Random Projections & Canonical Correlation Analysis

Course Webpage:
http://www.cs.cornell.edu/Courses/cs4786/2017fa/
The Tall, the Fat and the Ugly

X

n
d
The Tall, the Fat and the Ugly

\[
d X^\top \times n X \quad \xrightarrow{\text{}} \quad n = d \Sigma
\]
The Tall, the Fat and the Ugly

\[
\begin{align*}
&d \\
& X^T \\
& \times \\
& n \\
& X \\
& \left/ \right. \\
& n = d \\
& \sum \\
& d \\
& W \\
& K \\
& = \text{Eigs} \left( \sum, K \right)
\end{align*}
\]
The Tall, the Fat and the Ugly
THE TALL, the Fat AND the Ugly

\[ \text{SVD}(X) \]

\[ n \times d \]

\[ U \times \begin{pmatrix} \vdots \end{pmatrix} \]

\[ d \times d \]

\[ V^T \]
THE TALL, the Fat AND the Ugly

\[ \text{SVD}(X) \]

\[ n \times n \quad X \quad d \]

\[ U \times K \quad V^T \]
\begin{itemize}
  \item $d$ and $n$ so large we can’t even store in memory
  \item Only have time to be linear in $\text{size}(X) = n \times d$
\end{itemize}

I there any hope?
Pick a Random $W$

$$Y = X \times \begin{bmatrix} +1 & \ldots & -1 \\ -1 & \ldots & +1 \\ +1 & \ldots & -1 \\ \vdots \\ +1 & \ldots & -1 \end{bmatrix} \times \begin{bmatrix} d \\ \sqrt{K} \end{bmatrix}$$
Why should Random Projections even work?!
What does “it works” even mean?
What does “it works” even mean?

Distances between all pairs of data-points in low dim. projection is roughly the same as their distances in the high dim. space.
What does “it works” even mean?

Distances between all pairs of data-points in low dim. projection is roughly the same as their distances in the high dim. space.

That is, when \( K \) is “large enough”, with “high probability”, for all pairs of data points \( i, j \in \{1, \ldots, n\} \),

\[
(1 - \epsilon) \|y_i - y_j\|_2 \leq \|x_i - x_j\|_2 \leq (1 + \epsilon) \|y_i - y_j\|_2
\]
Say $K = 1$. Consider any vector $\tilde{x} \in \mathbb{R}^d$ and let $\tilde{y} = \tilde{x} W$. Note that
Say $K = 1$. Consider any vector $\tilde{x} \in \mathbb{R}^d$ and let $\tilde{y} = \tilde{x} W$. Note that

$$\tilde{y}^2 = \left( \sum_{i=1}^{d} W[i,1] \cdot \tilde{x}[i] \right)^2$$
Why should Random Projections even work?!

Say $K = 1$. Consider any vector $\tilde{x} \in \mathbb{R}^d$ and let $\tilde{y} = \tilde{x} W$. Note that

$$\tilde{y}^2 = \left( \sum_{i=1}^{d} W[i, 1] \cdot \tilde{x}[i] \right)^2$$

$$= \sum_{i=1}^{d} (W[i, 1] \cdot \tilde{x}[i])^2 + 2 \sum_{i' > i} (W[i, 1] \cdot \tilde{x}[i]) (W[i', 1] \cdot \tilde{x}[i'])$$
Say $K = 1$. Consider any vector $\tilde{x} \in \mathbb{R}^d$ and let $\tilde{y} = \tilde{x} W$. Note that

\[
\tilde{y}^2 = \left( \sum_{i=1}^{d} W[i, 1] \cdot \tilde{x}[i] \right)^2
\]

\[
= \sum_{i=1}^{d} (W[i, 1] \cdot \tilde{x}[i])^2 + 2 \sum_{i' > i} (W[i, 1] \cdot \tilde{x}[i]) (W[i', 1] \cdot \tilde{x}[i'])
\]

\[
= \sum_{i=1}^{d} W^2[i, 1] \tilde{x}^2[i] + \sum_{i' > i} (W[i, 1] \cdot W[i', 1]) \cdot (\tilde{x}[i] \cdot \tilde{x}[i'])
\]
Why should Random Projections even work?!

Say $K = 1$. Consider any vector $\tilde{x} \in \mathbb{R}^d$ and let $\tilde{y} = \tilde{x} W$. Note that

$$\tilde{y}^2 = \left( \sum_{i=1}^d W[i, 1] \cdot \tilde{x}[i] \right)^2$$

$$= \sum_{i=1}^d (W[i, 1] \cdot \tilde{x}[i])^2 + 2 \sum_{i' > i} (W[i, 1] \cdot \tilde{x}[i]) (W[i', 1] \cdot \tilde{x}[i'])$$

$$= \sum_{i=1}^d W^2[i, 1] \tilde{x}^2[i] + \sum_{i' > i} (W[i, 1] \cdot W[i', 1]) \cdot (\tilde{x}[i] \cdot \tilde{x}[i'])$$

However $W^2[i, 1] = 1/K = 1$ when $K = 1$

$$= \sum_{i=1}^d \tilde{x}^2[i] + \sum_{i' > i} (W[i, 1] \cdot W[i', 1]) \cdot (\tilde{x}[i] \cdot \tilde{x}[i'])$$
Hence,

\[
\mathbb{E}[\tilde{y}^2] = \sum_{i=1}^{d} \tilde{x}^2[i] + \sum_{i' > i} \mathbb{E}[W[i, 1] \cdot W[i', 1]] \cdot (\tilde{x}[i] \cdot \tilde{x}[i'])
\]
Hence,

\[ \mathbb{E}[\tilde{y}^2] = \sum_{i=1}^{d} \tilde{x}^2[i] + \sum_{i' > i} \mathbb{E}[W[i, 1] \cdot W[i', 1]] \cdot (\tilde{x}[i] \cdot \tilde{x}[i']) \]

However \( W[i, 1] \) and \( W[i', 1] \) are independent and so

\[ \mathbb{E}[W[i, 1] \cdot W[i', 1]] = \mathbb{E}[W[i, 1]] \cdot \mathbb{E}[W[i', 1]] = 0 \]
Hence,

\[ \mathbb{E}[\tilde{y}^2] = \sum_{i=1}^{d} \tilde{x}^2[i] + \sum_{i'>i} \mathbb{E}[W[i, 1] \cdot W[i', 1]] \cdot (\tilde{x}[i] \cdot \tilde{x}[i']) \]

However \( W[i, 1] \) and \( W[i', 1] \) are independent and so

\[ \mathbb{E}[W[i, 1] \cdot W[i', 1]] = \mathbb{E}[W[i, 1]] \cdot \mathbb{E}[W[i', 1]] = 0 \]

Using this we conclude that

\[ \mathbb{E}[\tilde{y}^2] = \sum_{i=1}^{d} \tilde{x}^2[i] = \|\tilde{x}\|^2 \]
Why should Random Projections even work?! 

Hence,

\[ \mathbb{E}[|\tilde{y}|^2] = \|\tilde{x}\|_2^2 \]
Hence,

\[ E[|\tilde{y}|^2] = \|\tilde{x}\|^2 \]

If we let \( \tilde{x} = x_s - x_t \) then

\[ \tilde{y} = \tilde{x}W = x_s W - x_t W = y_s - y_t \]
Hence,

$$\mathbb{E}[|\tilde{y}|^2] = \|\tilde{x}\|^2$$

If we let $\tilde{x} = x_s - x_t$ then

$$\tilde{y} = \tilde{x}W = x_sW - x_tW = y_s - y_t$$

Hence for any $s, t \in \{1, \ldots, n\}$,

$$\mathbb{E}[|y_s - y_t|^2] = \|x_s - x_t\|^2$$
Hence,

$$\mathbb{E}[|\tilde{y}|^2] = \|\tilde{x}\|^2$$

If we let $\tilde{x} = x_s - x_t$ then

$$\tilde{y} = \tilde{x}W = x_s W - x_t W = y_s - y_t$$

Hence for any $s, t \in \{1, \ldots, n\}$,

$$\mathbb{E}[|y_s - y_t|^2] = \|x_s - x_t\|^2$$

Let's try this in Matlab …
Why should Random Projections even work?! 

- Setting $K$ large is like getting $K$ samples.
Why should Random Projections even work?!

- Setting $K$ large is like getting $K$ samples.

- Specifically since we take $W$ to be random signs normalized by $\sqrt{K}$, for each $j \in [K]$, for any $\tilde{x}$ if $\tilde{y} = \tilde{x} W$, then

$$\mathbb{E}[\tilde{y}^2[j]] = \|\tilde{x}\|^2_2 / K$$
Why should Random Projections even work?!

- Setting $K$ large is like getting $K$ samples.
- Specifically since we take $W$ to be random signs normalized by $\sqrt{K}$, for each $j \in [K]$, for any $\tilde{x}$ if $\tilde{y} = \tilde{x} W$, then

$$E[\tilde{y}^2[j]] = \|\tilde{x}\|^2 / K$$

Hence we can conclude that

$$E\left[\sum_{j=1}^{K} \tilde{y}^2[j]\right] = \sum_{j=1}^{K} E[\tilde{y}^2[j]] = \sum_{j=1}^{K} \frac{\|\tilde{x}\|^2}{K} = \|\tilde{x}\|^2$$
Why should Random Projections even work?!

- Setting $K$ large is like getting $K$ samples.
- Specifically since we take $W$ to be random signs normalized by $\sqrt{K}$, for each $j \in [K]$, for any $\tilde{x}$ if $\tilde{y} = \tilde{x} W$, then

$$\mathbb{E}[\tilde{y}^2[j]] = \|\tilde{x}\|_2^2 / K$$

Hence we can conclude that

$$\mathbb{E}\left[\sum_{j=1}^{K} \tilde{y}^2[j]\right] = \sum_{j=1}^{K} \mathbb{E}[\tilde{y}^2[j]] = \sum_{j=1}^{K} \frac{\|\tilde{x}\|_2^2}{K} = \|\tilde{x}\|_2^2$$

This is like taking an average of $K$ independent measurements whose expectations are $\|\tilde{x}\|_2^2$.
Why should Random Projections even work?! 

For large $K$, not only true in expectation but also with high probability
Why should Random Projections even work?!

For large $K$, not only true in expectation but also with high probability

For any $\epsilon > 0$, if $K \approx \log (n/\delta) / \epsilon^2$, with probability $1 - \delta$ over draw of $W$, for all pairs of data points $i, j \in \{1, \ldots, n\}$,

$$(1 - \epsilon) \|y_i - y_j\|_2^2 \leq \|x_i - x_j\|_2 \leq (1 + \epsilon) \|y_i - y_j\|_2^2$$

Let's try on Matlab... This is called the Johnson-Lindenstrauss lemma or JL lemma for short.
Why should Random Projections even work?!

For large $K$, not only true in expectation but also with high probability.

For any $\epsilon > 0$, if $K \approx \log(n/\delta) / \epsilon^2$, with probability $1 - \delta$ over draw of $W$, for all pairs of data points $i, j \in \{1, \ldots, n\}$,

$$(1 - \epsilon) \|y_i - y_j\|_2^2 \leq \|x_i - x_j\|_2 \leq (1 + \epsilon) \|y_i - y_j\|_2^2$$

Let's try on Matlab...
Why should Random Projections even work?!

For large $K$, not only true in expectation but also with high probability.

For any $\epsilon > 0$, if $K \approx \log (n/\delta) / \epsilon^2$, with probability $1 - \delta$ over draw of $W$, for all pairs of data points $i, j \in \{1, \ldots, n\}$,

$$(1 - \epsilon) \| y_i - y_j \|_2^2 \leq \| x_i - x_j \|_2 \leq (1 + \epsilon) \| y_i - y_j \|_2^2$$

Lets try on Matlab . . .

This is called the Johnson-Lindenstrauss lemma or JL lemma for short.
Why is this so Ridiculously Magical?

\[ n = 1000 \]

\[ d = 1000 \]
If we take $K = 69.1/\epsilon^2$, with probability 0.99 distances are preserved to accuracy $\epsilon$. 

**Why is this so Ridiculously Magical?**
Why is this so Ridiculously Magical?

If we take $K = 69.1/\epsilon^2$, with probability $0.99$ distances are preserved to accuracy $\epsilon$. 

$n = 1000$

d = 10000
If we take $K = 69.1/\epsilon^2$, with probability 0.99 distances are preserved to accuracy $\epsilon$. 

$n = 1000$

$d = 1000000$
Two View Dimensionality Reduction

- Data comes in pairs $(x_1, x'_1), \ldots, (x_n, x'_n)$ where $x_t$'s are $d$ dimensional and $x'_t$'s are $d'$ dimensional

- Goal: Compress say view one into $y_1, \ldots, y_n$, that are $K$ dimensional vectors
  - Retain information redundant between the two views
  - Eliminate “noise” specific to only one of the views
Canonical Correlation Analysis
Canonical Correlation Analysis

+ Age
+ Gender
+ Angle

x
y
z
Canonical Correlation Analysis

[Diagram of people sitting on steps with axes labeled x, y, z, and additional variables Age, Gender, Angle]
Audio might have background sounds uncorrelated with video

Video might have lighting changes uncorrelated with audio

Redundant information between two views: the speech
Method A and Method B are both equally good feature extraction techniques.

Concatenating the two features blindly yields large dimensional feature vector with redundancy.

Applying techniques like CCA extracts the key information between the two methods.

Removes extra unwanted information.
How do we get the right direction? (say $K = 1$)

Age + Gender

Angle
Which Direction to Pick?

View I

View II
Which Direction to Pick?

View I

View II
Which Direction to Pick?

View I

View II
Which Direction to Pick?

View I

View II
Which Direction to Pick?

PCA direction
Which Direction to Pick?

Direction has large covariance
How do we pick the right direction to project to?
Say $\mathbf{w}_1$ and $\mathbf{v}_1$ are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

$$
\frac{1}{n} \sum_{t=1}^{n} \left( y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1] \right) \cdot \left( y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1] \right)
$$

where $y_t[1] = \mathbf{w}_1^\top \mathbf{x}_t$ and $y_t'[1] = \mathbf{v}_1^\top \mathbf{x}_t'$
What is the problem with the above?
Say \( \frac{1}{n} \sum_{t=1}^{n} x_t[2] \cdot x'_t[2] > 0 \)

Scaling up this coordinate we can blow up covariance...
Why not Maximize Covariance

Say \( \frac{1}{n} \sum_{t=1}^{n} x_t[2] \cdot x'_t[2] > 0 \)

Scaling up this coordinate we can blow up covariance
Why not Maximize Covariance

Relevant information

Say $\frac{1}{n} \sum_{t=1}^{n} x_t[2] \cdot x'_t[2] > 0$

Scaling up this coordinate we can blow up covariance
Say \( w_1 \) and \( v_1 \) are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

\[
\frac{1}{n} \sum_{t=1}^{n} (y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1]) \cdot (y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1])
\]

\[
\sqrt{\frac{1}{n} \sum_{t=1}^{n} (y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1])^2} \cdot \sqrt{\frac{1}{n} \sum_{t=1}^{n} (y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1])^2}
\]
Basic Idea of CCA

- Normalize variance in chosen direction to be constant (say 1)
- Then maximize covariance
- This is same as maximizing “correlation coefficient”
Covariance vs Correlation

\[ \text{Covariance}(A, B) = E[(A - E[A]) \cdot (B - E[B])] \]

Depends on the scale of \( A \) and \( B \). If \( B \) is rescaled, covariance shifts.

\[ \text{Correlation}(A, B) = \frac{E[(A - E[A]) \cdot (B - E[B])]}{\sqrt{\text{Var}(A)} \sqrt{\text{Var}(B)}} \]

Scale free.
Say \( w_1 \) and \( v_1 \) are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

\[
\frac{1}{n} \sum_{t=1}^{n} \left( y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1] \right) \cdot \left( y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1] \right)
\]

where \( y_t[1] = w_1^\top x_t \) and \( y_t'[1] = v_1^\top x_t' \)
Say \( w_1 \) and \( v_1 \) are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

\[
\frac{1}{n} \sum_{t=1}^{n} \left( y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1] \right) \cdot \left( y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1] \right)
\]

s.t. \( \frac{1}{n} \sum_{t=1}^{n} \left( y_t[1] - \frac{1}{n} \sum_{t=1}^{n} y_t[1] \right)^2 = \frac{1}{n} \sum_{t=1}^{n} \left( y_t'[1] - \frac{1}{n} \sum_{t=1}^{n} y_t'[1] \right)^2 = 1 \)

where \( y_t[1] = w_1^\top x_t \) and \( y_t'[1] = v_1^\top x_t' \)
Hence we want to solve for projection vectors $\mathbf{w}_1$ and $\mathbf{v}_1$ that

\[
\text{maximize } \frac{1}{n} \sum_{t=1}^{n} \mathbf{w}_1^\top (\mathbf{x}_t - \mu) \cdot \mathbf{v}_1^\top (\mathbf{x}_t' - \mu') \\
\text{subject to } \frac{1}{n} \sum_{t=1}^{n} (\mathbf{w}_1^\top (\mathbf{x}_t - \mu))^2 = \frac{1}{n} \sum_{t=1}^{n} (\mathbf{v}_1^\top (\mathbf{x}_t' - \mu'))^2 = 1
\]

where $\mu = \frac{1}{n} \sum_{t=1}^{n} \mathbf{x}_t$ and $\mu' = \frac{1}{n} \sum_{t=1}^{n} \mathbf{x}_t'$
Hence we want to solve for projection vectors $w_1$ and $v_1$ that

maximize $w_1^\top \Sigma_{1,2} v_1$

subject to $w_1^\top \Sigma_{1,1} w_1 = v_1^\top \Sigma_{2,2} v_1 = 1$
Hence we want to solve for projection vectors $\mathbf{w}_1$ and $\mathbf{v}_1$ that

\[
\begin{align*}
\text{maximize} & \quad \mathbf{w}_1^\top \Sigma_{1,2} \mathbf{v}_1 \\
\text{subject to} & \quad \mathbf{w}_1^\top \Sigma_{1,1} \mathbf{w}_1 = \mathbf{v}_1^\top \Sigma_{2,2} \mathbf{v}_1 = 1
\end{align*}
\]

\[
\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} = \text{COV} \left( \begin{pmatrix} X \\ X' \end{pmatrix} \right)
\]
\[ W_1 = \text{eigs}(\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}, K) \]

\[ W_2 = \text{eigs}(\Sigma_{22}^{-1}\Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}, K) \]
\[ \tilde{x}_t = x_t x'_t \]

the dimensional concatenated vectors.

Calculate covariance matrix of the joint data points

\[ \mathcal{C} = \begin{pmatrix} \mathcal{C}_1, 1 \\ \mathcal{C}_1, 2 \\ \mathcal{C}_2, 1 \\ \mathcal{C}_2, 2 \end{pmatrix} \]

Calculate \[ \mathcal{C}^{-1} \]

The top \( K \) eigen vectors of this matrix give us projection matrix for view I.

Calculate \[ \mathcal{C}^{-1} \]

The top \( K \) eigen vectors of this matrix give us projection matrix for view II.
CCA Algorithm

1. \( \mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & \mathbf{X}_2 \\ \end{pmatrix} \)

\( \mathbf{X} \) is the \( d_1 + d_2 \) dimensional concatenated vectors.

Calculate covariance matrix of the joint data points

\( \mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22} \end{pmatrix} \)

Calculate \( \mathbf{\Sigma}^{-1} \). The top \( K \) eigen vectors of this matrix give us projection matrix for view I.

Calculate \( \mathbf{\Sigma}^{-1} \). The top \( K \) eigen vectors of this matrix give us projection matrix for view II.
CCA Algorithm

1. \[ X = \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix} \]

2. \[ \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} = \text{cov}(\mathbf{X}) \]
CCA Algorithm

1. \( X = \begin{pmatrix} \mathbf{X}_1 & \mathbf{X}_2 \end{pmatrix} \)

2. \( \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} = \text{cov} \left( \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix} \right) \)

3. \( W_1 = \text{eigs} \left( \begin{pmatrix} \Sigma_{11}^{-1} & \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21} \\ \Sigma_{21} \Sigma_{22}^{-1} \Sigma_{12} & \Sigma_{22}^{-1} \Sigma_{21} \end{pmatrix}, K \right) \)
CCA Algorithm

1. \[ X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \]

2. \[ \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} = \text{cov}(X) \]

3. \[ W_1 = \text{eigs}(\Sigma_{11}^{-1} \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21}, K) \]

4. \[ Y_1 = (X - \mu_1) \times W_1 \]