1 Last time

Last Wednesday, we introduced Compressed Sensing. The setup was that we would like to find a vector \( x \in \mathbb{R}^n \). However, \( n \) is large, so we instead get an \( m \)-dimensional linear sketch \( Ax \), for \( m << n \). We would like to use \( Ax \) to recover an approximation \( x^* \) to \( x \). And in contrast to earlier algorithms in this course (namely Count-Min and Count-Sketch), we would like a single matrix \( A \) to work for all vectors \( x \).

We showed that if \( A \) is a random Gaussian matrix and \( m = \Theta(k \log(n/k)) \), with high probability \( A \) satisfies the Restricted Isometry Property with parameters \((ck, \delta)\), or the \((ck, \delta)\)-RIP:

\[(1 - \delta) \|x\|_2 \leq \|Ax\|_2 \leq (1 + \delta) \|x\|_2\]

for all \( ck \)-sparse \( x \) (that is, \( x \) with at most \( ck \) non-zero coefficients). We then stated, but did not show, that the \((2k, \delta)\)-RIP implies another property, the null space property: for any vector \( \eta \) with \( A\eta = 0 \) and any set of coordinates \( S \) with \( |S| \leq k \),

\[\|\eta_S\|_1 \leq \varepsilon \|\eta\|_1\]

for some \( \varepsilon = O(\delta) \). We then showed that the null space property implies that the L1 minimization program

\[
\begin{align*}
\min_{x} & \quad \|x^*\|_1 \\
\text{subject to} & \quad Ax^* = Ax
\end{align*}
\]

solves the sparse recovery problem, in the sense that

\[\|x - x^*\|_1 \leq C \min_{k\text{-sparse } x'} \|x - x'\|_1.\]

So if \( x \) is \( k \)-sparse, the program recovers \( x \) exactly. If \( x \) is close to \( k \)-sparse, the result is close to \( x \).

2 This time

Today, we'll show how to make \( A \) be a sparse matrix. This makes computing \( Ax \) much more efficient, and speeds up L1 minimization (whose bottleneck is computing matrix products). This class will proceed much like the previous one. We will show that a random sparse binary matrix \( A \) with high probability satisfies an \( \ell_1 \) norm analog of the RIP, known as the RIP-1. We will also show that they satisfy the same null space property as before, so L1 minimization still works.

Next time we'll show good, fast recovery algorithms to use instead of L1 minimization.
3 Dense v. Sparse matrices

Dense matrices have traditionally been used in compressed sensing. Dense Gaussian matrices allow very short sketches ($O(k \log(n/k))$), but are fairly slow to use: computing $Ax$ takes $O(nm)$ time, and reconstructing $x$ from $Ax$ with linear programming takes $\text{poly}(n)$ time. One can speed this up by using subsampled Fourier matrices, since computing $Ax$ can be done in $O(n \log n)$ time with an FFT. However, this requires sacrificing sketch size to $O(k \log \alpha n)$ for constant $\alpha \approx 4$.

By contrast, data stream algorithms have used sparse matrices (as with Count-Min or Count-Sketch from Lecture 6). These matrices only have $O(\log n)$ non-zero entries per column, so computing $Ax$ takes only $O(n \log n)$ time. Furthermore, reconstructing $x$ from $Ax$ also only takes $O(n \log n)$ time. Unfortunately, these algorithms are randomized, and they have larger sketches of size $O(k \log n)$.

The difference between $O(k \log(n/k))$ and $O(k \log n)$ may not seem like much, but it can be significant in practical situations where $k$ is large. If $k = n/C$, then the former has a relative “space savings” $n/m$ of $C/\log C$, while the latter only has $C/\log n$. If $C$ is relatively small, then the former is an improvement over the trivial algorithm and the latter may not be.

Ideally, we could get the “best of both worlds”: an $O(k \log(n/k))$ deterministic sketch using sparse matrices, with an efficient recovery algorithm. We’ll do this today and in the next lecture.

4 How could we do compressed sensing on sparse matrices?

Recall that an integral part of our analysis of dense matrices was the Restricted Isometry Property (RIP) [CRT06]: for all $k$-sparse $\Delta$,

$$(1 - \delta) \|\Delta\|_2 \leq \|A\Delta\|_2 \leq (1 + \delta) \|\Delta\|_2$$

for some small constant $\delta$. With high probability, a random Gaussian or Bernoulli $\pm 1$ matrix satisfies the RIP with $m = O(k \log(n/k))$. Can the same be true for sparse matrices? Or maybe even all $m \times n$ sparse binary matrices, with $d$ ones per column?

Not really. Binary matrices require $m = \Omega(\min(k^2, n))$ regardless of their sparsity [Cha10]. Even non-binary matrices require at least $\Omega(\min(n/m, k))$ non-zero entries per column.

So instead, we’ll avoid the RIP, and show an analog in the $\ell_1$ norm, the RIP-1 [BGI+08]: for all $k$-sparse $\Delta$,

$$d(1 - \delta) \|\Delta\|_1 \leq \|A\Delta\|_1 \leq d \|\Delta\|_1$$

for some constant $\delta$. This is just like the RIP-2, but in the $\ell_1$ norm, and rescaled for convenience.

We will show that a sufficient condition is when $A$ represents the adjacency matrix of a $(k, d(1 - \varepsilon/2))$-expander. It’s also a necessary condition, at least when $A$ is a binary matrix. We’ll define such expanders in the next section.

5 Expanders

**Definition 1.** A $(l, \varepsilon)$-unbalanced expander is a bipartite simple graph $G = (U, V, E)$, $|U| = n, |V| = m$, with left degree $d$ such that for any $X \subseteq U$ with $|X| \leq l$, the set of neighbors $N(X)$ of $X$ has size $|N(X)| \geq (1 - \varepsilon)d|X|$. 

We also define \( E(X : Y) = E \cap (X \times Y) \) to be the set of edges between the sets \( X \) and \( Y \).

Our goal is to find expanders with small \( m \) and \( d \), given \( n \) and \( \epsilon \). Expanders have been extensively studied in computer science and coding theory, and it is well known that a random graph is a good expander:

**Claim 1.** For any \( n/2 \geq l \geq 1, \epsilon > 0 \), there exists a \((l, \epsilon)\)-unbalanced expander with left degree \( d = O(\log(n/l)/\epsilon) \) and right set size \( m = O(ld/\epsilon) = O(l \log(n/l)/\epsilon^2) \).

The proof is "folklore". For completeness, we reproduce the proof given in [Ber09].

**Proof.** Consider graphs \( G = (U, V, E) \) with \( |U| = n \) and \( |V| = m \). Let \( d = \ln(ne^2/l)/\epsilon \) and \( m = e^2 k \ln(ne^2/l)/\epsilon^2 \). We show that a random graph \( G \) is with constant probability a \((l, d, \epsilon)\)-unbalanced expander\(^1\). A random graph is generated by randomly and independently choosing each of the \( d \) neighbors for each left vertex.

Consider any of the \( \binom{n}{s} \) left-vertex sets of size \( s \leq l \). The \( s \) vertices have \( d \) neighbors each; consider a sequence containing the \( ds \) vertex indices. For \( G \) to fail to be an expander on this set, at least \( \epsilon ds \) of these values must be “repeats”, i.e. be identical to some earlier value in the sequence. The probability that a given neighbor is a repeat is at most \( ds/m \). By the union bound, the probability that \( G \) fails to expand at least one of these sets is at most

\[
\left( \frac{n}{s} \right) \left( \frac{ds}{\epsilon ds} \right) \left( \frac{e^d}{m} \right) \leq \left( \frac{ne}{s} \right)^s \left( \frac{e^d}{s} \right) \left( \frac{se^2}{\epsilon s} \right) \leq \left( \frac{ne}{s} \right)^s \left( \frac{e^d}{s} \right) \leq \left( \frac{1}{e} \right)^{s} \left( \frac{e^d}{s} \right) \leq e^{-s}
\]

Thus the probability that \( G \) is not a \((k, d, \epsilon)\)-expander is at most

\[
\sum_{s=1}^{l} e^{-s} < 1 - \sum_{s=0}^{\infty} e^{-s} = \frac{1}{e} \frac{1}{1 - 1/e} = \frac{1}{e - 1} < 0.59
\]

Therefore, almost every graph with \( m = \Theta(l \log(n/l)) \) and \( d = \Theta(\log(n/l)) \) is a good expander. However, it’s not known how to efficiently find one explicitly, nor how to efficiently check whether a graph is an expander. The best known explicit constructions of expanders [GUV07] require \( m = \Theta(l^{1+\alpha} \log^{2+2\alpha} n) \) for some \( \alpha > 0 \).

Consider \( A \) to be the \( m \times n \) adjacency matrix of a \((2k, \epsilon/2)\)-unbalanced expander, where the left set corresponds to columns and the right set to rows. So \( A_{ij} = 1 \) if \((j, i)\) is an edge in the graph, and 0 otherwise. For constant \( \epsilon \), we can have \( m = O(k \log(n/k)) \). We will show that \( A \) satisfies the RIP-1.

\(^1\)Note that this analysis is not tight in terms of constants or success probability.
6 Expanders satisfy RIP-1

We want that for any $k$-sparse $\Delta$,

$$d(1 - \varepsilon) \|\Delta\|_1 \leq \|A\Delta\|_1 \leq d\|\Delta\|_1.$$ 

The right hand side trivially holds for any $\Delta$, since each column of $A$ has $d$ ones. So we focus on the left hand side.

Without loss of generality, suppose $\Delta$ is sorted by decreasing magnitude:

$$|\Delta_1| \geq |\Delta_2| \geq \cdots \geq |\Delta_k| \geq |\Delta_{k+1}| = \cdots = |\Delta_n| = 0.$$

Now consider the edges $e = (i,j)$ sorted in lexicographical order. For each edge $e = (i,j)$ define $r(e) = -1$ if there exists an edge $(i',j)$ with $i' < i$, and $r(e) = 1$ if there is no such edge. By the expansion property, we know that any prefix $P$ of the edges with $|P| \leq kd$ satisfies

$$|\{ e \in P \mid r(e) = -1 \}| \leq \varepsilon |P|/2. \quad (1)$$

We make two more claims:

$$\|A\Delta\|_1 \geq \sum_{e=(i,j)} |\Delta_i| r(e). \quad (2)$$

$$\sum_{e=(i,j)} |\Delta_i| r(e) \geq (1 - \varepsilon)d\|\Delta\|_1. \quad (3)$$

Claim (2) is quite straightforward:

$$\|A\Delta\|_1 = \sum_j \left| \sum_{e=(i,j)} \Delta_i \right| \geq \sum_j \sum_{e=(i,j)} |\Delta_i| r(e) = \sum_{e=(i,j)} |\Delta_i| r(e).$$

where the inequality is the triangle inequality: $|a_1 + \cdots + a_s| \geq |a_1| - |a_2| - \cdots - |a_s|$ for any $a,s$.

Claim (3) follows from Equation 1 and the fact that the $|\Delta_i|$ are sorted. Informally, the “worst cast” is if the $r(e)$ are $-1$ as early and often as possible. But this is only at a rate of $\varepsilon/2$, and $\varepsilon/2$ fraction negative mass can cancel out an $\varepsilon/2$ fraction of positive mass, leaving $1 - \varepsilon$ remaining.

More formally, define $z_e = |\Delta_i|$ for $e = (i,j)$. Now consider them in sequence as $z_1, \ldots, z_{kd}$, and similarly sort $r(e)$ as $r(1), \ldots, r(kd)$. Define the differences $y_i = z_i - z_{i+1}$ (for $i < kd$, with $y_{kd} = z_{kd}$), so $y_i \geq 0$ and $z_i = \sum_{j \geq i} y_j$ for all $i$.

From Equation 1, we have that for any $j \leq kd$,

$$\sum_{i=1}^{j} r(i) \geq (1 - \varepsilon)j.$$
Therefore
\[
\sum e z e r(e) = \sum_{i=1}^{kd} r(i) z_i \\
= \sum_{i=1}^{kd} r(i) \sum_{j=i}^{kd} y_j \\
= \sum_{1 \leq i \leq j \leq kd} r(i) y_j \\
= \sum_{j=1}^{kd} y_j \sum_{i=1}^{j} r(i) \\
\geq \sum_{j=1}^{kd} y_j (1 - \varepsilon) j \\
= (1 - \varepsilon) \sum_{1 \leq i \leq j \leq kd} y_j \\
= (1 - \varepsilon) \sum_{i=1}^{kd} z_i \\
= (1 - \varepsilon) \| \Delta \|_1
\]
as desired. So we have \( \| A \Delta \|_1 \geq \sum_{e=(i,j)} |\Delta_i| r(e) \geq (1 - \varepsilon) d \| \Delta \|_1 \), which gives the RIP-1.

7 Expanders satisfy null space property

We have shown the RIP-1 is satisfied by an expander, but we really want to show that L1 minimization works on expanders. To do this, we will show that expanders also satisfy the null space property (using both the RIP-1 and expansion properties), so L1 minimization works by the same argument as before.

Lemma 2. Let \( A \) be the \( m \times n \) adjacency matrix of an unbalanced \((2k, \varepsilon)\)-expander \( G \) with left degree \( d \), and let \( \alpha(\varepsilon) = (2\varepsilon)/(1 - 2\varepsilon) \). Consider any \( \eta \in \mathbb{R}^n \) such that \( A \eta = 0 \), and let \( S \) be any set of \( k \) coordinates of \( \eta \). Then we have

\[
\| \eta_S \|_1 \leq \alpha(\varepsilon) \| \eta \|_1
\]

Proof. Without loss of generality, we can assume that \( S \) consists of the largest (in magnitude) coefficients of \( \eta \). We partition coordinates into sets \( S_0, S_1, S_2, \ldots, S_t \), such that (i) the coordinates in the set \( S_l \) are no larger (in magnitude) than the coordinates in the set \( S_{l-1} \) for \( l \geq 1 \), and (ii) all sets but \( S_t \) have size \( k \). Therefore, \( S_0 = S \). Let \( A' \) be a submatrix of \( A \) containing rows from \( N(S) \). Define \( S^c = [n] \setminus S \).

The basic idea of the proof is as follows. Assume (by contradiction) that \( \| \eta_S \|_1 \) is “large” compared to \( \| \eta \|_1 \), which (by the RIP-1) implies that \( \| A' \eta_S \|_1 \) is “large”. Since \( 0 = \| A' \eta \|_1 = \| A' \eta_S + A' \eta^c \|_1 \), it follows that \( \| A' \eta^c \|_1 \) must be “large”, to cancel the contribution of \( A' \eta_S \). The
only way for this to happen though is if there are many edges in $G$ from $S$ to $N(S)$. This however would mean that the neighborhoods of $S$ and blocks $S_i$ have large overlaps, which cannot happen since the graph is an expander.

The formal proof follows.

From the RIP-1 property we know that $\|A' \eta_S\|_1 = \|A \eta_S\|_1 \geq d(1 - 2\epsilon) \|\eta_S\|_1$. At the same time, we know that $\|A' \eta\|_1 = 0$. Therefore

$$0 = \|A' \eta\|_1 \geq \|A' \eta_S\|_1 - \sum_{l \geq 1} \sum_{(i,j) \in E : i \in S_l, j \in N(S)} |\eta_i|$$

$$\geq d(1 - 2\epsilon) \|\eta_S\|_1 - \sum_{l \geq 1} |E(S_l : N(S))| \min_{i \in S_{l-1}} |\eta_i|$$

$$\geq d(1 - 2\epsilon) \|\eta_S\|_1 - \sum_{l \geq 1} |E(S_l : N(S))| \cdot \|\eta_{S_{l-1}}\|_1 / k$$

From the expansion properties of $G$ it follows that, for $l \geq 1$, we have $|N(S \cup S_l)| \geq d(1 - \epsilon)|S \cup S_l|$. It follows that at most $de2k$ edges can cross from $S_l$ to $N(S)$, and therefore

$$0 \geq d(1 - 2\epsilon) \|\eta_S\|_1 - \sum_{l \geq 1} |E(S_l : N(S))| \cdot \|\eta_{S_{l-1}}\|_1 / k$$

$$\geq d(1 - 2\epsilon) \|\eta_S\|_1 - de2k \sum_{l \geq 1} \|\eta_{S_{l-1}}\|_1 / k$$

$$\geq d(1 - 2\epsilon) \|\eta_S\|_1 - 2de \|\eta\|_1$$

It follows that $d(1 - 2\epsilon) \|\eta_S\|_1 \leq 2de \|\eta\|_1$, and thus $\|\eta_S\|_1 \leq (2\epsilon)/(1 - 2\epsilon) \|\eta\|_1$. \qed

Therefore $A$ satisfies the null space property, so L1 minimization works.

8 Conclusion

We now have a scheme with $O(k \log(n/k))$ sparse measurements. The recovery time is still polynomial, but it’s a smaller polynomial than before: large scale linear programming is usually solved by interior point methods, which work by iteratively computing $Az$ for vectors $z$. When $A$ is sparse, this is faster.

We can implement this, and see empirically how sparse matrices compare to dense ones. In particular, we can consider the probability of successfully recovering a sparse signal with L1 minimization for various values of $k$ and $m$. If you compare the empirical results on sparse binary matrices [Ber09] and the analytical results for Gaussian matrices [DT06], the two curves match almost exactly. One project idea is to come up with an analytical explanation for this. As fair warning, the analysis for Gaussians is horrendous, so a first step might be to simplify that.

Next time we’ll see how to avoid L1 minimization and go much faster.

References


