but in general this is NP hard. Instead, we consider an LP relaxation instead:

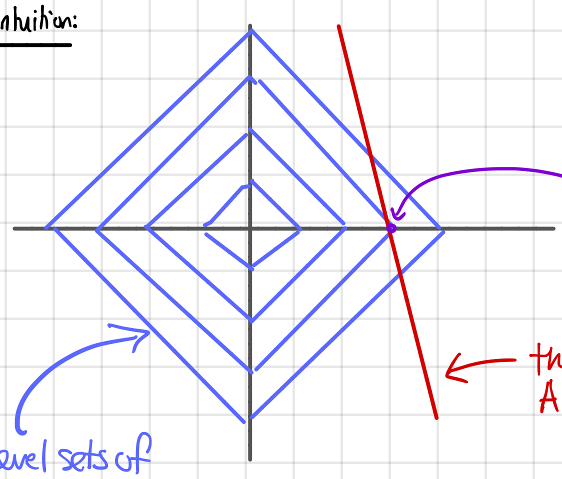
(LP) $$\boxed{\text{minimize} \quad \|x\|_1 \quad \text{s.t.} \quad Ax = b.}$$

$\|x\|_1 = \sum_{i=1}^n |x_i|$ is the ℓ_1 norm.

This is a linear program:
- the variables are $x_1,\ldots,x_n, y_1,\ldots,y_n$
- the objective fn is $\sum_i y_i$
- The constraints are
  $Ax=b$
  $y_i \geq -x_i$
  $y_i \leq x_i$

We can solve (LP) efficiently – the question is, does it give the correct answer?

Geometric intuition:



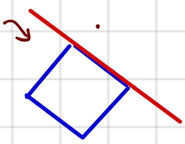
level sets of the objective function $\|x\|_1$

the linear space $Ax=b$

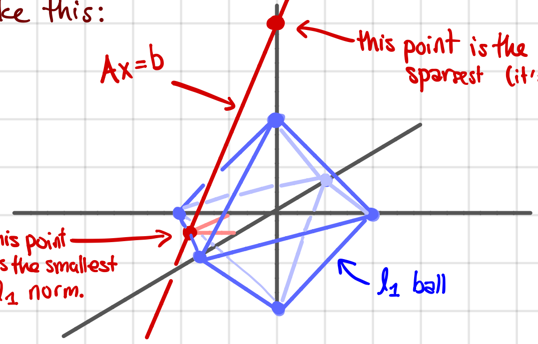
The point in this subspace that minimizes $\|x\|_1$ also happens to be sparse!

The picture indicates that this might be a good idea since the ℓ_1 ball is "pointy." Of course, it won't always work – but hopefully if the set $\{x : Ax=b\}$ is "generic" enough, our intuition will be OK.

WARNING: The 2-dimensional picture above might lead you to conclude that there's really no problem at all, and that the only bad case is like this



In fact, it shouldn't be obvious that this intuition works in higher dimensions. For example, in 3-dimensions, the picture could look like this:



$Ax=b$

this point is the sparsest (it's 1-sparse)

this point has the smallest ℓ_1 norm.

ℓ_1 ball

3B What is the condition we should place on A?

It's natural to look at sparse vectors in the kernel of A.
Indeed, an obvious obstacle to finding x is if there are TWO sparse solutions:

$$b = Ax_1 = Ax_2 \qquad \text{and } \|x_1\|_0, \|x_2\|_0 \leq k.$$

$$A(\underbrace{x_1 - x_2}_{2k\text{ - sparse.}}) = 0$$

So we definitely need:

There are no sparse vectors in the kernel of A.

We'll just strengthen this a little bit:

There are no SPARSE-ISH vectors in the kernel of A.

We'll say that a vector is "sparse-ish" if it's not too spread out.

DEF A vector $x \in \mathbb{R}^n$ is (k,c)-sparse-ish if

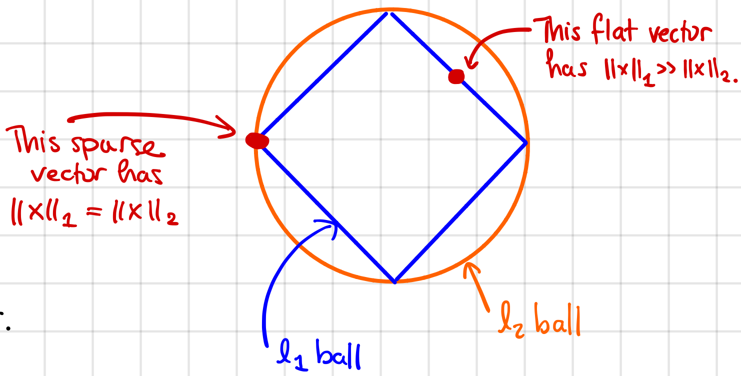
$$\|x\|_2 > \frac{c}{\sqrt{n}} \|x\|_1.$$

Why should this definition have anything to do with sparsity?

We always have

$$\frac{1}{\sqrt{n}}\|x\|_1 \overset{①}{\leq} \|x\|_2 \overset{②}{\leq} \|x\|_1$$

Cauchy-Schwarz (①) Just true (②)



② is tight for 1-sparse vectors

① is tight for completely flat vectors.

So the condition that $\|x\|_2 > \frac{c}{\sqrt{n}}\|x\|_1$ is bounding x away from the flat case, toward the sparse case.

4 FORMAL(ish) STATEMENT and PROOF.

Theorem 1. Suppose that A has no $\frac{1}{4}\sqrt{\frac{n}{k}}$ – sparseish vectors in $\mathrm{Ker}(A)$.
Let $x \in \mathbb{R}^n$ be k-sparse, and let $b = Ax$.
Then (LP) returns $\hat{x} = x$.

Theorem 1 is made much more interesting by the following fact.

Theorem 2. If $m = \Omega(k \log(n))$, then a $m \times n$ matrix A whose entries are iid Gaussian has no $\frac{1}{4}\sqrt{\frac{n}{k}}$ –vectors in its kernel with high probability.

We will punt on the proof of Thm 2 (it's not so hard), and focus here of the proof of Thm 1.

Proof of Theorem 1.

Suppose that (LP) returns w, while the true answer is x: So x is k-sparse
this implies that $\|w\|_1 \le \|x\|_1$.

We'd like to show that this can't happen (unless $w = x$).
That is, we'd like to show:

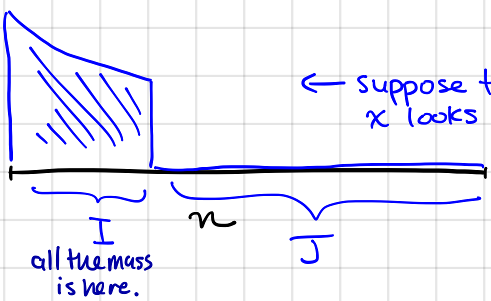
$$\boxed{\text{For all } \omega \text{ so that } A\omega = Ax, \quad \|\omega\|_1 > \|x\|_1.}$$

Write $w = x + y$ — $y \in \mathrm{Ker}(A)$, so by assumption it is NOT $\frac{1}{4}\sqrt{\frac{n}{k}}$ - sparseish.

Let I be the support of x, so $|I| \le k$.
Let $J = [n] \setminus I$

Aside: Thm 2 also holds if A has iid ± 1 entries, or many other natural random ensembles. It also holds (possibly at the cost of a few logarithmic factors) for randomly sampled rows of a DFT, as per the MRI example.
Finding explicit constructions of such matrices is a big open problem.

Another aside:
Another common sufficient condition is the "Restricted Isometry Property" (RIP), which says that $\forall$ sparse x,
$(1-\epsilon)\|x\|_2 \le \|Ax\|_2 \le (1+\epsilon)\|x\|_2$
for some small ϵ.



← suppose that x looks like this.

← y could look like anything that's not too sparse-ish.

For a vector z, let z_I denote the restriction of z to the indices in I, with the rest of the coordinates zeroed out.

Thus,

$$\|w\|_1 = \|x+y\|_1$$
$$= \|(x+y)_I\|_1 + \|(x+y)_J\|_1$$

Triangle inequality

$$\|(x+y)_I\|_1 \geq \|x_I\|_1 - \|y_I\|_1 = \|x\|_1 - \|y_I\|_1$$

$$\|(x+y)_J\|_1 = \|y_J\|_1 = \|y\|_1 - \|y_I\|_1$$

So $\|w\|_1 \geq \|x\|_1 + \|y\|_1 - \underbrace{2\|y_I\|_1}_{\text{This is small since } y \text{ isn't very sparse-ish.}}$

MORE PRECISELY

CLAIM: $\|y_I\|_1 \leq \frac{1}{4}\|y\|_1$

Proof. $\|y_I\|_1 \leq \sqrt{k}\,\|y_I\|_2 \leq \sqrt{k}\,\|y\|_2 \leq \sqrt{k}\cdot\left(\frac{1}{4}\sqrt{\frac{n}{k}}\right)\cdot\frac{1}{\sqrt{n}}\|y\|_1 = \frac{1}{4}\|y\|_1.$ ∎

Cauchy-Shwarz

y is just y_I with more stuff

def. of Sparse-ish

simplification.

Thus,

$$\|w\|_1 \geq \|x\|_1 + \frac{1}{2}\|y\|_1 \gneq \|x\|_1, \quad \text{unless } y=0 \text{ (in which case } x=w).$$

This is what we wanted to prove, so now we're done. :)

This shows that, for "most" matrices A, (LP) will actually solve our problem efficiently!

RECAP.

- Often, we want to find a (unique) sparse x so that $Ax=b$, for a short fat matrix A.
- This problem is impossible without the sparsity assm, and NP hard (w/ inputs A, b) in the worst-case.
- However, for "most" matrices A with $\Omega(k\log(n/k))$ rows, it is tractable, and an LP gives exactly the right answer!!
- We only talked about exact sparsity and exact recovery, but it turns out that these results can be made robust to noise.