Random Projections for Machine Learning and Data Mining: Theory and Applications

Robert J. Durrant & Ata Kabán

University of Birmingham

{r.j.durrant, a.kaban}@cs.bham.ac.uk

www.cs.bham.ac.uk/~durranrj

sites.google.com/site/rpforml

ECML-PKDD 2012, Friday 28th September 2012

Outline

Outline

Motivation - Dimensionality Curse

The 'curse of dimensionality': A collection of pervasive, and often counterintuitive, issues associated with working with high-dimensional data.
Two typical problems:

- Very high dimensional data (arity $\in \mathcal{O}(1000)$) and very many observations ($N \in \mathcal{O}(1000)$): Computational (time and space complexity) issues.
- Very high dimensional data (arity $\in \mathcal{O}(1000)$) and hardly any observations ($N \in \mathcal{O}(10)$): Inference a hard problem. Bogus interactions between features.

Curse of Dimensionality

Comment:
What constitutes high-dimensional depends on the problem setting, but data vectors with arity in the thousands very common in practice (e.g. medical images, gene activation arrays, text, time series, ...).

Issues can start to show up when data arity in the tens!

We will simply say that the observations, $\mathcal{T}$, are d-dimensional and there are N of them: $\mathcal{T} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^N$ and we will assume that, for whatever reason, d is too large.

Mitigating the Curse of Dimensionality

An obvious solution: Dimensionality d is too large, so reduce d to $k \ll d$.

How?
Dozens of methods: PCA, Factor Analysis, Projection Pursuit, Random Projection ...

We will be focusing on Random Projection, motivated (at first) by the following important result:

Outline

Johnson-Lindenstrauss Lemma

The JLL is the following rather surprising fact [DG02, Ach03]:

Theorem (Johnson and Lindenstrauss, 1984)

Let $\epsilon \in (0,1)$. Let $N, k \in \mathbb{N}$ such that $k \geqslant C\epsilon^{-2}\log N$, for a large enough absolute constant C. Let $V \subseteq \mathbb{R}^d$ be a set of N points. Then there exists a linear mapping $R : \mathbb{R}^d \to \mathbb{R}^k$, such that for all $u, v \in V$:

$$(1-\epsilon)\|u-v\|^2 \leqslant \|Ru - Rv\|^2 \leqslant (1+\epsilon)\|u-v\|^2$$

- Dot products are also approximately preserved by R since if JLL holds then: $u^T v - \epsilon \leqslant (Ru)^T Rv \leqslant u^T v + \epsilon$. (Proof: parallelogram law - see appendix).
- Scale of k is essentially sharp: $\forall N, \exists V$ s.t. $k \in \Omega(\epsilon^{-2}\log N / \log \epsilon^{-1})$ is required [Alo03].
- We shall prove shortly that with high probability *random projection* implements a suitable linear R.

Intuition

Geometry of data gets perturbed by random projection, but not too much:

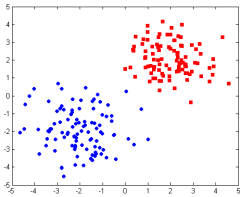


Figure: Original data

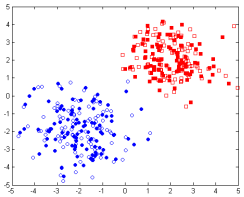


Figure: RP data & Original data

Applications

Random projections have been used for:

- Classification. e.g. [BM01, FM03, GBN05, SR09, CJS09, RR08, DK12b]
- Regression. e.g. [MM09, HWB07, BD09]
- Clustering and Density estimation. e.g. [IM98, AC06, FB03, Das99, KMV12, AV09]
- Other related applications: structure-adaptive kd-trees [DF08], low-rank matrix approximation [Rec11, Sar06], sparse signal reconstruction (compressed sensing) [Don06, CT06], data stream computations [AMS96].

Intuition

Geometry of data gets perturbed by random projection, but not too much:

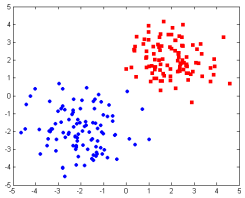


Figure: Original data

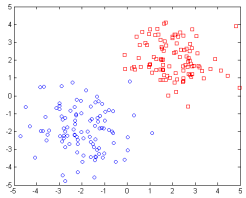


Figure: RP data (schematic)

What is Random Projection? (1)

Canonical RP:

- Construct a (wide, flat) matrix $R \in \mathcal{M}_{k\times d}$ by picking the entries from a univariate Gaussian $\mathcal{N}(0, \sigma^2)$.
- Orthonormalize the rows of R, e.g. set $R' = (RR^T)^{-1/2}R$.
- To project a point $v \in \mathbb{R}^d$, pre-multiply the vector v with RP matrix R'. Then $v \mapsto R'v \in R'(\mathbb{R}^d) \equiv \mathbb{R}^k$ is the projection of the d-dimensional data into a random k-dimensional projection space.

Comment (1)

If d is very large we can drop the orthonormalization in practice - the rows of R will be nearly orthogonal to each other and all nearly the same length.

For example, for Gaussian ($\mathcal{N}(0, \sigma^2)$) R we have [DK12a]:

$$\Pr\left\{(1-\epsilon)d\sigma^2 \leqslant \|R_i\|_2^2 \leqslant (1+\epsilon)d\sigma^2\right\} \geqslant 1-\delta,\ \forall \epsilon \in (0,1]$$

where R_i denotes the i-th row of R and
$\delta = \exp(-(\sqrt{1+\epsilon}-1)^2 d/2) + \exp(-(\sqrt{1-\epsilon}-1)^2 d/2)$.

Similarly [Led01]:

$$\Pr\{|R_i^T R_j|/d\sigma^2 \leqslant \epsilon\} \geqslant 1 - 2\exp(-\epsilon^2 d/2),\ \forall i \neq j.$$

Concentration in norms of rows of R

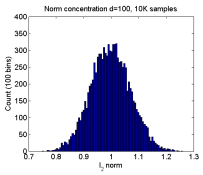


Figure: $d = 100$ norm concentration

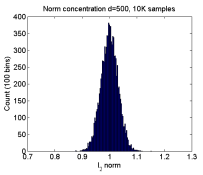


Figure: $d = 500$ norm concentration

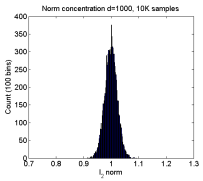


Figure: $d = 1000$ norm concentration

Near-orthogonality of rows of R

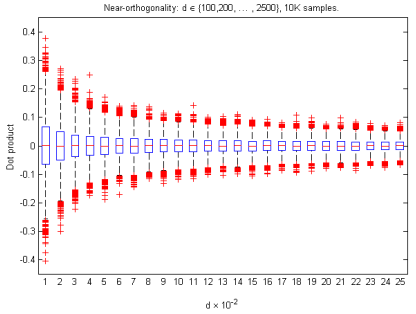


Figure: Normalized dot product is concentrated about zero, $d \in \{100, 200, \ldots, 2500\}$

Why Random Projection?

- Linear.
- Cheap.
- Universal – JLL holds w.h.p for any fixed finite point set.
- Oblivious to data distribution.
- Target dimension doesn't depend on data dimensionality (for JLL).
- Interpretable - approximates an isometry (when d is large).
- Tractable to analysis.

Jargon

'With high probability' (w.h.p) means with a probability as close to 1 as we choose to make it.

'Almost surely' (a.s.) or 'with probability 1' (w.p. 1) means so likely we can pretend it always happens.

'With probability 0' (w.p. 0) means so unlikely we can pretend it never happens.

Proof of JLL (1)

We will prove the following randomized version of the JLL, and then show that this implies the original theorem:

Theorem

Let $\epsilon \in (0,1)$. Let $k \in \mathbb{N}$ such that $k \geqslant C\epsilon^{-2}\log\delta^{-1}$, for a large enough absolute constant C. Then there is a random linear mapping $P : \mathbb{R}^d \to \mathbb{R}^k$, such that for any unit vector $x \in \mathbb{R}^d$:

$$Pr\left\{(1-\epsilon) \leqslant \|Px\|^2 \leqslant (1+\epsilon)\right\} \geqslant 1-\delta$$

- No loss to take $\|x\| = 1$, since P is linear.
- Note that this mapping is universal and the projected dimension k depends only on ϵ and δ.
- Lower bound [JW11, KMN11] $k \in \Omega(\epsilon^{-2}\log\delta^{-1})$

Proof of JLL (2)

Consider the following simple mapping:

$$Px := \frac{1}{\sqrt{k}} Rx$$

where $R \in \mathcal{M}_{k \times d}$ with entries $R_{ij} \overset{i,i,d}{\sim} \mathcal{N}(0,1)$.

Let $x \in \mathbb{R}^d$ be an arbitrary unit vector.
We are interested in quantifying:

$$\|Px\|^2 = \left\| \frac{1}{\sqrt{k}} Rx \right\|^2 := \left\| \frac{1}{\sqrt{k}} (Y_1, Y_2, \ldots, Y_k) \right\|^2 = \frac{1}{k} \sum_{i=1}^{k} Y_i^2 =: Z$$

where $Y_i = \sum_{j=1}^{d} R_{ij} x_j$.

Proof of JLL (3)

Recall that if $W_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ and the W_i are independent, then $\sum_i W_i \sim \mathcal{N}\left(\sum_i \mu_i, \sum_i \sigma_i^2\right)$. Hence, in our setting, we have:

$$Y_i = \sum_{j=1}^{d} R_{ij} x_j \sim \mathcal{N}\left(\sum_{j=1}^{d} \mathrm{E}[R_{ij} x_j], \sum_{j=1}^{d} \mathrm{Var}(R_{ij} x_j) \right) \equiv \mathcal{N}\left(0, \sum_{j=1}^{d} x_j^2\right)$$

and since $\|x\|^2 = \sum_{j=1}^{d} x_j^2 = 1$ we therefore have:

$$Y_i \sim \mathcal{N}(0,1), \ \forall i \in \{1, 2, \ldots, k\}$$

it follows that each of the Y_i are standard normal RVs and therefore $kZ = \sum_{i=1}^{k} Y_i^2$ is χ_k^2 distributed.
Now we complete the proof using a standard Chernoff-bounding approach.

Proof of JLL (4)

$$\begin{aligned}
\Pr\{Z > 1+\epsilon\} &= \Pr\{\exp(tkZ) > \exp(tk(1+\epsilon))\}, \ \forall t > 0 \\
\text{Markov ineq.} &\leqslant \mathrm{E}\left[\exp(tkZ)\right] / \exp(tk(1+\epsilon)), \\
Y_i \text{ indep.} &= \prod_{i=1}^{k} \mathrm{E}\left[\exp(tY_i^2)\right] / \exp(tk(1+\epsilon)), \\
\text{mgf of } \chi^2 &= \left[\exp(t)\sqrt{1-2t}\right]^{-k} \exp(-kt\epsilon), \forall t < 1/2 \\
\text{next slide} &\leqslant \exp\left(kt^2/(1-2t) - kt\epsilon\right), \\
&\leqslant e^{-\epsilon^2 k/8}, \text{ taking } t = \epsilon/4 < 1/2.
\end{aligned}$$

$\Pr\{Z < 1-\epsilon\} = \Pr\{-Z > \epsilon - 1\}$ is tackled in a similar way and gives same bound. Taking RHS as $\delta/2$ and applying union bound completes the proof (for single x).

Estimating $(e^t\sqrt{1-2t})^{-1}$

$$\begin{aligned}
\left(e^t\sqrt{1-2t}\right)^{-1} &= \exp\left(-t - \frac{1}{2}\log(1-2t)\right), \\
\text{Maclaurin S. for } \log(1-x) &= \exp\left(-t - \frac{1}{2}\left(-2t - \frac{(2t)^2}{2} - \ldots\right)\right), \\
&= \exp\left(\frac{(2t)^2}{4} + \frac{(2t)^3}{6} + \ldots\right), \\
&\leqslant \exp\left(t^2\left(1 + 2t + (2t)^2 \ldots\right)\right), \\
&= \exp\left(t^2/(1-2t)\right) \text{ since } 0 < 2t < 1
\end{aligned}$$

Randomized JLL implies Deterministic JLL

- Solving $\delta = 2\exp(-\epsilon^2 k/8)$ for k we obtain $k = 8/\epsilon^2 \log \delta^{-1} + \log 2$. i.e. $k \in \mathcal{O}\left(\epsilon^{-2} \log \delta^{-1}\right)$.
- Let $V = \{x_1, x_2, \ldots, x_N\}$ an arbitrary set of N points in $\mathbb{R}^d$ and set $\delta = 1/N^2$, then $k \in \mathcal{O}\left(\epsilon^{-2} \log N\right)$.
- Applying union bound to the randomized JLL proof for all $\binom{N}{2}$ possible interpoint distances, for N points we see a random JLL embedding of V into k dimensions succeeds with probability at least $1 - \binom{N}{2}\frac{1}{N^2} > \frac{1}{2}$.
- We succeed with positive probability for arbitrary V. Hence we conclude that, for any set of N points, there exists linear $P : \mathbb{R}^d \to \mathbb{R}^k$ such that:

$$(1-\epsilon)\|x_i - x_j\|^2 \leqslant \|Px_i - Px_j\|^2 \leqslant (1+\epsilon)\|x_i - x_j\|^2$$

which is the (deterministic) JLL.

Comment (2)

In the proof of the randomized JLL the only properties we used which are specific to the Gaussian distribution were:

1. Closure under additivity.
2. Bounding squared Gaussian RV using mgf of χ^2.

In particular, bounding via the mgf of χ^2 gave us exponential concentration about mean norm.
Can do similar for matrices with zero-mean *sub-Gaussian* entries also: Sub-Gaussians are those distributions whose tails decay no slower than a Gaussian, for example all bounded distributions have this property.
Can derive similar guarantees (i.e. up to small multiplicative constants) for sub-Gaussian RP matrices too!
This allows us to get around issue of dense matrix multiplication in dimensionality-reduction step.

What is Random Projection? (2)

Different types of RP matrix easy to construct - take entries i.i.d from *nearly any* zero-mean subgaussian distribution. All behave in much the same way.
Popular variations [Ach03, AC06, Mat08]:
The entries R_{ij} can be:

$$R_{ij} = \begin{cases} +1 & \text{w.p. } 1/2, \\ -1 & \text{w.p. } 1/2. \end{cases} \qquad R_{ij} = \begin{cases} \mathcal{N}(0,1/q) & \text{w.p. } q, \\ 0 & \text{w.p. } 1-q. \end{cases}$$

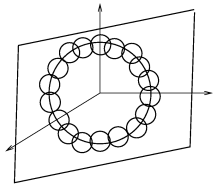
$$R_{ij} = \begin{cases} +1 & \text{w.p. } 1/6, \\ -1 & \text{w.p. } 1/6, \\ 0 & \text{w.p. } 2/3. \end{cases} \qquad R_{ij} = \begin{cases} +1 & \text{w.p. } q, \\ -1 & \text{w.p. } q, \\ 0 & \text{w.p. } 1-2q. \end{cases}$$

For the RH examples, taking q too small gives high distortion of sparse vectors [Mat08]. [AC06] get around this by using a randomized orthogonal (normalized Hadamard) matrix to ensure w.h.p all data vectors are dense.

Subspace JLL

Let S be an s-dimensional linear subspace. Let $\epsilon > 0$. For $k = \mathcal{O}\left(\epsilon^{-2} s \log(12/\epsilon)\right)$ [BW09] w.h.p. a JLL matrix R satisfies $\forall x, y \in S$:
$(1-\epsilon)\|x-y\| \leqslant \|Rx-Ry\| \leqslant (1+\epsilon)\|x-y\|$

1. R linear, so no loss to take $\|x-y\| = 1$.
2. Cover unit sphere in subspace with $\epsilon/4$-balls. Covering number $M = (12/\epsilon)^s$.
3. Apply JLL to centres of the balls. $k = \mathcal{O}(\log M)$ for this.
4. Extend to entire s-dimensional subspace by approximating any unit vector with one of the centres.



Fast, sparse variants

Achlioptas '01 [Ach03]: $R_{ij} = 0$ w.p. $2/3$

Ailon-Chazelle '06 [AC06]: Use $x \longmapsto PHDx$, P random and sparse, $R_{ij} \sim \mathcal{N}(0, 1/q)$ w.p $1/q$, H normalized Hadamard (orthogonal) matrix, $D = \text{diag}(\pm 1)$ random. Mapping takes $\mathcal{O}\left(d \log d + qd\epsilon^{-2} \log N\right)$.

Ailon-Liberty '09 [AL09]: Similar construction to [AC06].
$\mathcal{O}\left(d \log k + k^2\right)$.

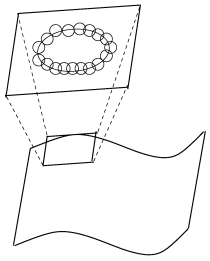
Dasgupta-Kumar-Sarlós '10 [DKS10]: Use sequence of (dependent) random hash functions. $\mathcal{O}\left(\epsilon^{-1} \log^2(k/\delta) \log \delta^{-1}\right)$ for $k \in \mathcal{O}\left(\epsilon^{-2} \log \delta^{-1}\right)$.

Ailon-Liberty '11 [AL11]: Similar construction to [AC06]. $\mathcal{O}(d \log d)$ provided $k \in \mathcal{O}\left(\epsilon^{-2} \log N \log^4 d\right)$.

Manifold JLL

Definition: $M \subset \mathbb{R}^d$ is an s-dimensional manifold if $\forall x \in M$ there is a smooth bijective map between $\mathbb{R}^s$ and a neighbourhood of x.

Let M be an s-dimensional manifold in $\mathbb{R}^d$ with bounded curvature. ($\Rightarrow M$ is locally like a linear subspace.) Let $\epsilon > 0$. For $k = \mathcal{O}\left(s\epsilon^{-2} \log d\right)$ [BW09] w.h.p a JLL matrix R satisfies $\forall x, y \in M$:
$(1-\epsilon)\|x-y\| \leqslant \|Rx-Ry\| \leqslant (1+\epsilon)\|x-y\|$



Proof idea:

1. Approximate manifold with tangent subspaces.
2. Apply subspace-JLL on each subspace.
3. Union bound over subspaces to preserve large distances.

(Same approach can be used to preserve geodesic distances.)

Generalizations of JLL to Manifolds

From JLL we obtain high-probability guarantees that for a suitably large k, independently of the data dimension, random projection approximately preserves data geometry of a finite point set. In particular norms and dot products approximately preserved w.h.p.

JLL approach can be extended to (compact) Riemannian manifolds: 'Manifold JLL'

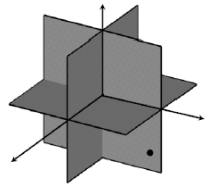
Key idea: Preserve ϵ-covering of smooth manifold under some metric instead of geometry of data points. Replace N with corresponding covering number M and take $k \in \mathcal{O}\left(\epsilon^{-2} \log M\right)$.

JLL for unions of axis-aligned subspaces → RIP

RIP = Restricted isometry property (more on this later).
Proof idea:

1. Note that s-sparse d-dimensional vectors live on a union of $\binom{d}{s}$ s-dimensional subspaces.
2. Apply subspace-JLL to each s-flat.
3. Apply union bound to all $\binom{d}{s}$ subspaces.

$k = \mathcal{O}\left(\epsilon^{-2} s \log(\frac{12}{\epsilon}\frac{d}{s})\right)$ [BDDW08]



Outline

Applications of Random Projection (1)

We have seen, via JLL, that with a suitable choice of k we can construct an 'ϵ-approximate' version of *any* algorithm which depends only on the geometry of the data, but in a much lower-dimensional space. This includes:

- Nearest-neighbour algorithms.
- Clustering algorithms.
- Margin-based classifiers.
- Least-squares regressors.

That is, we trade off some accuracy (perhaps) for reduced algorithmic time and space complexity.

Using one RP...

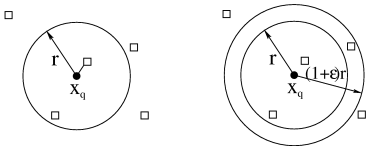
Diverse motivations for RP in the literature:

- To trade some accuracy in order to reduce computational expense and/or storage overhead (e.g. kNN).
- To bypass the collection of lots of data then throwing away most of it at preprocessing (Compressed sensing).
- To create a new theory of cognitive learning (RP Perceptron).
- To replace a heuristic optimizer with a provably correct algorithm with performance guarantees (e.g. mixture learning).

Solution: Work with random projections of the data.

Approximate Nearest Neighbour Search

- Kept theoreticians busy for over 40 years.
- Many applications: Machine Learning kNN rule; Database retrieval; Data compression (vector quantization).
- Exact Nearest Neighbour Search: Given a point set $\mathcal{T} = \{x_1, ..., x_N\}$ in $\mathbb{R}^d$, find the closest point to a query point x_q.
- Approximate NNS: Find $x \in \mathcal{T}$ that is ϵ-close to x_q. That is, such that $\forall x' \in \mathcal{T}, \|x - x_q\| \leqslant (1+\epsilon)\|x' - x_q\|$.



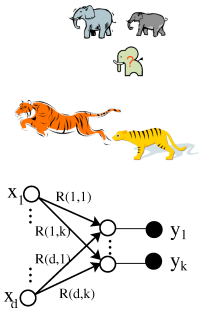
- The problem: Space or time complexity exponential in d even for sophisticated approximate NNS. [Kle97, HP01, AI06].

Nearest Neighbour Search

- The first known approximate NNS algorithm with space and time complexity polynomial in d is due to Indyk & Motwani '98[IM98]. It is based on the idea of locality sensitive hashing, and using the Johnson Lindenstrauss Lemma (JLL).
  - Have an algorithm with query time $\mathcal{O}\exp(d)$.
  - Apply JLL, so take $k = \mathcal{O}(\epsilon^{-2} C \log N)$ random projections.
  - This yields an algorithm that has query time $\mathcal{O}(N^{\epsilon^{-2} \cdot C})$.
- Since this important advance, there have been many further results on approximate NNS (including other uses of random projections! e.g. [Cha02]).

Neuronal RP Perceptron Learning

- Motivation: How does the brain learn concepts from a handful of examples when each example contains many features?
- Large margin ⇒ 'robustness' of concept.
- Idea:
  1. When the target concept robust, random projection of examples to a low-dimensional subspace preserves the concept.
  2. In the low-dimensional space, the number of examples and time required to learn concepts are comparatively small.



Definition. For any real number $\ell > 0$, a concept in conjunction with a distribution $\mathcal{D}$ on $\mathbb{R}^d \times \{-1, 1\}$, is said to be *robust*, if $\Pr\{x | \exists x' : label(x) \neq label(x'), \text{ and } \|x - x'\| \leqslant \ell\} = 0$.
Given $\mathcal{T} = \{(x_1, y_1), ..., (x_N, y_N)\} \sim \mathcal{D}^N$ labelled training set, $R \in \mathcal{M}_{k \times d}$ a random matrix with zero-mean sub-Gaussian entries.
Suppose $\mathcal{T}$ is a sample from a robust concept, i.e. $\exists h \in \mathbb{R}^d, \|h\| = 1$ s.t. $\forall n \in \{1, ..., N\}, y_n \cdot h^T x_n \geqslant \ell$.

Algorithm

1. Project $\mathcal{T}$ to $\mathcal{T}' = \{(Rx_1, y_1), ..., (Rx_N, y_N)\} \subseteq \mathbb{R}^k$.
2. Learn a perceptron $\hat{h}_R$ in $\mathbb{R}^k$ from $\mathcal{T}'$ (i.e. by minimizing training error).
3. Output R and $\hat{h}_R$.

For a query point x_q predict $sign(\hat{h}_R^T R x_q)$.

We now want to obtain a PAC learning guarantee on the generalization error and guarantee on running time of this algorithm.

Approach: Use known results on generalization [KV94] for halfspaces, and on the running time of Perceptron [MP69] in $\mathbb{R}^k$, and use JLL to ensure their preconditions hold w.h.p. Here we focus on the former (the latter goes similarly).

Theorem (Kearns & Vazirani '94) [for halfspaces, i.e. $VCdim = k + 1$].

Let $\mathcal{H}$ be the concept class of robust halfspaces. Let $\epsilon, \delta \in (0, 1)$, and let $h \in \mathcal{H}$ be a concept that is consistent with N i.i.d. labelled examples $\mathcal{T} \sim \mathcal{D}$.
Then, w.p. $\geqslant 1 - \delta$ (w.r.t. the random draws of the training set $\mathcal{T}$), h correctly classifies at least $1 - \epsilon$ fraction of $\mathcal{D}$ with probability at least $1 - \delta$ provided that $N > \frac{8k}{\epsilon} \log \frac{48}{\epsilon} + \frac{4}{\epsilon} \log \frac{2}{\epsilon}$.

Outline: First, w.r.t random draws of R, show the precondition of this theorem fails to hold w.p. $\leqslant \delta/2$. Next, w.r.t random draws of $\mathcal{T}$, apply the theorem so that it fails to hold w.p. $\leqslant \delta/2$. Union bound then gives PAC generalization guarantee w.p. $\geqslant 1 - \delta$.

First, consider $\hat{h} \in \mathbb{R}^d$ learned in the data space $\mathbb{R}^d$ by ERM, and note that $\hat{h}_R \in \mathbb{R}^k$ learned from the projected data in $\mathbb{R}^k$ must have training error no worse than any $R\hat{h}$, since $\hat{h}_R$ is the minimizer of the training error in $\mathbb{R}^k$.
We will work with $R\hat{h}$ rather than $\hat{h}_R$ and upper bound the error.

We need the following to hold except with probability less than $\delta/2$:

$$\forall x_n \in \mathcal{T}, \quad \text{if } \hat{h}^T x_n \geqslant \ell \text{ then } (R\hat{h})^T (Rx_n) \geqslant \ell/2; \tag{1}$$
$$\text{if } \hat{h}^T x_n \leqslant -\ell \text{ then } (R\hat{h})^T (Rx_n) \leqslant -\ell/2 \tag{2}$$

Denote $L = \max_{n=1,...,N} \|x_n\|^2$. We apply JLL to preserve all these N dot-products as well as all the lengths $\|x_n\|^2$ and $\|h\|^2$, with the choice $\ell/(2\sqrt{L})$ for the preservation tolerance parameter (i.e. in place of ϵ in JLL).
To have (1)-(2) except w.p. $\delta/2$ we need:

$$k = \mathcal{O}\left(\frac{L}{\ell^2} \cdot \log(12N/\delta)\right) \tag{3}$$

where L is the (squared) diameter of the data. We can take $L = 1$ w.l.o.g.
Therefore k of this order is needed to get the generalization bound via Kearns & Vazirani's theorem.

Comment: In [AV06] the authors obtain $k = \mathcal{O}\left(\frac{1}{\ell^2} \cdot \log(\frac{1}{\epsilon\ell\delta})\right)$ by taking the allowed misclassification rate as $\epsilon \geqslant \frac{1}{12N\ell}$. This somewhat obscures the logarithmic increase in the upper bound on generalization error with the number of training points N.

Provably Learning Mixtures of Gaussians

- Mixtures of Gaussians (MoG) are among the most fundamental and widely used statistical models. $p(x) = \sum_{y=1}^{K} \pi_y \mathcal{N}(x|\mu_y, \Sigma_y)$, where $\mathcal{N}(x|\mu_y, \Sigma_y) = \frac{1}{(2\pi)^{d/2}|\Sigma_y|^{1/2}} \exp(-\frac{1}{2}(x - \mu_y)^T \Sigma_y^{-1}(x - \mu_y))$.
- Given a set of unlabelled data points drawn from a MoG, the goal is to estimate the mean μ_y and covariance Σ_y for each source.
- Greedy heuristics (such as Expectation-Maximization) widely used for this purpose do not guarantee correct recovery of mixture parameters (can get stuck in local optima of the likelihood function).
- The first provably correct algorithm to learn a MoG from data is based on random projections.

Algorithm

Inputs: Sample S: set of N data points in $\mathbb{R}^d$; m = number of mixture components; ϵ,δ: resp. accuracy and confidence params. $\pi_{\min}$: smallest mixture weight to be considered.
(Values for other params derived from these via the theoretical analysis of the algorithm.)

1. Randomly project the data onto a k-dimensional subspace of the original space $\mathbb{R}^d$. Takes time $\mathcal{O}(Nkd)$.
2. In the projected space:
   - For $x \in S$, let r_x be the smallest radius such that there are $\geqslant p$ points within distance r_x of x.
   - Start with $S' = S$.
   - For $y = 1, ..., m$:
     - Let estimate $\hat{\mu}_y^*$ be the point x with the lowest r_x
     - Find the q closest points to this estimated center.
     - Remove these points from S'.
   - For each y, let S_y denote the l points in S which are closest to $\hat{\mu}_y^*$.
3. Let the (high-dimensional) estimate $\hat{\mu}_y$ be the mean of S_y in $\mathbb{R}^d$.

Definition

Two Gaussians $\mathcal{N}(\mu_1, \Sigma_1)$ and $\mathcal{N}(\mu_2, \Sigma_2)$ in $\mathbb{R}^d$ are said to be *c-separated* if $\|\mu_1 - \mu_2\| \geqslant c\sqrt{d \cdot \max\{\lambda_{\max}(\Sigma_1), \lambda_{\max}(\Sigma_2)\}}$. A mixture of Gaussians is c-separated if its components are c-separated.

Theorem

Let $\delta, \epsilon \in (0,1)$. Suppose the data is drawn from a mixture of m Gaussians in $\mathbb{R}^d$ which is c-separated, for $c > 1/2$, has (unknown) common covariance matrix Σ with condition number $\kappa = \lambda_{\max}(\Sigma)/\lambda_{\min}(\Sigma)$, and $\min_y \pi_y = \Omega(1/m)$. Then,

- w.p. $\geqslant 1 - \delta$, the centre estimates returned by the algorithm are accurate within ℓ_2 distance $\epsilon\sqrt{d\lambda_{\max}}$;
- if $\sqrt{\kappa} \leqslant \mathcal{O}(d^{1/2}/\log(m/(\epsilon\delta)))$, then the reduced dimension required is $k = \mathcal{O}(\log m/(\epsilon\delta))$, and the number of data points needed is $N = m^{\mathcal{O}(\log^2(1/(\epsilon\delta)))}$. The algorithm runs in time $\mathcal{O}(N^2 k + Nkd)$.

Construction of $z(\cdot)$ can be done using Bochner's theorem. This theorem says that every p.d. function on $\mathbb{R}^d$ can be written as the Fourier transform of a probability measure times a positive constant. So we have:

$$k(x_i, x_j) = Q(x_i - x_j) = \mathrm{E}_{w\sim p}[\alpha \exp(-iw(x_i - x_j))]$$

for some p and some $\alpha > 0$
Now, since $Q(x_i - x_j)$ is a real value, we can rewrite the above as $\mathrm{E}_{w\sim p}[\alpha \cos(w(x_i - x_j))]$.
Since cos is in $[-1, 1]$ using Hoeffding inequality we can approximate this expectation to within ϵ with a finite average:
$\frac{1}{N}\sum_{n=1}^{N} \alpha \cos(w_n(x_i - x_j))$, where $w_1, ..., w_N \sim_{i.i.d.} p$.
Rewriting this via trig identities we get:
$\frac{1}{M}\sum_{n=1}^{M} \alpha \cos(w_n x_i)\cos(w_n x_j) + \sin(w_n x_i)\sin(w_n x_j)) = z(x_i)^T z(x_j)$
where:
$z(x) := \sqrt{\frac{\alpha}{M}}(\cos(w_1 x), \sin(w_1 x), ..., \cos(w_M x), \sin(w_M x))$ and $w_1, , ..., w_M$ are iid random draws from p.
Example: for Gaussian kernel p is also Gaussian.

The proof is lengthy but it starts from the following observations:

- A c-separated mixture becomes a $(c \cdot \sqrt{1-\epsilon})$-separated mixture w.p. $1 - \delta$ after RP. This is because
  - JLL ensures that the distances between centers are preserved
  - $\lambda_{\max}(R\Sigma R^T) \leqslant \lambda_{\max}(\Sigma)$
- RP makes covariances more spherical (i.e. condition number decreases).

It is worth mentioning that the latest theoretical advances [KMV12] on learning of high dimensional mixture distributions under general conditions (i.e. overlap is allowed) in polynomial time also use RP.

We proved approximation of the kernel value for a fixed pair (x_i, x_j). To get this uniformly over all such pairs (not only in the training set) we cover $\mathcal{M} \times \mathcal{M}$, apply the above to each centre in the cover and take union bound. Finally extend to the whole space $\mathcal{M} \times \mathcal{M}$ using the fact that cos is a smooth function.
Code and examples are available at:
`berkeley.intel-research.net/arahimi/random-features/`

Other ways of constructing $z(\cdot)$ include hash kernels, for example [WDL$^+$09].

Random features as an alternative to the kernel trick

Kernel trick: $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$, $i, j \in 1, ..., N$.
Problem: When N is large, storage & computational cost is large.
Evaluating a test point x_q requires computing $f(x_q) = \sum_{i=1}^{N} c_i k(x_i, x_q)$.
This is $\mathcal{O}(Nd)$.
Idea: [RR08](also earlier in [Kon07]; later in work on Hash kernels [WDL$^+$09])
Construct $z : \mathbb{R}^d \to \mathbb{R}^k$ s.t. $|k(x_i, x_j) - z(x_i)^T z(x_j)| < \epsilon, \forall x_i, x_j \in \mathcal{M}$ where $\mathcal{M} \subset \mathbb{R}^d$ a compact domain.

For shift-invariant kernels this can be done for $k = \mathcal{O}(d\epsilon^{-2}\log(1/\epsilon^2))$.
Shift-invariant kernel: has the form $k(x_i, x_j) = Q(x_i - x_j)$, where Q is a positive definite function.

10 years later...

Outline

Compressed Sensing (1)

Often high-dimensional data is *sparse* in the following sense: There is some representation of the data in a linear basis such that most of the coefficients of the data vectors are (nearly) zero in this basis. For example, image and audio data in e.g. DCT basis.
Sparsity implies compressibility e.g. discarding small DCT coefficients gives us lossy compression techniques such as jpeg and mp3.

Idea: Instead of collecting sparse data and then compressing it to (say) 10% of its former size, what if we just captured 10% of the data in the first place?
In particular, what if we just captured 10% of the data at random? Could we reconstruct the original data?

Compressed (or Compressive) Sensing [Don06, CT06].

Compressed Sensing (2)

Problem: Want to reconstruct sparse d-dimensional signal x, with s non-zero coeffs. in sparse basis, given only k random measurements. i.e. we observe:

$$y = Rx,\ y \in \mathbb{R}^k,\ R \in \mathcal{M}_{k \times d},\ x \in \mathbb{R}^d,\ k \ll d.$$

and we want to find x given y. Since R is rank $k \ll d$ no unique solution in general.
However we also know that x is s-sparse...

Compressed Sensing (3)

Basis Pursuit Theorem (Candès-Tao 2004)

Let R be a $k \times d$ matrix and s an integer such that:

- $y = Rx$ admits an s-sparse solution $\hat{x}$, i.e. such that $\|\hat{x}\|_0 \leqslant s$.
- R satisfies the restricted isometry property (RIP) of order $(2s, \delta_{2s})$ with $\delta_{2s} \leqslant 2/(3 + \sqrt{7/4}) \simeq 0.4627$

Then:

$$\hat{x} = \arg\min_x \{\|x\|_1 : y = Rx\}$$

- If R and x satisfy the conditions on the BPT, then we can reconstruct x perfectly from its compressed representation, using efficient ℓ_1 minimization methods.
- We know x needs to be s-sparse. Which matrices R then satisfy the RIP?

Restricted Isometry Property

Restricted Isometry Property

Let R be a $k \times d$ matrix and s an integer. The matrix R satisfies the RIP of order (s, δ) provided that, for all s-sparse vectors $x \in \mathbb{R}^d$:

$$(1 - \delta)\|x\|_2^2 \leqslant \|Rx\|_2^2 \leqslant (1 + \delta)\|x\|_2^2$$

One can show that random projection matrices satisfying the JLL w.h.p also satisfy the RIP w.h.p provided that $k \in \mathcal{O}(s \log d)$. [BDDW08] does this using JLL combined with a covering argument in the projected space, finally union bound over all possible $\binom{d}{s}$ s-dimensional subspaces.
N.B. For signal reconstruction, data must be sparse: no perfect reconstruction guarantee from random projection matrices if $s > d/2$.

Compressed Learning

Intuition: If the data are s-sparse then one can perfectly reconstruct the data w.h.p from its randomly projected representation, provided that $k \in \mathcal{O}(s \log d)$. It follows that w.h.p no information was lost by carrying out the random projection.
Therefore one should be able to construct a classifier (or regressor) from the RP data which generalizes as well as the classifier (or regressor) learned from the original (non-RP) data.

Fast learning of SVM from sparse data

Theorem

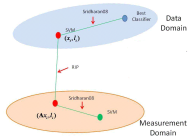
Calderbank et al. [CJS09] Let R be a $k \times d$ random matrix which satisfies the RIP. Let $RS = \{(Rx_1, y_1), ..., (Rx_N, y_N)\} \sim \mathcal{D}^N$.
Let $\hat{z}_{RS}$ be the soft-margin SVM trained on RS.
Let w_0 be the best linear classifier in the data domain with low hinge loss and large margin (hence small $\|w_0\|$).
Then, w.p. $1 - 2\delta$ (over RS):

$$H_{\mathcal{D}}(\hat{z}_{RS}) \leqslant H_{\mathcal{D}}(w_0) + \mathcal{O}\left(\sqrt{\|w_0\|^2\left(L^2\epsilon + \frac{\log(1/\delta)}{N}\right)}\right) \quad (4)$$

where $H_{\mathcal{D}}(w) = E_{(x,y)\sim\mathcal{D}}[1 - yw^T x]$ is the true hinge loss of the classifier in its argument, and $L = \max_n \|x_n\|$.

The proof idea is somewhat analogous to that in Arriaga & Vempala, with several differences:

- Major:
  - Data is assumed to be sparse. This allows using RIP instead of JLL and eliminates the dependence of the required k on the sample size N. Instead it will now depend (linearly) on s.
- Minor:
  - Different classifier
  - The best classifier is not assumed to have zero error



Proof sketch:

- Risk bound of Sridharan et al. [SSSS08] bounds the true SVM hinge loss of a classifier learned from data from that of the best classifier. Used twice: once in the data space, and again in the projection space.
- By definition (of best classifier), the true error of the best classifier in projected space is smaller than that of the projection of the best classifier in the data space.
- From RIP, derive the preservation of dot-products (similarly as previously in the case of JLL) which is then used to connect between the two spaces.

Outline

Compressive Linear Least Squares Regression

Given $\mathcal{T} = \{(x_1, y_1), ..., (x_N, y_N)\}$ with $x_n \in \mathbb{R}^d, y_n \in \mathbb{R}$.
Algorithm.

1. Let R a $k \times d$ RP matrix with entries $R_{ij} \sim \mathcal{N}(0, 1)$, let $P := R/\sqrt{k}$, and project the data: XP^T to $\mathbb{R}^k$.
2. Run a regression method in $\mathbb{R}_k$.

Result. Using JLL in conjunction with bounds on the excess risk of regression estimators with least squares loss, the gap between the true loss of the obtained estimator in the projected space and that of the optimal predictor in the data space can be bounded with high probability, provided that $k \in \mathcal{O}(\frac{8}{\epsilon^2}\log(8N/\delta))$.
For full details see [MM09].
Here we detail the special case of ordinary least squares regession (OLS).

Denote by X the design matrix having x_n in its rows, and Y a column vector with elements y_n.
Assume X is fixed (not random), and we want to learn an estimator $\hat{\beta}$ so that $X\hat{\beta}$ approximates $\mathrm{E}[Y|X]$.
Definitions in $\mathbb{R}^d$.
Squared loss: $L(w) = \frac{1}{N}\mathrm{E}_Y[\|Y - Xw\|^2]$ (where E_Y denotes $\mathrm{E}_{Y|X}$).
Optimal predictor: $\beta = \arg\min_w L(w)$.
Excess risk of an estimator: $R(\hat{\beta}) = L(\hat{\beta}) - L(\beta)$.
For linear regression this is: $(\hat{\beta} - \beta)^T\Sigma(\hat{\beta} - \beta)$ where $\Sigma = X^TX/N$.
OLS estimator: $\hat{\beta} := \arg\min_w \frac{1}{N}\|Y - Xw\|^2$

Proposition: OLS. If $\mathrm{Var}(Y_i) \leqslant 1$ then $\mathrm{E}_Y[R(\hat{\beta})] \leqslant \frac{d}{N}$.

Definitions in $\mathbb{R}^k$.
Square loss: $L_P(w) = \frac{1}{N}\mathrm{E}_Y[\|Y - (XP^T)w\|^2]$ (where E_Y denotes $\mathrm{E}_{Y|X}$).
Optimal predictor: $\beta_P = \arg\min_w L_P(w)$.

RP-OLS estimator: $\hat{\beta}_P := \arg\min_w \frac{1}{N}\|Y - (XP^T)w\|^2$

Proposition: Risk bound for RP-OLS
Assume $\mathrm{Var}(Y_i) \leqslant 1$, and let P as defined earlier. Then, for $k = \mathcal{O}(\log(8N/\delta)/\epsilon^2$ and any $\epsilon, \delta > 0$, w.p. $1 - \delta$ we have:

$$\mathrm{E}_Y[L_P(\hat{\beta}_P)] - L(\beta) \leqslant \frac{k}{N} + \|\beta\|^2\|\Sigma\|_{\mathrm{trace}}\epsilon^2 \tag{5}$$

Low-rank Matrix Approximation

Problem definition: Given a $d \times n$ matrix A, and an integer $s = k + p$, where $k = \mathrm{rank}(A)$, p is an oversampling factor, find a $d \times s$ orthonormal matrix B s.t. $A \approx BB^TA$.

Algorithm:

1. Generate *random matrix* R of size $n \times s$, by drawing i.i.d. entries from $\mathcal{N}(0, 1)$.
2. $B := \mathrm{orth}(AR)$ // columns of B form an orthonormal basis for the range of A. (Can be done by Gram-Schmidt or QR decomposition)

Theorem For B constructed as above, we have:

$$\mathrm{E}_R[\|A - BB^TA\|_F] \leqslant \left(1 + \frac{k}{p-1}\right)^{1/2}\left(\sum_{j>k}\sigma_j^2\right)^{1/2} \tag{11}$$

By Eckart-Young theorem, last term on RHS is minimal Frobenius norm error for rank k approximation of A.
More results in [HMT11, Mah11].

Proof. Applying Proposition OLS in $\mathbb{R}^k$, we get:

$$\mathrm{E}_Y[R(\hat{\beta}_P)] \leqslant \frac{k}{N} \tag{6}$$

Using definition of $R(\cdot)$, $\mathrm{E}_Y[R(\hat{\beta}_P)] = E_Y[L_P(\hat{\beta}_P)] - L_P(\beta_P)$. By definition of optimal predictor, $L_P(\beta_P) \leqslant L_P(P\beta)$. We rewrite and bound RHS using JLL for N dot products:

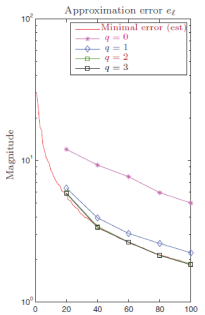
$$\begin{aligned} L_P(P\beta) &= \frac{1}{N}\mathrm{E}_Y[\|Y - XP^TP\beta\|^2] && (7)\\ &= \frac{1}{N}\mathrm{E}_Y[\|Y - X\beta\|^2] + \frac{1}{N}\|X\beta - XP^TP\beta\|^2 && (8)\\ &= L(\beta) + \frac{1}{N}\sum_{n=1}^{N}(x_nP^TP\beta - x_n\beta)^2 && (9)\\ &\leqslant L(\beta) + \frac{1}{N}\sum_{n=1}^{N}\|\beta\|^2\|x_n\|^2\epsilon^2 && (10)\end{aligned}$$

Noting that $\frac{1}{N}\sum_{n=1}^{N}\|x_n\|^2 = \|\Sigma\|^2_{\mathrm{trace}}$, and combining with eq. (6) gives the result.

Improved algorithm when spectrum decays slowly

1. Generate *random matrix* R of size $n \times s$, e.g. by drawing i.i.d. entries from $N(0, 1)$.
2. $B := \mathrm{orth}((AA^T)^qAR)$

Key point is that $(AA^T)^qA$ has the same eigenvectors as A but power iteration increases rate of SV decay.



Application to SVD

1. Construct B as above, i.e. so that $A \approx BB^TA$.
2. Let $C := B^TA$
3. Compute SVD of C: $C = \hat{U}\Sigma V^T$
4. Set $U := B\hat{U}$.

Application example: Computing eigenfaces with $A \in \mathcal{M}_{98304\times7254}$.
Graph shows approximation error versus s (From [HMT11]).

Extensions of Compressive Regression

- Compressive reinforcement learning [GLMM10].
- Compressive regression for sparse data, $k = \mathcal{O}(s \log d)$ [FGPP12].

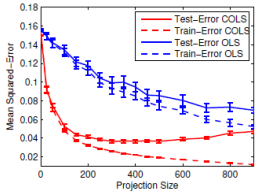


Figure: Example application to music similarity prediction from $d = 10^6$, very sparse; $N = 2000$. Taken from [FGPP12]

Randomized approximate SVM solver

In the (near-)separable case the dual formulation of SVM optimization problem is equivalent to minimizing distance between (reduced) convex hulls of classes [BB00, KBH08].
Minimizing distance between convex hulls belongs to class of abstract LP-type optimization problems - for fixed d there are *linear time* randomized algorithms e.g. [Sei91] for solving these.

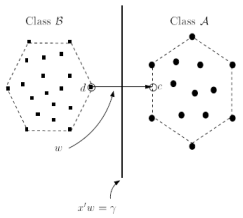


Figure: SVM dual formulation

Abstract LP-type Problems

Definition [MS03]

An abstract LP-type problem is a pair (H, w) where:

- H a finite set of constraints.
- $w : H^2 \to \mathbb{R} \cup (-\infty, \infty)$ an objective function to be minimized which satisfies, for any $h \in H$ and any $F \subseteq G \subseteq H$:
  - Monotonicity: $w(F) \leqslant w(G) \leqslant w(H)$.
  - Locality: If $w(F) = w(G) = w(F \cup h)$ then $w(F) = w(G \cup h)$.

Interpretation: $w(G)$ is the minimum value of a solution satisfying all constraints on G.

Basis and Combinatorial Dimension

Definitions: Basis, Combinatorial Dimension

$L = (H, w)$ abstract LP-type problem then:

- A basis for $F \subseteq H$ is a minimal set of constraints $B \subseteq F$ such that $w(B) = w(F)$.
- The combinatorial dimension of L is the size of the largest basis. Combinatorial dimension examples for problems in $\mathbb{R}^d$:
  - Smallest enclosing ball, $d + 1$
  - Linear program, $d + 1$
  - Distance between hyperplanes, $d + 2$

How to leverage JLL to solve SVM?

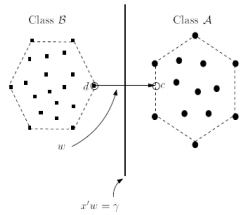


Figure: SVM dual formulation

Recall that SVM problem is to minimize $\|z_1 - z_2\|$ subject to: $z_1 \in \text{Conv}\{x_i : y_i = 1\}$ and $z_2 \in \text{Conv}\{x_i : y_i = -1\}$
How do we utilise random projections to solve this efficiently?

Solving LP-type problems

Let B be a basis for $F \subseteq H$ and $h \notin F$ a constraint. We need two primitive operations:

- Test for violation: Is $w(B \cup h) > w(B)$?
- Basis update: Set $B' = \text{basis}(B \cup h)$.

Random sampling for LP-type problems

Input: (H, w), Output: A basis B for H such that $w^\star = w(B)$.

LP-type(C, G) $G \subseteq H$, C some basis.
 if $G = C$
  return C
 else
  $S =$ random subset of H of size M
  $B = \text{basis}(\emptyset, S)$
  $V = \text{violators}(G - S, B)$
  while $(|V| > 0)$
   $R =$ a subset of size $M - |B|$ chosen randomly from V.
   $B = \text{basis}(B, R)$
   $V = \text{violators}(G - R, B)$
  end while
 return B

Sub-exponential algorithm for LP-type problems

Matousek, Sharir and Welzl 1996 [MSW96]

The expected running time of the algorithm **LP-type** is $\mathcal{O}\left(N \cdot e^{2\sqrt{d \log d}}\right)$, where $N = |\text{basis}(H)|$.

This gives a *linear time* algorithm for LP in fixed dimension.
What about for SVM? Fast SVM solver of Suresh et al., NIPS 2007 [KBH08].

Randomized SVM solver [KBH08]

Primitives: svmSolve(A, B) returns support vectors of $A \cup B$,
randSubset(A, i) selects i elements of A uniformly at random.
Input: Train set $\mathcal{T}_N$; S, est. # supp. vecs.; Sample size $M = ck$, $c > 1$.
Output: Set of support vectors SV.

RandSVM($\mathcal{T}_N, k, M$)
$\mathcal{T}_M$ = randSubset($\mathcal{T}_N, M$) // a random subset of M training examples.
SV = svmSolve($\emptyset, \mathcal{T}_M$) // set of support vectors of $\mathcal{T}_M$.
V = violators($T_N - T_M$) // set of KKT violators of $T_N - T_M$.
while ($|V| > 0$)&&($|SV| < k$) **do**
  R = randSubset($V, M - |SV|$) // random subset of V of size $M - |SV|$.
  SV' = svmSolve(SV, R)
  $SV = SV'$
  V = violators($T_N - (SV \cup R)$) // set of KKT violators from unsampled training examples.
end while
return SV

Running Time for Randomized SVM Solver

- Standard SVM solver has time complexity between $\mathcal{O}(N^2)$ (lower bound) and $\mathcal{O}(N^3)$ [BEWB05].
- For data in $\mathbb{R}^k$ expected time complexity of the randomized solver is $\mathcal{O}\left(N \cdot \exp(2\sqrt{k \log k})\right)$.
- Taking $k \in \mathcal{O}\left(\frac{1}{\epsilon^2} \log N\right)$ we can preserve the margin, w.h.p, apart from a scaling factor of $1 - \epsilon$.
- Then our approximate SVM solver has expected time complexity: $\mathcal{O}\left(N \cdot \exp\left(2\sqrt{k \log k}\right)\right) = \mathcal{O}\left(N \cdot \exp\left(2\sqrt{\log N \log\log N}\right)\right) = \mathcal{O}(N^2)$.
- Note no explicit random projection of the data, instead choose k large enough to guarantee w.h.p the solver can find a set of support vectors giving near-optimal margin.

Outline

Approaches not leveraging JLL or CS (1)

Recall our two initial problem settings:

- Very high dimensional data (arity $\in \mathcal{O}(1000)$) and very many observations ($N \in \mathcal{O}(1000)$): Computational (time and space complexity) issues.
- Very high dimensional data (arity $\in \mathcal{O}(1000)$) and hardly any observations ($N \in \mathcal{O}(10)$): Inference a hard problem. Bogus interactions between features.

Approaches not leveraging JLL or CS (2)

- What if we have many, many observations, $N \in \mathcal{O}(\exp(d))$ for example? Inefficient to use all of the data, and JLL guarantees now require $k \in \mathcal{O}(d)$ so no help there. Can we quantify the generalization error cost of RP without appealing to JLL or sparsity? RP-FLD or 'Compressed FLD'
- What if we have hardly any observations, $N \in \mathcal{O}(\log d)$ say? JLL then only requires $k \in \mathcal{O}(\log\log d)$, so $N \in \mathcal{O}(\exp(k))$. Can better parameter estimates in the projected space compensate for the distortion introduced by RP? Ensemble of RP-FLD classifiers

Guarantees without JLL

We can obtain guarantees for randomly-projected *classification* algorithms *without* directly applying the JLL, by using measure concentration and random matrix theoretic-based approaches.
Tackling the problem in this way removes the dependency on the number of observations, but at the cost of a dependency on the data dimensionality or a related quantity.
We shall consider two specific examples; a simple linear classifier and classifier ensemble, namely:

- RP Fisher's Linear Discriminant [DK10a, DK10b].
- An ensemble of RP-FLD classifiers [DK12b]

In these settings we can quantify the price we pay in exchange for lower algorithmic complexity, and our bounds will tighten in a natural way with increasing sample size.

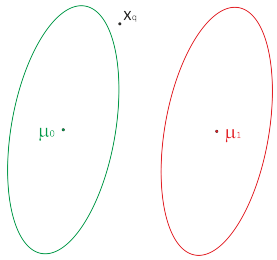
RP-FLD main ideas

- Classification is a much simpler task than perfect signal reconstruction - can we drop the sparsity condition of CS?
- For classification, often some distances are more important than others - can we guarantee good classification performance without preserving *all* of the data geometry?

By focusing on the classification problem we can get good performance guarantees for RP-FLD, including for non-sparse data, without directly applying JLL.

Fisher's Linear Discriminant

- Supervised learning approach.
- Simple and popular linear classifier, in widespread application.
- Classes are modelled as identical multivariate Gaussians.
- Assign class label to query point according to Mahalanobis distance from class means.
- FLD decision rule:



$$\hat{h}(\mathbf{x}_q) = \mathbf{1}\left\{(\hat{\mu}_1 - \hat{\mu}_0)^T \hat{\Sigma}^{-1}\left(\mathbf{x}_q - \frac{(\hat{\mu}_0 + \hat{\mu}_1)}{2}\right) > 0\right\}$$

Compressed Fisher's Linear Discriminant

'Compressed FLD': Learn the classifier and carry out the classification in the RP space.
Interested in quantifying the effect of random projection on the performance of the FLD classifier.
In particular average classifier performance, over the random picks of R, when $k \ll d$.
Things we don't need to worry about:

- Important points lying in the null space of R: Happens with probability 0.
- Problems mapping means, covariances in data space to RP space: All well-defined due to linearity of R and E[·].

RP-FLD decision rule:

$$\hat{h}_R(\mathbf{x}_q) = \mathbf{1}\left\{(\hat{\mu}_1 - \hat{\mu}_0)^T R^T \left(R\hat{\Sigma}R^T\right)^{-1} R\left(\mathbf{x}_q - \frac{(\hat{\mu}_0 + \hat{\mu}_1)}{2}\right) > 0\right\}$$

Guarantee on Compressed FLD

Theorem (Bound on Average Misclassification Error)

Let $\hat{h}$ be the FLD classifier learned from a fixed training set. Let $\mathbf{x}_q \sim \sum_{y=0}^{1} \pi_y \mathcal{N}(\mu_y, \Sigma)$, where $\Sigma \in \mathcal{M}_{d\times d}$ is a full rank covariance matrix. Let $R \in \mathcal{M}_{k\times d}$ be a random projection matrix with entries drawn i.i.d from the univariate Gaussian $\mathcal{N}(0,1)$. Then the estimated misclassification error $\hat{Pr}_{R,(\mathbf{x}_q,y_q)}[\hat{h}(R\mathbf{x}_q) \neq y_q]$ is bounded above by:

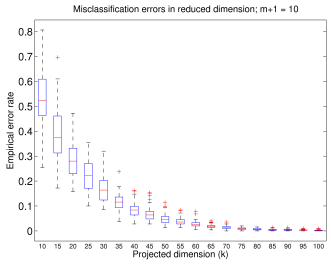
$$\left(1 + \frac{1}{4} g(\hat{\Sigma}^{-1}\Sigma) \cdot \frac{1}{d} \frac{\|\hat{\mu}_1 - \hat{\mu}_0\|^2}{\lambda_{\max}(\Sigma)}\right)^{-k/2} \quad (12)$$

with μ_y the mean of the class from which $\mathbf{x}_q$ was drawn, estimated class means $\hat{\mu}_0$ and $\hat{\mu}_1$, model covariance $\hat{\Sigma}$, and $g(Q) = 4 \cdot \frac{\lambda_{\max}(Q)}{\lambda_{\min}(Q)} \cdot \left(1 + \frac{\lambda_{\max}(Q)}{\lambda_{\min}(Q)}\right)^{-2}$.

Proof Outline

- Bound expected generalization error in data space (w.r.t. data distribution) - Chernoff bounding via m.g.f of Gaussian gives tractable form for working in the RP space. Bounding via m.g.f admits sub-Gaussian data distributions.
- Optimize bound - this brings in condition on true and estimated class centres.
- Bound expected generalization error in data space (w.r.t. data distribution, picks of random projection matrix R) - simplify using tricks from matrix analysis, ultimately bound error via m.g.f of χ^2. Bounding via m.g.f admits entries in R with sub-Gaussian distribution.

Validation - Effect of reducing k



Experiment showing effect of reducing k, $m+1$ 7-separated unit variance Gaussian classes. ($12\ln(10) \approx 28$). Cf. theorem (4), misclassification error reduces nearly exponentially as $k \nearrow d$. Worst performing R still allows weak learning.

Guarantee on Compressed FLD (2)

Theorem (Bound on Average Misclassification Error [DK11])

Under the same conditions as (4)*, the estimated misclassification error* $\hat{Pr}_{R,(\mathbf{x}_q,y_q)}[\hat{h}(R\mathbf{x}_q) \neq y_q]$ *is bounded above by:*

$$\exp\left(-\frac{k}{2}\log\left(1+\frac{1}{8d}\cdot\|\hat{\mu}_1-\hat{\mu}_0\|^2\cdot\frac{\lambda_{\min}(\hat{\Sigma}^{-1})}{\lambda_{\max}(\Sigma\hat{\Sigma}^{-1})}\right)\right) \quad (13)$$

Comments: Tighter than (4) sometimes (tight when $\hat{\Sigma}=\Sigma$, or when $\hat{\Sigma}$ or Σ are spherical) but less representative of error behaviour.

Corollary to (4) - Sufficient Dimensionality

Corollary (Sufficient Projection Dimensionality)

Under the same conditions as (4)*, for an* $m+1$*-class problem, in order that the probability of misclassification in the projected space remains below* δ *it is sufficient to take:*

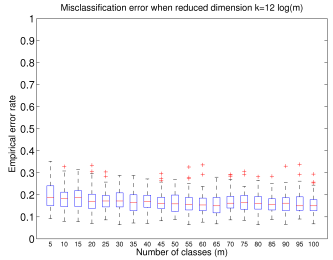
$$k \geqslant 8\cdot\frac{d\lambda_{\max}(\Sigma)}{\min_{i,j\in\mathcal{C},i\neq j}\|\hat{\mu}_i-\hat{\mu}_j\|^2}\cdot\frac{1}{g(\hat{\Sigma}^{-1}\Sigma)}\cdot\log(m/\delta) \quad (14)$$

Comments:
Compare with [AV99] for 2-class perceptron:

$$k=\mathcal{O}\left(\frac{L}{\ell^2}\cdot\log(12N/\delta)\right) \quad (15)$$

where L/ℓ^2 is the (squared) $\varnothing$ of the data ($L=\max_{n=1,\ldots,N}\|\mathbf{x}_n\|^2$) divided by the margin.

Validation - Corollary to Theorem (4)



Experiment confirming theorem (4) and corollary: Error is estimated from 500 random query points, and remains about constant when $k=12\log m$, $m+1$ 7-separated unit variance Gaussian classes.

Ensembles of RP-based methods

Motivation:

- Reducing variance
- Adding robustness
- Dealing with singularity

Applications of RP Ensembles: Clustering in RP space [FB03]. Face recognition [GBN05]. Covariance estimation when $N \ll d$ [MTS11]. Classification [SR09, DK12b].

Ensembles of RP-FLD

Assume a two-class classification problem, with N real-valued d-dimensional training observations:
$\mathcal{T}=\{(\mathbf{x}_i,y_i):(\mathbf{x},y)\in\mathbb{R}^d\times\{0,1\}\}_{i=1}^N$.

Furthermore assume that $N \ll d$, which is a common situation in practice (e.g. medical imaging, genomics, proteomics, face recognition, etc.), and that the unknown data distribution is full rank i.e. $\text{rank}(\Sigma)=d$. (Can relax to $\text{rank}(\Sigma) > N-2$.)

Challenges (1)

Problems:
Inferential issues: N is too small (for good estimation of model) w.r.t d $\iff$ d is too large w.r.t N.
Computational issues: Σ is singular (and must be inverted to construct classifier).

Solution: Compress data by random projection to $\mathbb{R}^k$, $k \leqslant N$. (Can relax to $k \leqslant d$.)

Challenges (2)

We just saw that for a *single* RP-FLD classification error grows nearly exponentially as $k \searrow 1$.

Solution:
Recover performance using an ensemble of RP FLD classifiers.

Ensembles that use some form of randomization in the design of the base classifiers have a long and successful history in machine learning: E.g. bagging [Bre96]; random subspaces [Ho98]; random forests [Bre01]; random projection ensembles [FB03, GBN05].

Comment: Potential for substantial computational savings, e.g. inversion of covariance matrix using Gauss-Jordan $\mathcal{O}\left(k^3\right)$ -vs- $\mathcal{O}\left(d^3\right)$ or Strassen algorithm $\mathcal{O}\left(k^{2.807}\right)$ -vs- $\mathcal{O}\left(d^{2.807}\right)$ where $k \ll d$.

Our Questions

- Can we recover (or improve on) level of classification performance in data space, using the RP FLD ensemble?
- Can we understand how the RP FLD ensemble acts to improve performance?
- Can we overfit the data with too large an RP-FLD ensemble?
- Can we interpret the RP ensemble classifier parameters in terms of data space parameters?

RP FLD Classifier Ensemble

For a single RP FLD classifier, the decision rule is given by:

$$\mathbf{1}\left\{(\hat{\mu}_1 - \hat{\mu}_0)^T R^T \left(R\hat{\Sigma}R^T\right)^{-1} R\left(\mathbf{x}_q - \frac{\hat{\mu}_1 + \hat{\mu}_0}{2}\right) > 0\right\}$$

which is the randomly projected analogue of the FLD decision rule. For the ensemble we use an equally weighted linear combination of RP FLD classifiers:

$$\mathbf{1}\left\{\frac{1}{M}\sum_{i=1}^{M}(\hat{\mu}_1 - \hat{\mu}_0)^T R_i^T \left(R_i\hat{\Sigma}R_i^T\right)^{-1} R_i\left(\mathbf{x}_q - \frac{\hat{\mu}_1 + \hat{\mu}_0}{2}\right) > 0\right\} \quad (16)$$

Linear combination rules are a common choice for ensembles. This rule works well in practice and it is also tractable to analysis.

Observation

We can rewrite decision rule as:

$$\mathbf{1}\left\{(\hat{\mu}_1 - \hat{\mu}_0)^T \frac{1}{M}\sum_{i=1}^{M} R_i^T \left(R_i\hat{\Sigma}R_i^T\right)^{-1} R_i \left(\mathbf{x}_q - \frac{\hat{\mu}_1 + \hat{\mu}_0}{2}\right) > 0\right\}$$

Then, for average case analysis with a *fixed* training set, it is enough to consider:

$$\lim_{M\to\infty} \frac{1}{M}\sum_{i=1}^{M} R_i^T \left(R_i\hat{\Sigma}R_i^T\right)^{-1} R_i = \mathrm{E}\left[R^T \left(R\hat{\Sigma}R^T\right)^{-1} R\right]$$

Proof Techniques(1)

Rows (and columns) of R drawn from a spherical Gaussian, hence for any orthogonal matrix U, $R \sim RU$. Eigendecomposing $\hat{\Sigma} = U\hat{\Lambda}U^T$ and using $UU^T = I$ we find that:

$$\mathrm{E}\left[R^T \left(R\hat{\Sigma}R^T\right)^{-1} R\right] = U\,\mathrm{E}\left[R^T \left(R\hat{\Lambda}R^T\right)^{-1} R\right] U^T \quad (17)$$

Furthermore since a matrix A is diagonal if and only if $VAV^T = A$ for all *diagonal* orthogonal matrices $V = \mathrm{diag}\{\pm 1\}$ we can similarly show that the expectation on RHS is diagonal.
Now enough to evaluate the diagonal terms on RHS!

Proof Techniques(2)

Define $\rho := \mathrm{rank}(\hat{\Lambda}) = \mathrm{rank}(\hat{\Sigma})$.
Write R as the concatenation of two submatrices: $R = [P|S]$ where P is $k \times \rho$ and S is $k \times d - \rho$. Decompose expectation on RHS of (17) as two diagonal blocks:

$$\mathrm{E}\left[R^T \left(R\Lambda R^T\right)^{-1} R\right] = \left[\begin{array}{c|c} \mathrm{E}[P^T \left(P\hat{\Lambda}P^T\right)^{-1} P] & 0 \\ \hline 0 & \mathrm{E}[S^T \left(P\hat{\Lambda}P^T\right)^{-1} S] \end{array}\right]$$

Finally estimate the remaining expectations.

Comment: For $1 \leqslant k \leqslant \rho - 2$ this expectation is evaluated exactly in [MTS11] using a complicated procedure. We are more interested in how it relates to characteristics of $\hat{\Sigma}$ so we prefer simply interpretable estimates.

Proof Techniques(3)

Work with positive semidefinite ordering: $A \succcurlyeq B \iff A - B$ is positive semidefinite (p.s.d $\equiv$ symmetric with all eigenvalues $\geqslant 0$).
Upper and lower bound the diagonal matrix expectation (17) in the p.s.d ordering with spherical matrices $\alpha_{\max} \cdot I$, $\alpha_{\min} \cdot I$ to bound its condition number in terms of *data space parameters*:

$$\alpha_{\max} \cdot I \succcurlyeq \mathrm{E}\left[R^T \left(R \Lambda R^T\right)^{-1} R\right] \succcurlyeq \alpha_{\min} \cdot I$$

Where $\alpha = \alpha(k, \rho, \lambda_{\max}, \lambda_{\min \neq 0})$, k is the projected dimensionality, $\rho = \mathrm{rank}(\hat{\Lambda}) = \mathrm{rank}(\hat{\Sigma})$, $\lambda_{\max}$ and $\lambda_{\min \neq 0}$ are respectively the greatest and least non-zero eigenvalues of $\hat{\Sigma}$.

Theory(1):Regularization

For fixed training set ρ, d are constant and $1 \leqslant k \leqslant d$ is the integer regularization parameter. There are three cases, and each implements a different regularization scheme:

$1 \leqslant k \leqslant \rho - 2$ Shrinkage regularization [LW04] in range of $\hat{\Sigma}$. Ridge regularization [HTF01] in null space of $\hat{\Sigma}$. As $k \nearrow \rho - 1$ less regularization.

$\rho + 2 \leqslant k \leqslant d$ Individual matrices in projected ensemble are singular, expectation is not. Pseudoinverting individual classifiers in the ensemble gives: No regularization in range of $\hat{\Sigma}$. Ridge regularization in null space of $\hat{\Sigma}$. As $k \searrow \rho + 1$ less regularization.

$\rho - 1 \leqslant k \leqslant \rho + 1$ Shrinkage regularization ($k = \rho - 1$) or no regularization ($k \in [\rho, \rho + 1]$) in range of $\hat{\Sigma}$. No regularization in null space of $\hat{\Sigma}$.

Theory(2):Exact Error of the Ensemble

Theorem (Ensemble error with Gaussian classes)

Let $\mathbf{x}_q \sim \sum_{y=0}^{1} \pi_y \mathcal{N}(\mu_y, \Sigma)$, where $\Sigma \in \mathcal{M}_{d \times d}$ is a full rank covariance matrix. Let $R \in \mathcal{M}_{k \times d}$ be a random projection matrix with i.i.d. Gaussian entries and denote $S_R^{-1} := \frac{1}{M} \sum_{i=1}^{M} R_i^T \left(R_i \hat{\Sigma} R_i^T\right)^{-1} R_i$. Then the exact error of the randomly projected ensemble classifier (16)*, conditioned on the training set, is given by:*

$$\sum_{y=0}^{1} \pi_y \Phi\left(-\frac{1}{2} \frac{(\hat{\mu}_{\neg y} - \hat{\mu}_y)^T S_R^{-1} (\hat{\mu}_0 + \hat{\mu}_1 - 2\mu_y)}{\sqrt{(\hat{\mu}_1 - \hat{\mu}_0)^T S_R^{-1} \Sigma S_R^{-1} (\hat{\mu}_1 - \hat{\mu}_0)}}\right)$$

For the converged ensemble, substitute the expectation (17) for S_R^{-1} above.

Experiments: Datasets

Table: Datasets

Name	Source	#samples	#features
colon	Alon et al. [ABN$^+$99]	62	2000
leukemia	Golub et al. [GST$^+$99]	72	3571
leukemia large	Golub et al. [GST$^+$99]	72	7129
prostate	Singh et al. [SFR$^+$02]	102	6033
duke	West et al. [WBD$^+$01]	44	7129

Experiments: Protocol

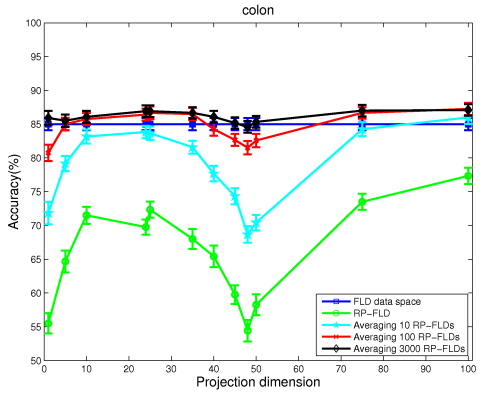
- Standardized features to have mean 0 and variance 1 and ran experiments on 100 independent splits. In each split took 12 points for testing, rest for training.
- For data space experiments on colon and leukemia used ridge-regularized FLD and fitted regularization parameter using 5-fold CV on the first five data splits following [MRW$^+$02].
- For other datasets we used diagonal FLD in the data space (size, no sig. diff. in error on colon, leuk.).
- RP base learners: FLDs with full covariance and no regularization when $k \leqslant \rho$ and pseudoinverted FLD when $k > \rho$.
- Compared performance with SVM with linear kernel as in [FM03].

Experiments: Results for $k = \rho/2$

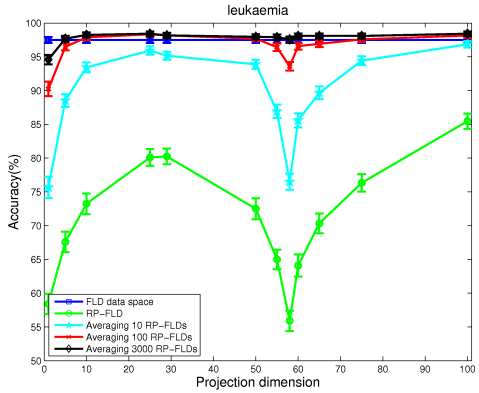
Table: Mean error rates $\pm$ 1 standard error, estimated from 100 independent splits when $k = \rho/2$.

Dataset	$\rho/2$	100 RP-FLD	1000 RP-FLD	SVM
colon	24	13.58 $\pm$ 0.89	13.08 $\pm$ 0.86	16.58 $\pm$ 0.95
leuk.	29	1.83 $\pm$ 0.36	1.83 $\pm$ 0.37	1.67 $\pm$ 0.36
leuk.lg.	29	4.91$\pm$ 0.70	3.25 $\pm$ 0.60	3.50 $\pm$ 0.46
prost.	44	8.00 $\pm$ 0.76	8.00 $\pm$ 0.72	8.00 $\pm$ 0.72
duke	15	17.41 $\pm$1.27	16.58 $\pm$ 1.27	13.50 $\pm$ 1.10

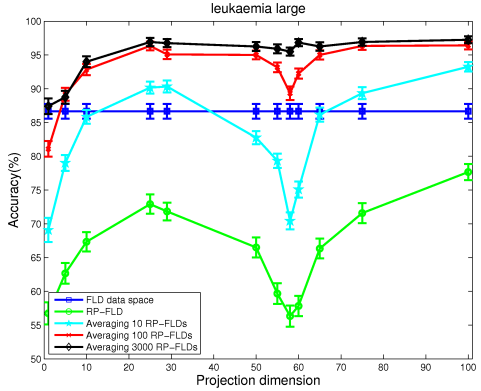
Experiments: Colon, $R_{ij} \sim \mathcal{N}(0,1)$



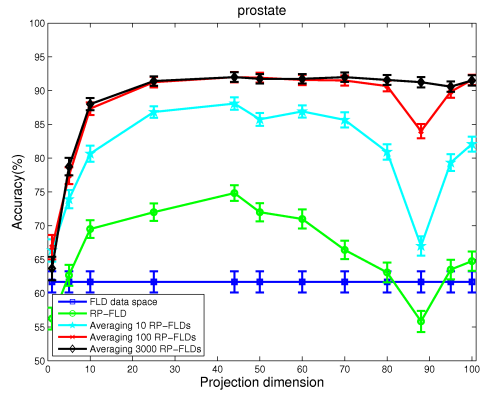
Experiments: Leukemia, $R_{ij} \sim \mathcal{N}(0,1)$



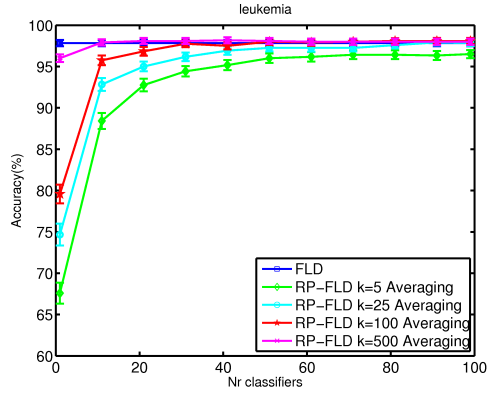
Experiments: Leukemia Large, $R_{ij} \sim \mathcal{N}(0,1)$



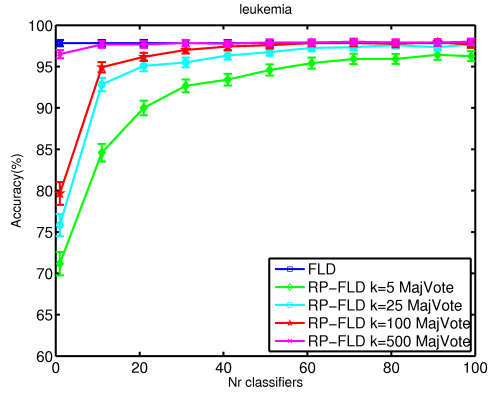
Experiments: Effect of Ensemble Size, $R_{ij} \sim \mathcal{N}(0,1)$



Experiments: Leukemia, $R_{ij} \sim \{-1, 0, +1\}$



Experiments: Leukemia, $R_{ij} \sim \mathcal{N}(0,1)$, Majority Vote



Answers from Ensembles of RP-FLD

- Can we recover (or improve on) level of classification performance in data space, using the RP FLD ensemble? **YES**
- Can we understand how the RP FLD ensemble acts to improve performance? **YES**
- Can we overfit the data with the RP FLD ensemble? **NO (with appropriate choice of k)**
- Can we interpret the ensemble classifier parameters in terms of data space parameters? **YES**

Take me home!

- Random projections have a wide range of *theoretically well-motivated* and *effective* applications in machine learning and data mining.
- In particular, random projections:
  - are easy to implement,
  - can reduce time and space complexity of algorithms for small performance cost, and
  - can be used to construct parallel implementations of existing algorithms.
- Because random projection is *independent of data distribution*, theoretical analysis possible unlike many deterministic approaches.
- In particular, can derive guarantees for random projections but not for approaches such as PCA.
- In high dimensions RP matrices act like approximate isometries, preserving geometric properties of data well but in a lower dimensional space.

References I

[ABN+99] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proceedings of the National Academy of Sciences **96** (1999), no. 12, 6745.

[AC06] N. Ailon and B. Chazelle, *Approximate nearest neighbors and the fast johnson-lindenstrauss transform*, Proceedings of the thirty-eighth annual ACM symposium on Theory of computing, ACM, 2006, pp. 557–563.

[Ach03] D. Achlioptas, *Database-friendly random projections: Johnson-Lindenstrauss with binary coins*, Journal of Computer and System Sciences **66** (2003), no. 4, 671–687.

[AI06] A. Andoni and P. Indyk, *Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions*, Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on, IEEE, 2006, pp. 459–468.

[AL09] N. Ailon and E. Liberty, *Fast dimension reduction using rademacher series on dual bch codes*, Discrete & Computational Geometry **42** (2009), no. 4, 615–630.

[AL11] _____, *An almost optimal unrestricted fast johnson-lindenstrauss transform*, Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2011, pp. 185–191.

[Alo03] N. Alon, *Problems and results in extremal combinatorics, Part I*, Discrete Math **273** (2003), 31–53.

References II

[AMS96] N. Alon, Y. Matias, and M. Szegedy, *The space complexity of approximating the frequency moments*, Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, ACM, 1996, pp. 20–29.

[AV99] R.I. Arriaga and S. Vempala, *An algorithmic theory of learning: Robust concepts and random projection*, Foundations of Computer Science, 1999. 40th Annual Symposium on, IEEE, 1999, pp. 616–623.

[AV06] R. Arriaga and S. Vempala, *An algorithmic theory of learning*, Machine Learning **63** (2006), no. 2, 161–182.

[AV09] R. Avogadri and G. Valentini, *Fuzzy ensemble clustering based on random projections for dna microarray data analysis*, Artificial Intelligence in Medicine **45** (2009), no. 2, 173–183.

[BB00] K.P. Bennett and E.J. Bredensteiner, *Duality and geometry in svm classifiers*, 17th International Conference on Machine Learning (ICML 2000), 2000, pp. 57–64.

[BD09] C. Boutsidis and P. Drineas, *Random projections for the nonnegative least-squares problem*, Linear Algebra and its Applications **431** (2009), no. 5-7, 760–771.

[BDDW08] R.G. Baraniuk, M. Davenport, R.A. DeVore, and M.B. Wakin, *A Simple Proof of the Restricted Isometry Property for Random Matrices* , Constructive Approximation **28** (2008), no. 3, 253–263.

References III

[BEWB05] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, *Fast kernel classifiers with online and active learning*, The Journal of Machine Learning Research **6** (2005), 1579–1619.

[BM01] E. Bingham and H. Mannila, *Random projection in dimensionality reduction: applications to image and text data.*, Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001) (F. Provost and R. Srikant, ed.), 2001, pp. 245–250.

[Bre96] L. Breiman, *Bagging predictors*, Machine learning **24** (1996), no. 2, 123–140.

[Bre01] _____, *Random forests*, Machine learning **45** (2001), no. 1, 5–32.

[BW09] R.G. Baraniuk and M.B. Wakin, *Random projections of smooth manifolds*, Foundations of Computational Mathematics **9** (2009), no. 1, 51–77.

[Cha02] M.S. Charikar, *Similarity estimation techniques from rounding algorithms*, Proceedings of the thirty-fourth annual ACM symposium on Theory of computing, ACM, 2002, pp. 380–388.

[CJS09] R. Calderbank, S. Jafarpour, and R. Schapire, *Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain*, Tech. report, Rice University, 2009.

References IV

[CT06] E.J. Candes and T. Tao, *Near-optimal signal recovery from random projections: Universal encoding strategies?*, Information Theory, IEEE Transactions on **52** (2006), no. 12, 5406–5425.

[Das99] S. Dasgupta, *Learning Mixtures of Gaussians*, Annual Symposium on Foundations of Computer Science, vol. 40, 1999, pp. 634–644.

[DF08] S. Dasgupta and Y. Freund, *Random projection trees and low dimensional manifolds*, Proceedings of the 40th annual ACM symposium on Theory of computing, ACM, 2008, pp. 537–546.

[DG02] S. Dasgupta and A. Gupta, *An Elementary Proof of the Johnson-Lindenstrauss Lemma*, Random Struct. Alg. **22** (2002), 60–65.

[DK10a] R.J. Durrant and A. Kabán, *A bound on the performance of LDA in randomly projected data spaces*, Proceedings 20th International Conference on Pattern Recognition (ICPR 2010), 2010, pp. 4044–4047.

[DK10b] _____, *Compressed Fisher Linear Discriminant Analysis: Classification of Randomly Projected Data*, Proceedings16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2010), 2010.

[DK11] _____, *A tight bound on the performance of Fisher's linear discriminant in randomly projected data spaces.*, Pattern Recognition Letters (2011).

References V

[DK12a] _____, *Error bounds for Kernel Fisher Linear Discriminant in Gaussian Hilbert space*, Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AIStats 2012), 2012.

[DK12b] _____, *Random Projections as Regularizers: Learning a Linear Discriminant Ensemble from Fewer Observations than Dimensions*, Tech. Report CSR-12-01, University of Birmingham, School of Computer Science, 2012.

[DKS10] A. Dasgupta, R. Kumar, and T. Sarlós, *A sparse johnson-lindenstrauss transform*, Proceedings of the 42nd ACM symposium on Theory of computing, ACM, 2010, pp. 341–350.

[Don06] D.L. Donoho, *Compressed Sensing*, IEEE Trans. Information Theory **52** (2006), no. 4, 1289–1306.

[FB03] X.Z. Fern and C.E. Brodley, *Random projection for high dimensional data clustering: A cluster ensemble approach*, International Conference on Machine Learning, vol. 20, 2003, p. 186.

[FGPP12] M. Fard, Y. Grinberg, J. Pineau, and D. Precup, *Compressed least-squares regression on sparse spaces*, Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[FM03] D. Fradkin and D. Madigan, *Experiments with random projections for machine learning*, Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2003, pp. 522–529.

References VI

[GBN05] N. Goel, G. Bebis, and A. Nefian, *Face recognition experiments with random projection*, Proceedings of SPIE, vol. 5779, 2005, p. 426.

[GLMM10] M. Ghavamzadeh, A. Lazaric, O.A. Maillard, and R. Munos, *Lstd with random projections*, Advances in Neural Information Processing Systems 23 (J. Lafferty, C.K.I Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, eds.), 2010, pp. 721–729.

[GST+99] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander, *Molecular classification of cancer: class discovery and class prediction by gene expression monitoring*, Science **286** (1999), no. 5439, 531.

[HMT11] N. Halko, P.G. Martinsson, and J.A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM review **53** (2011), no. 2, 217–288.

[Ho98] T.K. Ho, *The random subspace method for constructing decision forests*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **20** (1998), no. 8, 832–844.

[HP01] S. Har-Peled, *A replacement for voronoi diagrams of near linear size*, Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on, IEEE, 2001, pp. 94–103.

[HTF01] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning; data mining, inference, and prediction*, Springer, 2001.

References VII

[HWB07] C. Hegde, M.B. Wakin, and R.G. Baraniuk, *Random projections for manifold learningproofs and analysis*, Neural Information Processing Systems, 2007.

[IM98] P. Indyk and R. Motwani, *Approximate nearest neighbors: towards removing the curse of dimensionality*, Proceedings of the thirtieth annual ACM symposium on Theory of computing, ACM New York, NY, USA, 1998, pp. 604–613.

[JW11] T.S. Jayram and D. Woodruff, *Optimal bounds for johnson-lindenstrauss transforms and streaming problems with sub-constant error*, Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2011, pp. 1–10.

[KBH08] S. Krishnan, C. Bhattacharyya, and R. Hariharan, *A randomized algorithm for large scale support vector learning*, 2008, pp. 793–800.

[Kle97] J.M. Kleinberg, *Two algorithms for nearest-neighbor search in high dimensions*, Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, ACM, 1997, pp. 599–608.

[KMN11] D. Kane, R. Meka, and J. Nelson, *Almost optimal explicit johnson-lindenstrauss families*, Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (2011), 628–639.

[KMV12] A.T. Kalai, A. Moitra, and G. Valiant, *Disentangling gaussians*, Communications of the ACM **55** (2012), no. 2, 113–120.

References VIII

[Kon07] L. Kontorovich, *A universal kernel for learning regular languages*, Machine Learning in Graphs (2007).

[KV94] M.J. Kearns and U.V. Vazirani, *An introduction to computational learning theory*, MIT Press, 1994.

[Led01] M. Ledoux, *The concentration of measure phenomenon*, vol. 89, American Mathematical Society, 2001.

[LW04] O. Ledoit and M. Wolf, *A well-conditioned estimator for large-dimensional covariance matrices*, Journal of multivariate analysis **88** (2004), no. 2, 365–411.

[Mah11] M.W. Mahoney, *Randomized algorithms for matrices and data*, arXiv preprint arXiv:1104.5557 (2011).

[Mat08] J. Matoušek, *On variants of the johnson–lindenstrauss lemma*, Random Structures & Algorithms **33** (2008), no. 2, 142–156.

[MM09] O. Maillard and R. Munos, *Compressed Least-Squares Regression*, NIPS, 2009.

[MP69] M. Minsky and S. Papert, *Perceptrons*.

[MRW+02] S. Mika, G. Ratsch, J. Weston, B. Schölkopf, and KR Mullers, *Fisher discriminant analysis with kernels*, Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop, IEEE, 2002, pp. 41–48.

[MS03] J. Matousek and P. Skovron, *Three views of lp-type optimization problems*, 2003.

References IX

[MSW96] J. Matoušek, M. Sharir, and E. Welzl, *A subexponential bound for linear programming*, Algorithmica **16** (1996), no. 4, 498–516.

[MTS11] T.L. Marzetta, G.H. Tucci, and S.H. Simon, *A Random Matrix–Theoretic Approach to Handling Singular Covariance Estimates*, IEEE Trans. Information Theory **57** (2011), no. 9, 6256–71.

[Rec11] B. Recht, *A simpler approach to matrix completion*, Journal of Machine Learning Research **12** (2011), 3413–3430.

[RR08] A. Rahimi and B. Recht, *Random features for large-scale kernel machines*, Advances in neural information processing systems **20** (2008), 1177–1184.

[Sar06] T. Sarlos, *Improved approximation algorithms for large matrices via random projections*, Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on, IEEE, 2006, pp. 143–152.

[Sei91] R. Seidel, *Small-dimensional linear programming and convex hulls made easy*, Discrete & Computational Geometry **6** (1991), no. 1, 423–434.

[SFR+02] D. Singh, P.G. Febbo, K. Ross, D.G. Jackson, J. Manola, C. Ladd, P. Tamayo, A.A. Renshaw, A.V. D'Amico, J.P. Richie, E.S. Lander, M. Loda, P.W. Kantoff, T.R. Golub, and W.S. Sellers, *Gene expression correlates of clinical prostate cancer behavior*, Cancer cell **1** (2002), no. 2, 203–209.

References X

[SR09] A. Schclar and L. Rokach, *Random projection ensemble classifiers*, Enterprise Information Systems (Joaquim Filipe, Jos Cordeiro, Wil Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, and Clemens Szyperski, eds.), Lecture Notes in Business Information Processing, vol. 24, Springer, 2009, pp. 309–316.

[SSSS08] K. Sridharan, N. Srebro, and S. Shalev-Shwartz, *Fast rates for regularized objectives*, Advances in Neural Information Processing Systems 21, 2008, pp. 1545–1552.

[WBD+01] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, J.R. Marks, and J.R. Nevins, *Predicting the clinical status of human breast cancer by using gene expression profiles*, Proceedings of the National Academy of Sciences **98** (2001), no. 20, 11462.

[WDL+09] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, *Feature hashing for large scale multitask learning*, Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 1113–1120.

Appendix

Proposition JLL for dot products.
Let $x_n, n = \{1 \dots N\}$ and u be vectors in $\mathbb{R}^d$ s.t. $\|x_n\|, \|u\| \leqslant 1$.
Let R be a $k \times d$ RP matrix with i.i.d. entries $R_{ij} \sim \mathcal{N}(0, 1/\sqrt{k})$ (or with zero-mean sub-Gaussian entries).
Then for any $\epsilon, \delta > 0$, if $k \in \mathcal{O}\left(\frac{8}{\epsilon^2} \log(4N/\delta)\right)$ w.p. at least $1 - \delta$ we have:

$$|x_n^T u - (Rx_n)^T Ru| < \epsilon \tag{18}$$

simultaneously for all $n = \{1 \dots N\}$.

Proof of JLL for dot products

Outline: Fix one n, use parallelogram law and JLL twice, then use union bound.

$$\begin{aligned} 4(Rx_n)^T(Ru) &= \|Rx_n + Ru\|^2 - \|Rx_n - Ru\|^2 && (19)\\ &\geqslant (1-\epsilon)\|x_n + u\|^2 - (1+\epsilon)\|x_n - u\|^2 && (20)\\ &= 4x_n^T u - 2\epsilon(\|x_n\|^2 + \|u\|^2) && (21)\\ &\geqslant 4x_n^T u - 4\epsilon && (22) \end{aligned}$$

Hence, $(Rx_n)^T(Ru) \geqslant x_n^T u - \epsilon$, and because we used two sides of JLL, this holds except w.p. no more than $2\exp(-k\epsilon^2/8)$.
The other side is similar and gives $(Rx_n)^T(Ru) \leqslant x_n^T u + \epsilon$ except w.p. $2\exp(-k\epsilon^2/8)$.
Put together, $|(Rx_n)^T(Ru) - x_n^T u| \leqslant \epsilon \cdot \frac{\|x\|^2 + \|u\|^2}{2} \leqslant \epsilon$ holds except w.p. $4\exp(-k\epsilon^2/8)$.
This holds for a fixed x_n. To ensure that it holds for all x_n together, we take union bound and obtain eq.(18) must hold except w.p. $4N\exp(-k\epsilon^2/8)$. Finally, solving for δ we obtain that $k \geqslant \frac{8}{\epsilon^2}\log(4N/\delta)$.