Today we'll talk about dimensionality reduction, and some related topics in data streaming.

1 Dimension Reduction

Suppose we are given a set of n points $\{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^D$. How small can we make D and still maintain the Euclidean distances between the points? Clearly, we can always make $D = n - 1$, since any set of n points lies on a $n-1$-dimensional subspace. And this is (existentially) tight: e.g., the case when $x_2 - x_1, x_3 - x_1, \ldots, x_n - x_1$ are all orthogonal vectors.

But what if we were OK with the distances being approximately preserved? In HW#3, you saw that while there could only be D orthogonal unit vectors in $\mathbb{R}^D$, there could be as many as $\exp(c\varepsilon^2 D)$ unit vectors which are ε-orthogonal—i.e., whose mutual inner products all lie in $[-\varepsilon, \varepsilon]$. Near-orthogonality allows us to pack exponentially more vectors!

Put another way, note that

$$\|\vec{a} - \vec{b}\|_2^2 = \langle \vec{a} - \vec{b}, \vec{a} - \vec{b} \rangle = \langle \vec{a}, \vec{a} \rangle + \langle \vec{b}, \vec{b} \rangle - 2\langle \vec{a}, \vec{b} \rangle = \|\vec{a}\|_2^2 + \|\vec{b}\|_2^2 - 2\langle \vec{a}, \vec{b} \rangle.$$

And hence the squared Euclidean distance between any pair of the points defined by these ε-orthogonal vectors falls in $2(1 \pm \varepsilon)$. So, if we wanted n points exactly at unit (Euclidean) distance from each other, we would need $n-1$ dimensions. (Think of a triangle in 2-dims.) But if we wanted to pack in n points which were at distance $(1 \pm \varepsilon)$ from each other, we could pack them into

$$O\left(\frac{\log n}{\varepsilon^2}\right)$$

dimensions.

1.1 The Johnson Lindenstrauss lemma

The Johnson Lindenstrauss "flattening" lemma says that such a claim is true not just for equidistant points, but for any set of n points in Euclidean space:

Lemma 1 *Let $\varepsilon \in (0, 1/2)$. Given any set of points $X = \{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^D$, there exists a map $A : \mathbb{R}^D \to \mathbb{R}^k$ with $k = O(\varepsilon^{-2} \log n)$ such that*

$$1 - \varepsilon \leq \frac{\|A(x_i) - A(x_j)\|_2^2}{\|x_i - x_j\|_2^2} \leq 1 + \varepsilon.$$

Note that the target dimension k is independent of the original dimension D, and depends only on the number of points n and the accuracy parameter ε.

This lemma is tight up to the constant term: it is easy to see that we need at least $\Omega(\frac{1}{\varepsilon} \log n)$ using a packing argument. Noga Alon showed a lower bound of $\Omega(\frac{\log n}{\varepsilon^2 \log 1/\varepsilon})$.

1.2 The construction

The JL lemma is pretty surprising, but the construction of the map is perhaps even more surprising: it is a super-simple random construction. Let M be a $k \times D$ matrix, such that every entry of M is filled with an i.i.d. draw from a standard normal $N(0,1)$ distribution (a.k.a. the "Gaussian" distribution). For $x \in \mathbb{R}^D$, define

$$A(x) = \frac{1}{\sqrt{k}} Mx.$$

That's it. You hit the vector x with a Gaussian matrix M, and scale it down by $\sqrt{k}$. That's the map A. Note that it is a linear map: $A(x) + A(y) = A(x+y)$. So suppose we could show the following lemma:

Lemma 2 *Let $\varepsilon \in (0, 1/2)$. If A is constructed as above with $k = c\varepsilon^{-2} \log \delta^{-1}$, and $x \in \mathbb{R}^D$ is a unit vector, then*

$$\Pr[\|A(x)\|_2^2 \in 1 \pm \varepsilon] \geq 1 - \delta.$$

Then we'd get a proof of Lemma 1. Indeed, set $\delta = 1/n^2$, and hence $k = O(\varepsilon^{-2} \log n)$. Now for each $x_i, x_j \in X$ we get that the squared length of $x_i - x_j$ is maintained to within $1 \pm \varepsilon$ with probability at least $1 - 1/n^2$. By a union bound, all $\binom{n}{2}$ pairs of distances in $\binom{X}{2}$ are maintained with probability at least $1 - \binom{n}{2}\frac{1}{n^2} \geq 1/2$. This proves Lemma 1.

A few comments about this construction:

- The above proof shows not only the existence of a good map, we also get that a random map as above works with constant probability! In other words, a Monte-Carlo randomized algorithm for dimension reduction. (Since we can efficiently check that the distances are preserved to within the prescribed bounds, we can convert this into a Las Vegas algorithm.)
- The algorithm (at least the Monte Carlo version) *does not even look* at the set of points X: it works for any set X with high probability. Hence, we can pick this map A before the points in X arrive.
- Given a set $X \subseteq \mathbb{R}^D$, one can get deterministic poly-time algorithms constructing a dimension reduction map $A : \mathbb{R}^D \to \mathbb{R}^k$ for $k = O(\varepsilon^{-2} \log |X|)$: the first one was given in this paper of Lars Engebretsen, Piotr Indyk and Ryan O'Donnell; another construction is due to D. Sivakumar.

A SODA 2011 paper of T.S. Jayram and David Woodruff shows that this dependence of $O(\varepsilon^{-2} \log \delta^{-1})$ is the best possible. Note that *if we use this approach using this lemma and the union bound* to prove JL, then $O(\varepsilon^{-2} \log n)$ is the best bound possible. (An earlier version of these notes incorrectly claimed that the Jayram-Woodruff paper also showed an unconditional lower bound for JL, thanks to Jelani for pointing out the mistake.)

1.3 The proof

Now, on to the proof of Lemma 2. Here's the main idea. Imagine that the vector we're considering is just the elementary unit vector $e_1 = (1, 0, \ldots, 0)$. Then $M\, e_1$ is just a vector with independent and identical Gaussian values, and we're interested in its length—the sum of squares of these Gaussians.

If these were bounded r.v.s, we'd be done—but they are not. However, their tails are very small, so things should work out

But what's a Gaussian $N(0,1)$? Well, it looks like this:

Which is not too different from this (bounded) random variable, if you squint a bit:

Which has constant mean. So, if we take a sum of a bunch of such random variables (actually of their squares), it should behave pretty much like its mean (which is $\propto k$), because of a Chernoff-like argument. And so the expected length is close to $\sqrt{k}$, which explains the division by $\sqrt{k}$.

Now we just need to make all this precise, and remove the assumption that the vector was just e_1. That's what the rest of the formal proof does: it has a few steps, but each of them is fairly elementary.

1.4 The proof, this time for real

We'll be using basic facts about Gaussians, let's just recall them. The probability density function for the Gaussian $N(\mu, \sigma^2)$ is

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}}.$$

We also use the following; the proof just needs some elbow grease.

Proposition 3 *If $Y_1 \sim N(\mu_1, \sigma_1^2)$ and $Y_2 \sim N(\mu_2, \sigma_2^2)$, then*

$$Y_1 + Y_2 \sim N(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2).$$

Recall that we want to argue about the squared length of $A(x) \in \mathbb{R}^k$. To start off, observe that each coordinate of the vector Mx behaves like

$$Y \sim \langle G_1, G_2, \ldots, G_D \rangle \cdot x = \sum x_i \, G_i$$

where the G_i's are i.i.d. $N(0,1)$ r.v.s. But then the proposition tells us that $Y \sim N(0, x_1^2 + x_2^2 + \ldots + x_D^2)$. And since x is a unit length vector, this is simply $N(0,1)$. So, each of the k coordinates of Mx behaves just like an independent Gaussian!

What is the squared length of $A(x) = \frac{1}{\sqrt{k}} Mx$, then? It is

$$Z := \sum_{i=1}^{k} \frac{1}{k} \cdot Y_i^2$$

where each $Y_i \sim N(0,1)$, independent of the others. And since $E[Y_i^2] = \mathrm{Var}(Y_i) + E[Y_i]^2 = 1$, we get $E[Z] = 1$.

Now to show that Z does not deviate too much from 1. And Z is the sum of a bunch of independent and identical random variables. If only the Y_i's were all bounded, we could have used a Chernoff bound and be done. But these are not bounded, so this is finally where we'll need to do a little work. (Note: we could take the easy way out, observe that the squares of Gaussians are chi-squared r.v.s, the sum of k of them is *chi-squared with k degrees of freedom*, and the internets conveniently has tail bounds for these things. But we digress.)

So let's start down the ye olde Chernoff path, for the upper tail, say:

$$\Pr[Z \geq 1+\varepsilon] \leq \Pr[e^{tkZ} \geq e^{tk(1+\varepsilon)}] \leq E[e^{tkZ}]/e^{tk(1+\varepsilon)} = \prod_i \left(E[e^{tY_i^2}]/e^{t(1+\varepsilon)} \right) \tag{1}$$

for every $t > 0$. And what is $E[e^{tY^2}]$ for $Y \sim N(0,1)$? Let's calculate it:

$$\frac{1}{\sqrt{2\pi}} \int_y e^{ty^2} e^{-y^2/2} dy = \frac{1}{\sqrt{2\pi}} \int_z e^{-z^2/2} \frac{dz}{\sqrt{1-2t}} = \frac{1}{\sqrt{1-2t}}. \tag{2}$$

for $t < 1/2$. So our current bound on the upper tail is that for all $t \in (0, 1/2)$ we have

$$\Pr[Z \geq (1+\varepsilon)] \leq \left(\frac{1}{e^{t(1+\varepsilon)}\sqrt{1-2t}} \right)^k.$$

Let's just focus on part of this expression:

$$\begin{aligned}
\left(\frac{1}{e^t \sqrt{1-2t}} \right) &= \exp\left(-t - \frac{1}{2}\log(1-2t) \right) \\
&= \exp\left((2t)^2/4 + (2t)^3/6 + \cdots \right) \leq \exp\left(t^2(1 + 2t + 2t^2 + \cdots) \right) \\
&= \exp(t^2/(1-2t)).
\end{aligned}$$

Plugging this back, we get

$$\begin{aligned}
\Pr[Z \geq (1+\varepsilon)] &\leq \left(\frac{1}{e^{t(1+\varepsilon)}\sqrt{1-2t}} \right)^k \\
&\leq \exp(kt^2/(1-2t) - kt\varepsilon) \leq e^{-k\varepsilon^2/8},
\end{aligned}$$

if we set $t = \varepsilon/4$ and use the fact that $1 - 2t \geq 1/2$ for $\varepsilon \leq 1/2$. (Note: this setting of t also satisfies $t \in (0, 1/2)$, which we needed from our previous calculations.)

Almost done: let's take stock of the situation. We observed that $\|A(x)\|_2^2$ was distributed like a sum of squares of Gaussians, and using that we proved that

$$\Pr[\|A(x)\|_2^2 > 1 + \varepsilon] \leq \exp(-k\varepsilon^2/8) \leq \delta/2$$

for $k = \frac{8}{\varepsilon^2} \ln \frac{2}{\delta}$. A similar calculation bounds the lower tail, and finishes the proof of Lemma 2.

Citations: The JL Lemma was first proved in this paper of Bill Johnson and Joram Lindenstrauss. There have been several proofs after theirs, usually trying to tighten their results, or simplify the algorithm/proof (see citations in some of the newer papers): the proof follows some combinations of the proofs in this STOC '98 paper of Piotr Indyk and Rajeev Motwani, and this paper by Sanjoy Dasgupta and myself.

2 The data stream model

The JL map we considered was a linear map, and that has many advantages. One of them is that we can use it in a distributed context: if t players each have a vector $\vec{y}_i$ and each knows the JL matrix A, then to compute $A(\sum_i \vec{y}_i)$ each person can just compute $A\vec{y}_i$, send their answers out, and then someone can sum up the answers to get $\sum_i A(\vec{y}_i) = A(\sum_i \vec{y}_i)$. Since these vectors $A\vec{y}_i$ are smaller than $\vec{y}_i$ (they lie in $\mathbb{R}^k$ instead of $\mathbb{R}^D$), this can result in significant savings in communication. (We need all players to know the matrix A, but if they have shared randomness they can generate this matrix themselves.)

This same idea is useful in the context of data streaming: suppose you have a data stream of a large number of elements $\sigma_1, \sigma_2, \ldots$, whizzing past you, each element σ_j drawn from the universe $[D]$. This stream defines a frequency vector $\vec{x} \in \mathbb{R}^D$, where x_i is the number of times element i is seen. People working on data streams want to calculate statistics of this vector $\vec{x}$—e.g., how many non-zeroes does it have? What is the ℓ_1 length of this? (Duh! it's just the length of the data stream.) What is $\sum_i x_i^2 = \|\vec{x}\|_2^2$? Etc.

The Space Crunch. All this can be trivially done if we use D space to actually store the vector $\vec{x}$. Suppose we do not want to store the frequency vector explicitly, but are OK with approximate answers. We can use JL or similar schemes to approximately calculate $\sum_i x_i^2$. Suppose A is a random $k \times D$ Gaussian matrix, then by the guarantee of the JL lemma, the estimate $\|A\vec{x}\|_2^2 \in (1 \pm \varepsilon)\|\vec{x}\|_2^2$ with probability $1 - \delta$, if $k = \Omega(1/\varepsilon^2 \log 1/\delta)$. (Note: this is the error for a single query—so we're not guaranteeing the counts at *all times* are close, just at the time the query is made.)

And the algorithm is simple: maintain a vector $y \in \mathbb{R}^k$, initially zero. When the element $j \in [D]$ comes by, add in the j^{th} column of A to y. Finally, answer with $\|y\|_2^2$. (If you have to answer t queries, choose k appropriately larger.)

Of course, you've realized I am cheating. In order to save space we used JL. But the JL matrix itself uses kD entries, which is a lot of space, much more than the D entries of the frequency vector $\vec{x}$! Also, we now need to maintain a matrix of reals, whereas $\vec{x}$ just has integers!

We can handle both issues. The former issue can directly be handled by using a *pseudorandom generator* that "fools" low-space computation—we will not talk about this solution in this lecture. Instead we'll give a different (though weaker) solution which handles both issues: it will use less space, and will maintain only integer values (if the input has integers).

3 Using random signs instead of Gaussians

While Gaussians have all kinds of nice properties, they are real-valued distributions and hence require attention to precision. How about populating A with draws from other, simpler distributions? How about setting each $M_{ij} \in_R \{-1, +1\}$, and letting $A = \frac{1}{\sqrt{k}} M$? (A random sign is also called a *Rademacher random variables*, btw, the name Bernoulli being already taken for a random bit in $\{0, 1\}$.)

Now, we want to study the properties of

$$Z := \sum_{i=1}^{k} \left(\sum_{j=1}^{D} A_{ij} \cdot x_j \right)^2 . \tag{3}$$

To keep subscripts to a minimum, consider the inner expression

$$Y = (\sum_j X_j \cdot x_j)^2$$

where each $X_i \in_R \{-1, 1\}$. Then

$$\begin{aligned} E[Y] &= E[(\textstyle\sum_j X_j x_j)(\sum_l X_l x_l)] \\ &= E[\textstyle\sum_j X_j^2 x_j^2 + \sum_{j \neq l} X_j X_l x_j x_l] \\ &= \textstyle\sum_j E[X_j^2] x_j^2 + \sum_{j \neq l} E[X_j X_l] x_j x_l = \sum_j x_j^2. \end{aligned}$$

if the X_j's are pairwise independent, since $X_j^2 = 1$ and $E[X_j X_l] = E[X_j]E[X_l] = 0$ by independence. Plugging this into (3) and recalling that $A_{ij} \in \{-\frac{1}{\sqrt{k}}, +\frac{1}{\sqrt{k}}\}$, we get

$$E[Z] = \sum_{i=1}^{k} \frac{1}{k} \sum_j x_j^2 = \|x\|_2^2.$$

Just what we like! To show that Z is indeed close to its mean, we will use Chebyshev, and this requires us to compute the variance of Z.

If the rows of A are independent, then $\mathrm{Var}(Z)$ is the sum of the variances from each row, which in terms of the variable Y defined above is:

$$\mathrm{Var}(Z) = \sum_{i=1}^{k} \frac{1}{k^2} \mathrm{Var}(Y) = \frac{\mathrm{Var}(Y)}{k}.$$

But $\mathrm{Var}(Y) = E[Y^2] - E[Y]^2$, we know what $E[Y]^2$ is. For the other term,

$$\begin{aligned} E[Y^2] &= E[\sum_{p,q,r,s} X_p X_q X_r X_s x_p x_q x_r x_s] \\ &= \sum_p E[X_p^4 x_p^4] + 6 \sum_{p<q} E[X_p^2 X_q^2 x_p^2 x_q^2] + \text{ other terms} \\ &= \sum_p x_p^4 + 6 \sum_{p<q} x_p^2 x_q^2 \end{aligned}$$

(The other terms disappear because of 4-wise independence.) And plugging this into the definition of $\mathrm{Var}(Y)$, we get

$$\mathrm{Var}(Y) = \sum_p x_p^4 + 6 \sum_{p<q} x_p^2 x_q^2 - (\sum_p x_p^2)^2 = 4 \sum_{p<q} x_p^2 x_q^2 \leq 2E[Z]^2.$$

Interesting, the variance Y is just twice the squared mean—that's good, since the variance of Z (which was the final answer, obtained by taking the average of k such variables) is $1/k$ as much, since averaging reduces the variance. So $\mathrm{Var}Z \leq \frac{2}{k} E[Z]^2$. And finally, we can set $k = \frac{2}{\varepsilon^2 \delta}$ and use Chebyshev to get

$$\Pr[Z \notin (1 \pm \varepsilon) E[Z]] \leq \frac{\mathrm{Var}(Z)}{(\varepsilon E[Z])^2} \leq \delta.$$

Great! So, if we take a $k \times D$ matrix A whose $k = \frac{2}{\varepsilon^2 \delta}$ rows were independent, each row having $\{-1, +1\}$ values drawn from a 4-wise independent sample space. We maintain a k-dimensional

vector y, and whenever an element j in $[D]$ comes by in the stream, we just add in the j^{th} column of A to y. And when we want the answer, we reply with $\frac{1}{\sqrt{k}}\|y\|$—this will be correct with probability at least $1-\delta$.

Why 4-wise independence? Well, the calculation of $\mathrm{Var}(Z)$ only used the fact that any four entries of each row behaved independently of each other. And it is possible to generate D values from $\{-1,+1\}$ which is 4-wise independent, using hash functions that require only $O(\log D)$ bits of space. (We'll talk more about this later in the course.) So the total space usage is: $O(k \log D)$ bits to store the hash functions, $O(k \log(LD)) = O(\varepsilon^{-2} \log(LD))$ to store vector y if the frequency of each element is at most L, and that's it.

Citations: This scheme is due to the Gödel prize winning paper of Noga Alon, Yossi Matias, and Mario Szegedy. There has been a lot of interesting work on moment estimation: see, e.g., this STOC 2011 paper of Daniel Kane, Jelani Nelson, Ely Porat and David Woodruff on getting lower bounds for ℓ_p-norms of the vector x, and the many references therein.

4 Subgaussian Behavior

In the previous section, we saw that if each row of the matrix A was drawn from a 4-wise independent sample space (and hence generating any column of A could be done in $O(k \log n)$ space), setting $k = O(\varepsilon^{-2}\delta^{-1})$ would suffice to give answers within $(1 \pm \varepsilon)$ with probability at least $1-\delta$. Note that the number of rows went from $O(\varepsilon^{-2} \log \delta^{-1})$ to $O(\varepsilon^{-2}\delta^{-1})$; this increase typical of cases where we only use the second moment (and limited independence) instead of all the moments (complete independence).

So suppose we did have the luxury of full independence, could we match the JL bound using Rademacher matrices? Or does moving to the $\{-1,1\}$ case already lose something in the performance? It turns out we can also prove Lemma 2 for a Rademacher matrix, losing only constants—we'll now prove this.

Let's look over the proof in Section 1.4, and see what we need to do. We take an arbitrary unit vector x, and define

$$Y \sim \langle R_1, R_2, \ldots, R_d \rangle \cdot x = \sum_i x_i R_i \qquad \text{and} \qquad Z = \frac{1}{k}\sum_{i=1}^{k} Y_i^2$$

for $R_i \in_R \{-1,1\}$, and Y_i's being i.i.d and $Y_i \sim Y$. If we could show that

- $E[Z] = \|x\|_2^2$, and
- $E[e^{tY^2}] \leq \frac{1}{\sqrt{1-ct}}$ for some constant c,

then the rest of the proof of Section 1.4 does not use any other facts about Gaussians. And the first fact $E[Z] = \|x\|_2^2$ follows by the calculations from the previous section, so all we need to do is to bound the moment generating function for Y^2!

We can do this by explicit calculations, but instead let's give a useful abstraction:

Definition 4 *A random variable V is said to be subgaussian with parameter c and for all real s, we have $E[e^{sV}] \leq e^{cs^2}$.*

(You can define subgaussian-ness alternatively as in these notes by Roman Vershynin, which also shows the two definitions are equivalent for symmetric distributions.) A simple calculation shows that for $G \sim N(0,1)$ then $E[e^{sG}] = e^{s^2/2}$—good to know that the Gaussian is also subgaussian!

The following lemma gives a slick way to bound the mgf for the square of a subgaussian, now that we've done the hard work for the Gaussians.

Lemma 5 *If V is subgaussian with parameter c, then $E[e^{sV^2}] \leq \frac{1}{\sqrt{1-4c\,s}}$ for $s > 0$.*

PROOF: Well, suppose $G \sim N(0,1)$ is an independent Gaussian, then

$$E_V[e^{sV^2}] = E_{G,V}[e^{\sqrt{2s}VG}]$$

by the calculation we just did for Gaussians. (Note that we've just introduced a Gaussian into the mix, without any provocation! But it will all work out.) Let just rewrite that

$$E_{G,V}[e^{\sqrt{2s}VG}] = E_G[E_V[e^{(\sqrt{2s}G)V}]].$$

Using the c-subgaussian behavior of V we bound this by

$$E_G[e^{c(\sqrt{2s}|G|)^2}] = E_G[e^{2csG^2}].$$

Finally, the calculation (2) gives this to be $\frac{1}{\sqrt{1-4cs}}$. □

Good. Now if Y were subgaussian, we'd be done. We know that Y is a weighted sum of Rademacher varaibles. A Rademacher random variable is indeed 1/2-subgaussian

$$E[e^{sR}] = \frac{e^s + e^{-s}}{2} = \cosh s = 1 + \frac{s^2}{2!} + \frac{s^4}{4!} + \cdots \leq e^{s^2/2}.$$

And if V_i's are independent and c-subgaussian, and $\|x\|_2 = 1$, then $V = \sum_i x_i V_i$ has

$$E[e^{sV}] = E[e^{\sum_i (sx_i)V_i}] \leq \prod_i e^{c(sx_i)^2} = e^{cs^2 \sum_i x_i^2} = e^{cs^2}.$$

To summarize: R_i's are 1/2-subgaussian, so $Y = \sum_i x_i R_i$ is too. And hence $E[e^{tY^2}] \leq \frac{1}{\sqrt{1-2t}}$ for $\{-1,+1\}$-random variables as well. This, in turn, completes the proof that the Rademacher matrix also has the JL property! Note that the JL matrix A now just requires us to pick $kD = O(D\varepsilon^{-2}\log\delta^{-1})$ random bits (instead of kD random Gaussians); also, there are fewer precision issues to worry about. One can consider other distributions to stick into the matrix A—all you need to show is that Z has the right mean, and that the entries are subgaussian.

Citations: The scheme of using Rademacher matrices instead of Gaussian matrices for JL was first proposed in this paper by Dimitris Achlioptas. The idea of extending it to subgaussian distributions appears in this paper of Indyk and Naor, and this paper of Matousek. The paper of Klartag and Mendelson generalizes this even further.

BTW, one can define subgaussian distributions as ones that satisfy $E[e^{sV}] \leq e^{cs^2}$ only for $c > 0$, or as variables for which $\Pr[V \geq \lambda] \leq e^{-c\lambda^2}$ for $\lambda > 0$ (the upper tail is subgaussian), and prove JL bounds—see, e.g., the paper of Matousek—but it does not matter for distributions symmetric about 0 with bounded variance, since these definitions are then essentially the same.

Fast J-L: Do we really need to plug in non-zero values into every entry of the matrix A? What if most of A is filled with zeroes? The first problem is that if x is a very sparse vector, then Ax might be zero with high probability? Achlioptas showed that having a random two-thirds of the entries of A being zero still works fine: the paper of Nir Ailon and Bernard Chazelle showed that if you first hit x with a suitable matrix P which caused Px to be "well-spread-out" whp, and then $\|APx\| \approx \|x\|$ would still hold for a much sparser A. Moreover, this P requires much less randomless, and furthermore, the computations can be done faster too! There has been much work on fast and sparse versions of JL: see, e.g., this SODA 11 paper of Ailon and Edo Liberty, and this arxiv preprint by Daniel Kane and Jelani Nelson. Jelani has some notes on the Fast JL Transform.

Compressed Sensing: Finally, the J-L lemma is closely related to compressed sensing: how to reconstruct a sparse signal using very few measurements. See these notes by Jiri Matousek, or these by Baraniuk and others for a proof of the beautiful connection. I will say more about this connection in a later post.