Fast Random Projections

Edo Liberty

September 18, 2007

---

1Yale University, New Haven CT, supported by AFOSR and NGA (www.edoliberty.com) Advised by Steven Zucker.
This talk will survey a few random projection algorithms, From the classic result by W.B. Johnson and J. Lindenstrauss (1984) to a recent faster variant of the FJLT algorithm [2] which was joint work with Nir Ailon (Google research). Many thanks also to Mark Tygert and Tali Kaufman.

Since some of the participants are unfamiliar with the classic results, I will also show these, later this week, for those who are interested.
We look for a mapping \( f \) from dimension \( d \) to dimension \( k \) such that \(| \|u_i - u_j\| - \|f(u_i) - f(u_j)\| | < \epsilon \). And \( k \) is much smaller than \( d \).

This idea is critical in many algorithms such as:

- Approximate nearest neighbors searches
- Rank \( k \) approximation
- Compressed sensing

and the list continues...
Random Projections introduction

More precisely:

**Lemma (Johnson, Lindenstrauss (1984) [3])**

For any set of $n$ points $u_1 \ldots u_n$ in $\mathbb{R}^d$ there exists a linear mapping $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ such that all pairwise distances are preserved up to distortion $\epsilon$

$$\forall i, j \ (1 - \epsilon) \|u_i - u_j\|^2 \leq \|f(u_i) - f(u_j)\|^2 \leq (1 + \epsilon) \|u_i - u_j\|^2$$

if

$$k > \frac{9 \ln n}{\epsilon^2 - \epsilon^3}$$
All random projection algorithms have the same basic idea:

1. Set $f(x) = Ax$ and $A \in \mathbb{R}^{k \times d}$.
2. Choose $A$ from a probability distribution such that each distance $\|u_i - u_j\|$ is preserved with very high probability.
3. Union bound on the failure probabilities of all $\binom{n}{2}$ distances.
4. Choose $k$ such that the failure probability is constant.
Random Projections introduction

Similar to a definition given by Matousek,

**Definition**

A distribution $D(d, k)$ on $k \times d$ real matrices ($k \leq d$) has the Johnson-Lindenstrauss property (JLP) if for any unit vector $x \in \ell_2^d$ and $0 \leq \epsilon < 1/2$,

$$\Pr_{A \sim D_d,k} \left[ 1 - \epsilon \leq \|Ax\| \leq 1 + \epsilon \right] \geq 1 - c_1 e^{-c_2 k \epsilon^2} \quad (1)$$

for some global $c_1, c_2 > 0$.

A union bound on $\binom{n}{2}$ distance vectors ($x = u_i - u_j$) gives a constant success probability for $k = O\left(\frac{\log(n)}{\epsilon^2}\right)$

Proving the existence of a length preserving mapping reduces to finding distributions with the JLP.
Classic constructions

Classic distributions that exhibit the JLP.

- The original proof and construction, W.B. Johnson and J. Lindenstrauss (1984). They used $k$ rows from random orthogonal matrix (random projection matrix).

- P. Indyk and R. Motowani (1998) use a random Gaussian distribution, $A(i, j) \sim N(0, 1)$. Although it is conceptually not different from previous results it is significantly easier to prove due to the rotational invariance of the normal distribution.

- Dimitris Achlioptas (2003) showed that a dense $A(i, j) \in \{0, -1, 1\}$ matrix also exhibits the JLP.

Some other JLP distributions and proofs:

- P. Frankl and H. Meahara (1987)
- S. DasGupta and A. Gupta (1999)
Let’s think about applications

The amount of space needed is $O(dk)$ and the time to apply the mapping to any vector takes $O(dk)$ operations.

Try to apply the mapping to a 5Mp image, and project it down to $10^4$ coordinates, that is roughly a 10G matrix! (somewhat unpleasant)
(In some situations one can generate and forget $A$ on the fly and thereby reducing the space constraint.)

Can we save on time and space by making $A$ sparse?
Can $A$ be sparse?

The short answer is no. Let $\mathbf{x}$ contains only 1 non zero entry, say $i$, then:

$$\|A\mathbf{x}\| = \|A(i)\|$$

We need each column’s norm to concentrate around 1 with deviation $k^{-1/2}$. It therefore must contain at least $O(k)$ entries.
Fast Johnson Lindenstrauss Transform

Maybe we should first make $x$ dense?

One way to achieve that is to first map $x \mapsto HDx$ and then use a sparse matrix $P$ to project it.

**Lemma (Ailon, Chazelle (2006) [1])**

- Let $P \in \mathbb{R}^{k \times d}$ be a sparse matrix. Let $q = \Theta\left(\frac{\log^2(n)}{d}\right)$, set $P(i, j) \sim N(0, q^{-1})$ w.p $q$ and $P(i, j) = 0$ else.
- Let $H$ denote the $d \times d$ Walsh Hadamard matrix.
- and let $D$ denote a $d \times d$ diagonal random $\pm 1$ matrix.

The matrix $A = PHD$ exhibits the JLP.

Notice that $P$ contains only $O(k^3)$ entrees (in expectancy) which is much less then $kd$. 
What did we gain?

- Time to apply the matrix $A$ to a vector is now reduced to $O(d \log(d) + k^3)$ which is much less than $dk$.
- The space needed for storing $A$ is $d + k^3 \log(d)$. (during application one needs $d \log(d) + k^3 \log(d)$ space).
- We also save on randomness, constructing $A$ requires $O(d + k^3 \log(d))$ random bits. (Vs. $O(dk)$ for classic constructions)
Can we do any better?

1. We are computing $d$ coefficients of the Walsh Hadamard matrix although we use at most $k^3$ of them. Can we effectively reduce computation?
2. Where does the $k^3$ term come from? can we reduce it?
3. Can we save on randomness?

Answers:
1. Yes. We can reduce $d \log(d)$ to $d \log(k)$.
2. Yes. We can eliminate the $k^3$ term.
3. Yes. We can derandomize $P$ all together.

Unfortunately, we only know how to do this for $k = O(d^{1/2-\delta})$ for some arbitrarily small delta.
Faster JL Transform

Theorem (Ailon, Liberty (2007) [2])

Let $\delta > 0$ be some arbitrarily small constant. For any $d, k$ satisfying $k \leq d^{1/2-\delta}$ there exists an algorithm constructing a random matrix $A$ of size $k \times d$ satisfying JLP, such that the time to compute $x \mapsto Ax$ for any $x \in \mathbb{R}^d$ is $O(d \log k)$. The construction uses $O(d)$ random bits and applies to both the Euclidean and the Manhattan cases.

<table>
<thead>
<tr>
<th>$k$ in</th>
<th>$k$ in</th>
<th>$k$ in</th>
<th>$k$ in</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o(\log d)$</td>
<td>$\omega(\log d)$ and $o(\text{poly}(d))$</td>
<td>$\Omega(\text{poly}(d))$ and $o((d \log d)^{1/3})$</td>
<td>$\omega((d \log d)^{1/3})$ and $O(d^{1/2-\delta})$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fast</th>
<th>This work</th>
<th>This work</th>
<th>This work, FJLT</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>JL</td>
<td>FJLT</td>
<td>JL</td>
<td>JL</td>
<td>JL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Slow</th>
<th>FJLT</th>
<th>JL</th>
<th>JL</th>
</tr>
</thead>
</table>

This work, FJLT
Answer for the first question, can we compute only the coefficients that we need from the transform?

The Hadamard matrix has a recursive structure as such:

\[
H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_d = \begin{pmatrix} H_{d/2} & H_{d/2} \\ H_{d/2} & -H_{d/2} \end{pmatrix}
\] (2)

Let us look at the product \( PHDx \), let \( z = Dx \) and let \( z_1 \) and \( z_2 \) be the first and second half of \( z \), also \( P_1 \) and \( P_2 \) are the left and right halves of \( P \). Assume that \( |P| = k \)
Trimming the Hadamard transform

\[ PH_q z = \begin{pmatrix} P_1 & P_2 \end{pmatrix} \begin{pmatrix} H_{q/2} & H_{q/2} \\ H_{q/2} & -H_{q/2} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \]

\[ = P_1 H_{q/2}(z_1 + z_2) + P_2 H_{q/2}(z_1 - z_2) \]

Which gives the relation \( T(d, k) = T(d/2, k_1) + T(d/2, k_2) + d \).

We use induction to show that \( T(d, k) \leq 2d \log(k + 1) \), \( T(d, 1) = d \).

\[ T(d, k) = T(d/2, k_1) + T(d/2, k_2) + d \]
\[ \leq d \log(2(k_1 + 1)(k_2 + 1)) \]
\[ \leq d \log((k_1 + k_2 + 1)^2) \text{ for any } k_1 + k_2 = k \geq 1 \]
\[ \leq 2d \log(k + 1) \]

Finally \( T(d, k) = O(d \log(k)) \).
Modifying the FJLT algorithm

Notice that by applying the trimmed Walsh Hadamard transform one can use the FJLT algorithm as is with running time $O(d \log(k) + k^3)$ which is $O(d \log(k))$ for any $k = O(d^{1/3})$.

We move to deal with a harder problem which is to construct an algorithm that holds up to $k = O(d^{1/2 - \delta})$. 
Rademacher random variables

Answer for the second question, where does $k^3$ come from?

The hardest vectors to project correctly are sparse ones. Ailon and Chazelle bound $\|HDx\|_\infty$ and then project the sparsest $z$ such vectors. $z(i) \in \{0, \|HDx\|_\infty\}$, Intuitively these are actually very rare.

Let’s try to bound $\|PHDx\|_2$ directly.
Rademacher random variables

- Let $M$ be a real $m \times d$ matrix,
- Let $z$ be a random vector $z \in \{-1, 1\}^d$
- $Mz \in \ell^m_2$ is known as a Rademacher random variable.
- $Z = \|Mz\|_2$ is the norm of a Rademacher random variable in $\ell^d_2$ corresponding to $M$

We associate two numbers with $Z$,
- The deviation $\sigma$, defined as $\|M\|_{2\to2}$, and
- a median $\mu$ of $Z$.

Theorem (Ledoux and Talagrand (1991))

For any $t \geq 0$, $\Pr[|Z - \mu| > t] \leq 4e^{-t^2/(8\sigma^2)}$. 

Rademacher random variables

We write $PHDx$ as $PHXz$ where $X$ is $\text{diag}(x)$ and $z$ is a random $\pm 1$, and recall the JLP definition:

$$\Pr[|\|Mz\| - \mu| > t] \leq 4e^{-t^2/(8\sigma^2)}$$

$$\Pr[|\|PHXz\| - 1| \geq \epsilon] \leq c_1 e^{-c_2 k\epsilon^2}$$

To show that $PHD$ has the JLP we need only show that:

1. $\sigma = \|PHX\|_{2\to2} = O(k^{-1/2})$.
2. $|\mu - 1| = O(\sigma)$.

Notice that $P$ does not need to be random any more! From this point on we replace $PH$ with $B \in \mathbb{R}^{k \times d}$, We will choose $B$ later.
Bounding $\sigma$

Reminder $M = BDX$ and $\sigma = \|M\|_{2 \rightarrow 2}$.

$$\sigma = \|M\|_{2 \rightarrow 2} = \sup_{\substack{y \in \ell^k_2 \\ \|y\| = 1}} \|y^T M\|_2$$

$$= \sup \left( \sum_{i=1}^{d} x_i^2 (y^T B^{(i)})^2 \right)^{1/2}$$

$$\leq \|x\|_4 \sup \left( \sum_{i=1}^{d} (y^T B^{(i)})^4 \right)^{1/4}$$

$$= \|x\|_4 \|B^T\|_{2 \rightarrow 4}.$$
Choosing $B$

**Definition**
A matrix $A(i, j) \in \{+k^{-1/2}, -k^{-1/2}\}$ of size $k \times d$ is 4-wise independent if for each $1 \leq i_1 < i_2 < i_3 < i_4 \leq k$ and $(b_1, b_2, b_3, b_4) \in \{+1, -1\}^4$, the number of columns $A(j)$ for which $(A^{(j)}_{i_1}, A^{(j)}_{i_2}, A^{(j)}_{i_3}, A^{(j)}_{i_4}) = k^{-1/2}(b_1, b_2, b_3, b_4)$ is exactly $d/2^4$.

**Lemma**
There exists a 4-wise independent matrix $A$ of size $k \times d_{bch}$, $d_{bch} = \Theta(k^2)$, such that $A$ consists of $k$ rows of $H_d$. We take $B$ to be $\lceil d/d_{bch} \rceil$ copies of $A$ side by side. Clearly $B$ is still 4-wise independent. $^2$

---

$^2$The family of matrices is known as binary dual BCH codes of designed distance 5. Under the usual transformation $(+)$ $\to$ 0, $(-)$ $\to$ 1 (and normalized).
Bounding $\|B\|_{2\rightarrow 4}$

**Lemma**
Assume $B$ is a $k \times d$ 4-wise independent code matrix. Then $\|B^T\|_{2\rightarrow 4} \leq cd^{1/4}k^{-1/2}$.

**Proof.**
For $y \in \ell^k_2$, $\|y\| = 1$,

$$\|y^T B\|_4^4 = dE_{j \in [d]}[(y^T B(j))^4]$$

$$= dk^{-2} \sum_{i_1, i_2, i_3, i_4 = 1}^k E_{b_1, b_2, b_3, b_4} [y_{i_1} y_{i_2} y_{i_3} y_{i_4} b_1 b_2 b_3 b_4] \quad (3)$$

$$= dk^{-2}(3\|y\|_2^4 - 2\|y\|_4^4) \leq 3dk^{-2}.$$
Reducing $\|x\|_4$

Reminder: we need $\sigma \leq \|x\|_4 \|B^T\|_{2\rightarrow 4} = O(k^{-1/2})$,

We already have that $\|B^T\|_{2\rightarrow 4} \leq cd^{1/4}k^{-1/2}$.

The objective is to get $\|x\|_4 = O(d^{-1/4})$ But $x$ is given to us and $\|x\|_4$ might be 1.

The solution is to map $x \mapsto \Phi x$ where $\Phi$ is a randomized isometry. Such that with high probability $\|\Phi x\|_4 = O(d^{1/4})$. 
Reducing \( \|x\|_4 \)

The idea is to compose \( r \) Walsh Hadamard matrices with different random diagonal matrices.

**Lemma**

[\( \ell_4 \) reduction for \( k < d^{1/2-\delta} \)] Let \( \Phi = HD_r \cdots HD_2 HD_1 \), with probability \( 1 - O(e^{-k}) \)

\[
\|\Phi^{(r)} x\|_4 = O(d^{-1/4})
\]

for \( r = \lceil 1/2\delta \rceil \).

Note that the constant hiding in the bound (9) is exponential in \( 1/\delta \).
Putting it all together

We have that $\|\Phi^{(r)} x\|_4 = O(d^{-1/4})$ and $\|B^T\|_{2\rightarrow 4} = O(d^{1/4} k^{-1/2})$ and so we gain $\sigma = O(k^{-1/2})$, finally

**Lemma**

*The matrix $A = BD\Phi$ exhibits the JLP.*
But what about the running time?

Notice that applying $\Phi$ takes $O(d \log(d))$ time. Which is bad if $d \gg k^2$.

Remember that $B$ is built out of many copies of the original $k \times d_{BCH}$ code matrix ($d_{BCH} = \Theta(k^2)$). It turns out that $\Phi$ can also be constructed of blocks of size $d_{BCH} \times d_{BCH}$ and $\Phi$ can also be applied in $O(d \log(k))$.
Let $\delta > 0$ be some arbitrarily small constant. For any $d, k$ satisfying $k \leq d^{1/2-\delta}$ there exists an algorithm constructing a random matrix $A$ of size $k \times d$ satisfying JLP, such that the time to compute $x \mapsto Ax$ for any $x \in \mathbb{R}^d$ is $O(d \log k)$. The construction uses $O(d)$ random bits and applies to both the Euclidean and the Manhattan cases.

| $k$ in $o(\log d)$ and $o(\text{poly}(d))$ | $k$ in $\omega(\log d)$ and $\omega(\text{poly}(d))$ | $k$ in $\Omega(\text{poly}(d))$ and $o((d \log d)^{1/3})$ | $k$ in $\omega((d \log d)^{1/3})$ and $O(d^{1/2-\delta})$
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>this work</td>
<td>this work</td>
<td>this work, FJLT</td>
<td>this work</td>
</tr>
<tr>
<td>fast</td>
<td>this work, FJLT</td>
<td>fast</td>
<td>fast</td>
</tr>
<tr>
<td>slow</td>
<td>FJLT</td>
<td>JL</td>
<td>JL</td>
</tr>
<tr>
<td>slow</td>
<td>FJLT</td>
<td>JL</td>
<td>JL</td>
</tr>
</tbody>
</table>
Future work

- Going beyond $k = d^{1/2-\delta}$. As part of our work in progress, we are trying to push the result to higher values of the target dimension $k$ (the goal is a running time of $O(d \log d)$). We conjecture that this is possible for $k = d^{1-\delta}$, and have partial results in this direction. A more ambitious goal is $k = \Omega(d)$.

- Lower bounds. A lower bound on the running time of applying a random matrix with a JL property on a vector will be extremely interesting. Any nontrivial (superlinear) bound for the case $k = d^{\Omega(1)}$ will imply a lower bound on the time to compute the Fourier transform, because the bottleneck of our constructions is a Fourier transform.

- If there is no lower bound, can we devise a linear time JL projection? This will of course be very interesting, it seems that this might be possible for very large values of $d$ relative to $n$. 
Thank you for listening
Nir Ailon and Bernard Chazelle.
Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform.

Nir Ailon and Edo Liberty.
Fast dimension reduction using rademacher series on dual bch codes.
In *Symposium on Discrete Algorithms (SODA), accepted*, 2008.

W. B. Johnson and J. Lindenstrauss.
Extensions of Lipschitz mappings into a Hilbert space.

J. Matousek.
On variants of the Johnson-Lindenstrauss lemma.
*Private communication*, 2006.
\(|\mu - 1| = O(\sigma)\)

Reminder:

- \(Z\) is our random variable \(Z = \|Mz\|_2\).
- \(E(Z^2) = 1\).
- \(\Pr[|Z - \mu| > t] \leq 4e^{-t^2/(8\sigma^2)}\)

Let us bound \(|1 - \mu|\)

\[
E[(Z - \mu)^2] = \int_0^\infty \Pr[(Z - \mu)^2] > s] ds \\
\leq \int_0^\infty 4e^{-s/(8\sigma^2)} ds = 32\sigma^2
\]

\[
E[Z] = E[\sqrt{Z^2}] \leq \sqrt{E[Z^2]} = 1 \text{ (by Jensen)}
\]

\[
E[(Z - \mu)^2] = E[Z^2] - 2\mu E[Z] + \mu^2 \geq 1 - 2\mu + \mu^2 = (1 - \mu)^2
\]

\[
|1 - \mu| \leq \sqrt{32}\sigma
\]