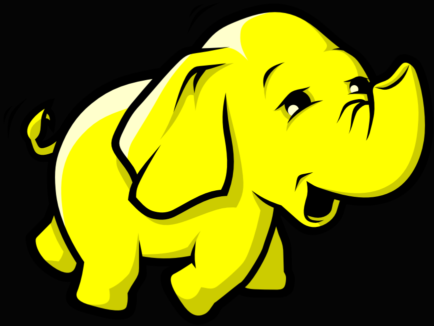
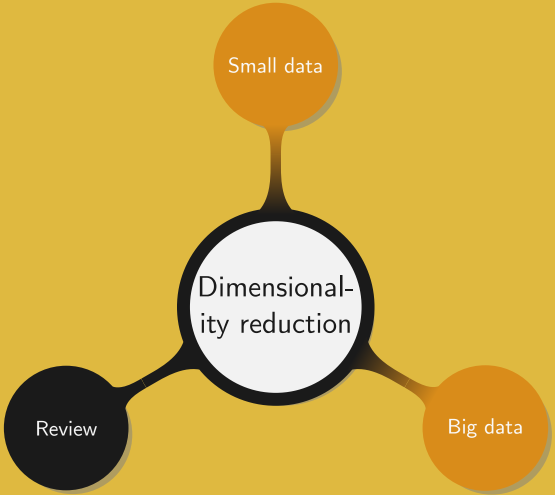


Methods and tools for big data

4. Dimensionality reduction

Manuel – Summer 2023

Chapter organisation



Statistical strategy

Before the big data age:

- Extract a sample from a large population, e.g. phone survey
- Derive measurements on the population based on the sample
- Classify the population with respect to the measurements
- Take decisions based on some given goals

Limitations:

- Not all people in a same category behave similarly
- Goals must be general and cannot be personalised

Statistical reminders

Given a dataset $X = [X_1, \cdots, X_n]$, we define the

- *Median*: which corresponds to the "middle" value
- *Mean*: $\bar{X} = \sum_{i=1}^{n} \frac{X_i}{n}$
- *Standard deviation* as the average distance between X_i and $\bar{X}$:
  - It represents the spread of the dataset
  - It is given by $\sigma = \sqrt{\frac{\left(\sum_{i=1}^{n} X_i - \bar{X}\right)^2}{n-1}}$
- *Variance* which also measures the spread and is given by σ^2

Those measures are one-dimensional

Statistical reminders

Given a dataset $[X_1, X_2] = \big[[X_{1,1}, \cdots, X_{1,n}], [X_{2,1}, \cdots, X_{2,n}]\big]$, *covariance*:

- Is given by
$$\sigma_{X,Y} = \frac{\sum_{i=1}^{n}(X_{1,i} - \bar{X_1})(X_{2,i} - \bar{X_2})}{n-1}$$
- Verifies $\sigma_{X_1,X_2} = \sigma_{X_2,X_1}$ and $\sigma_{X,X} = \sigma^2$
- Evaluates how much two dimensions vary from the mean with respect to each other
- Provides information on whether both dimensions "increase together"

Given n variables $X_1, \cdots, X_n$, their covariance matrix is given by

$$\begin{pmatrix} \sigma_{X_1,X_1} & \cdots & \sigma_{X_1,X_n} \\ \vdots & \ddots & \vdots \\ \sigma_{X_n,X_1} & \cdots & \sigma_{X_n,X_n} \end{pmatrix}$$

Algebraic reminders

Let $\mathcal{B}$ be a basis of a vector space V. For $M \in \mathcal{M}_n(\mathbb{K})$, $\lambda \in \mathbb{K}$ is an *eigenvalue* of M if and only there exists an *eigenvector* $X \in \mathcal{M}_{n,1}(\mathbb{K})$ such that $MX = \lambda X$.

Important properties related to eigenvectors and eigenvalues:

- If M has rank n then it has n non-zero eigenvalues
- If M has n eigenvalues, distinct two by two, then it is diagonalizable
- M is diagonalizable if and only if its eigenvectors form an orthogonal basis $\mathcal{B}'$
- $M = PDP^{-1}$, where D is a diagonal matrix featuring the eigenvalues of M on its diagonal, and P is the transition matrix from $\mathcal{B}'$ into $\mathcal{B}$
- For any $k \in \mathbb{N}$, $M^k = PD^kP^{-1}$

Algebraic reminders

The *characteristic polynomial* of M is defined as

$$\begin{aligned} \chi_M : \mathbb{K} &\longrightarrow \mathbb{K} \\ \lambda &\longmapsto \det(M - \lambda\, \mathrm{I}_n). \end{aligned}$$

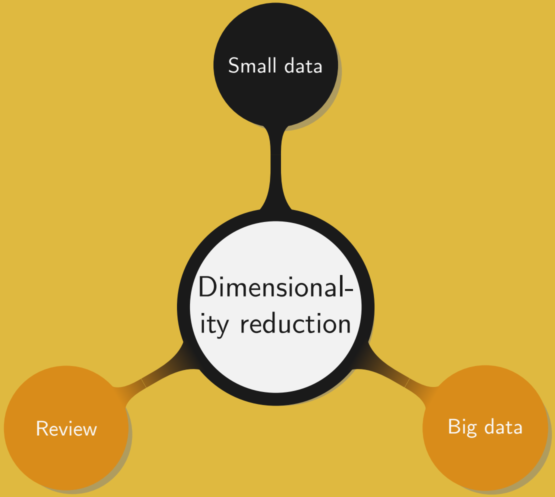
Properties of χ_M:

- If λ is an eigenvalue of M, then it is a root χ_M
- M if diagonalizable if and only if χ_M splits on $\mathbb{K}$ and the dimension of the eigenspace associated to each eigenvalue λ is equal to the multiplicity of λ

General remarks:

- Not all matrices are diagonalizable
- Not all square matrices are diagonalizable
- In general the determinant is "complicated" to compute
- A matrix that is not invertible is called *singular*

Chapter organisation



I admire the elegance of your method of computation; it must be nice to ride through these fields upon the horse of true mathematics while the like of us have to make our way laboriously on foot.

Albert Einstein

Principal component analysis

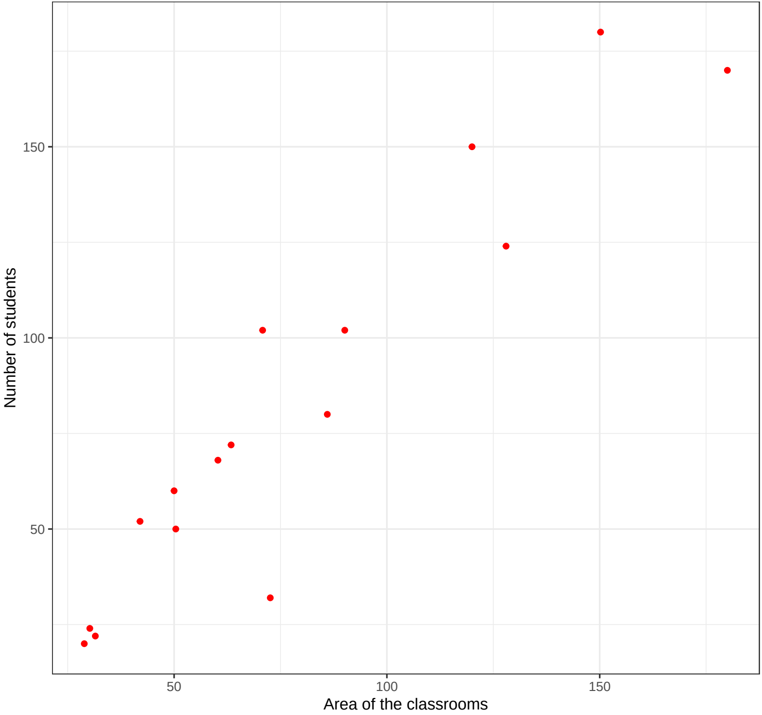
Basic idea behind Principal Component Analysis (PCA):

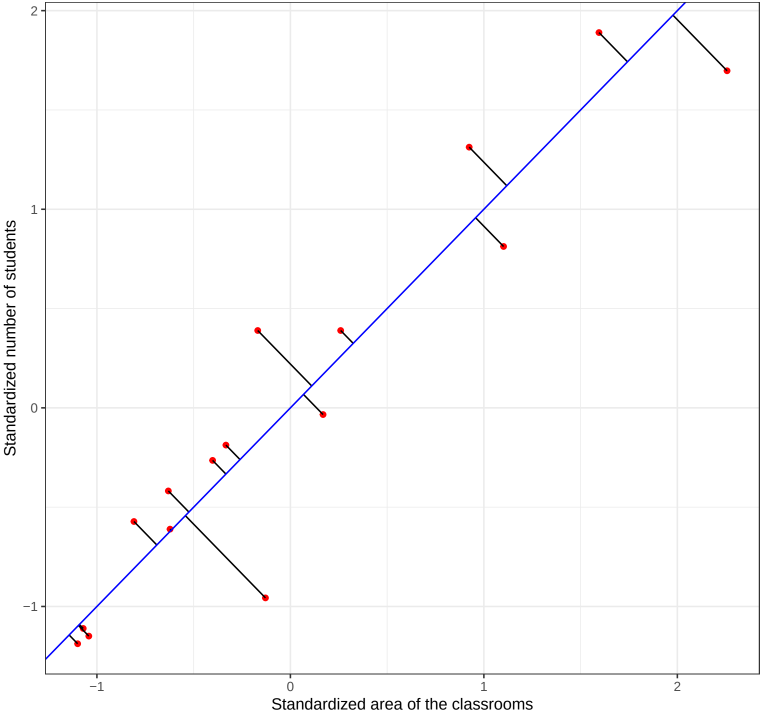
- Provides the best "perspective" that emphasises similarities and differences in the data
- This new perspective combines the original "characteristics" in order to best summarize the data

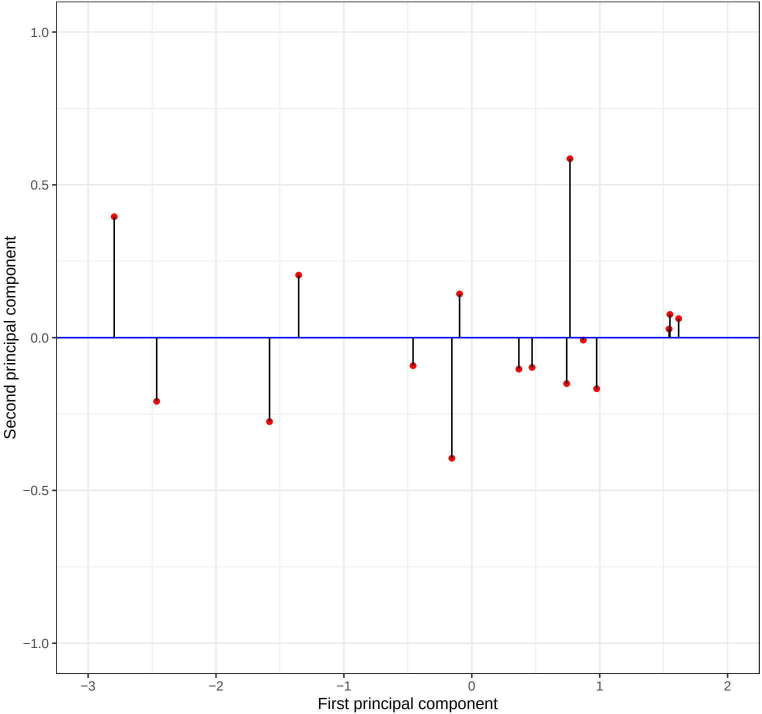
Example. Data that can be collected about teaching classrooms:

- The area
- The number of students who can seat in
- The number of blackboards
- The number of desktop computers

Which are useful, useless, or redundant?


Number of students
150
100
50
50
100
150
Area of the classrooms


2
1
0
−1
−1
0
1
2
Standardized number of students
Standardized area of the classrooms


1.0
0.5
0.0
−0.5
−1.0
Second principal component
−3
−2
−1
0
1
2
First principal component

Principal component analysis

Process:

1. Standardize the range of the variables:
   - Prevents large differences in the range of the variables
   - Ensures all variables "contribute equally"
   - Compute

$$Z_{i,j} = \frac{X_{i,j} - \bar{X}_i}{\sigma_{X_i}}$$

2. Determine the "relationship" between the variables:
   - Find correlations between the variables
   - Construct the covariance matrix over all the variables
3. Identify the principal components:
   - Compute the eigenvalues and eigenvectors of the covariance matrix
   - Reorder the eigenvalues in non-increasing order

Numerical stability

The stability of a method defines how it "reacts" to small perturbations

Since $M = PDP^{-1}$ we have $D = P^{-1}MP$, and for a small perturbation δM we get

$$D + \delta D = P^{-1}(M + \delta M)P.$$

This yields $\delta D = P^{-1}\delta MP$, which in term of norms translates as

$$\|\delta D\| \leq \|P^{-1}\|\|P\|\|\delta M\|, \tag{4.1}$$

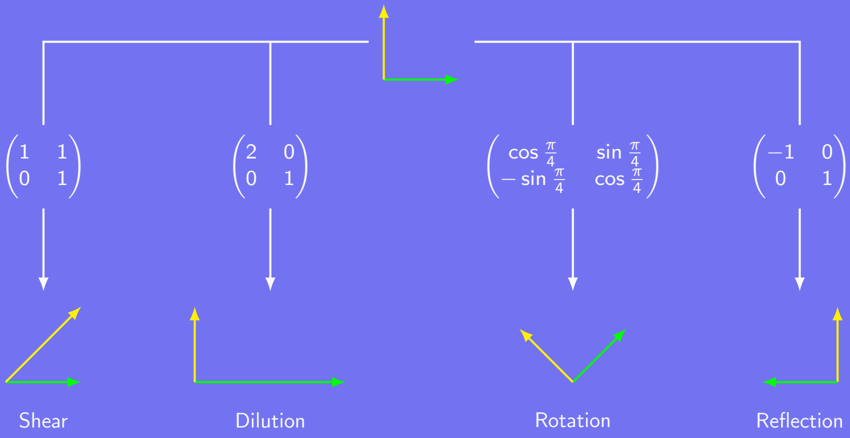
where the p-norm of an $m \times n$ matrix $A = (a_{i,j})_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}$ is defined as

$$\|A\|_p = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{i,j}|^p\right)^{\frac{1}{p}}.$$

Equation (4.1) means that a perturbation $\|\delta M\|$ might be magnified by a factor as large as $\|P^{-1}\|\|P\|$.

Linear transformations

For the sake of simplicity we consider the dimension 2 case



$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\frac{\pi}{4} & \sin\frac{\pi}{4} \\ -\sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{pmatrix} \quad \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$$

Singular value decomposition

Let $\{v_1, v_2\}$ be an orthonormal basis, and M be the matrix of a linear transformation. Then for two unit vectors u_1 and u_2 we obtain

$$Mv_1 = u_1\sigma_1 \quad \text{and} \quad Mv_2 \ = u_2\sigma_2,$$

with σ_1 and σ_2 in $\mathbb{R}$. For a vector $x = \langle x, v_1\rangle v_1 + \langle x, v_2\rangle v_2$, we have

$$\begin{aligned} Mx &= \langle x, v_1\rangle Mv_1 + \langle x, v_2\rangle Mv_2 \\ &= \langle x, v_1\rangle u_1\sigma_1 + \langle x, v_2\rangle u_2\sigma_2. \end{aligned}$$

Recalling that $\langle x, v_1\rangle = x^\top v_1 = v_1^\top x$, we get $Mx = u_1\sigma_1 v_1^\top x + u_2\sigma_2 v_2^\top x$, which yields $M = u_1\sigma_1 v_1^\top + u_2\sigma_2 v_2^\top$. In term of matrices this translates into

$$M = \begin{pmatrix} u_1 & u_2 \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \begin{pmatrix} v_1^\top \\ v_2^\top \end{pmatrix}.$$

Singular value decomposition

More generally, if M is an $m \times n$ real matrix we can write $M = U\Sigma V^\top$, where both $U = \begin{pmatrix} u_1 & \cdots & u_m \end{pmatrix}$ and $V = \begin{pmatrix} v_1 & \cdots & v_n \end{pmatrix}$ are rotation matrices, and Σ is diagonal.

Size of the matrices:

- M: $m \times n$
- U: $m \times m$
- Σ: $m \times n$
- V: $n \times n$

The elements on the diagonal of Σ are called *singular values*, the columns of U *left singular vectors*, and the rows of $V^\top$ *right singular vectors*.

If $U\Sigma V^\top$ is an SVD for an $m \times n$ matrix X of rank r:

- Σ has exactly r strictly positive elements which are the square roots of the r eigenvalues of $X^\top X$, with corresponding multiplicities
- The columns of V are eigenvectors of $X^\top X$
- The columns of U are eigenvectors or $XX^\top$

Singular value decomposition

General remarks on the SVD:

- SVD is not unique and exists for any matrix, whether invertible or not
- Singular values can be re-ordered as long as their corresponding singular vectors are re-arranged accordingly
- If $U\Sigma V^\top$ is an SVD for $X^\top$, then $V\Sigma^\top U^\top$ is an SVD for X

As U and V are rotation matrices, they are orthogonal, i.e. $U^\top U = \mathrm{I}_m$ and $V^\top V = \mathrm{I}_n$. In particular, referring to slide 4.13, we see that in term of stability a small perturbation gets increased by a factor:

- $\|P^{-1}\|\|P\|$, for eigen decomposition
- $\|U^{-1}\|\|(V^\top)^{-1}\| = \|U^\top\|\|V\| = 1$, for SVD

QR decomposition

Any $X \in \mathcal{M}_{m,n}(\mathbb{K})$, with linearly independent columns, can be written $X = QR$ where:

- $Q \in \mathcal{M}_{m,n}(\mathbb{K})$ is orthogonal
- $R \in \mathcal{M}_{n,n}(\mathbb{K})$ is upper triangular

Remark. If X is invertible, then so is R.

Various approaches can be applied to compute the QR decomposition of X. The most famous one, Gram-Schmidt is unfortunately unstable. The two other most common options are based on Givens rotations and on Householder reflections. The former is slower but simpler to parallelize.

Given a vector u, a *Householder reflection* is a linear transformation $H_u = I - \frac{2uu^\top}{u^\top u}$. For instance in an orthonormal basis $\{u, v\}$ we can write $w = c_1 u + c_2 v$, with $c_1 = \frac{\langle w,u\rangle}{\|u\|_2^2}$ and $c_2 = \frac{\langle w,v\rangle}{\|v\|_2^2}$.

QR decomposition

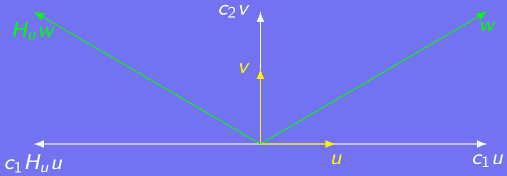
Applying H_u to w we get

$$H_u w = \left(I - \frac{2uu^\top}{u^\top u} \right) (c_1 u + c_2 v) = c_1 u + c_2 v - 2\frac{uu^\top}{u^\top u}(c_1 u + c_2 v)$$

$$= c_1 u + c_2 v - 2c_1 = -c_1 u + c_2 v - 2c_2 \frac{u}{u^\top u} \langle u, v \rangle$$

$$= -c_1 u + c_2 v.$$

This example can be visually represented as a reflection about the plane orthogonal to u and v.



QR decomposition

Householder reflections can be used to obtain the QR decomposition of a matrix X. More specifically, *Householder reflection theorem* states that for two vectors x and y with similar norm, there exists an orthogonal matrix Q such that $y = Qx$. This provides a method for iteratively constructing R and Q.

Idea on how to apply the above discussion:

- Determine an orthogonal matrix Q_1 such that $Q_1 c_1 = (\gamma_1, 0, \cdots, 0)^\top$, where γ_1 has similar norm as c_1. At this stage we have

$$R_1 = \left(\begin{array}{c|c} \gamma_1 & * \cdots\cdots * \\ \hline 0 & \\ \vdots & X_1 \\ \vdots & \\ 0 & \end{array}\right).$$

SVD and QR decomposition

- Determine an orthogonal matrix $\tilde{Q}_2$ such that $\tilde{Q}_2 \tilde{c}_1 = (\gamma_2, 0, \cdots 0)^\top$. Then define $Q_2 = diag(\mathrm{Id}_1, \tilde{Q}_2)$ and compute

$$R_2 = Q_1 R_1 = \left(\begin{array}{cc|c} \gamma_1 & \multicolumn{2}{c}{* \cdots\cdots\cdots\cdots *} \\ 0 & \gamma_2 & * \cdots\cdots * \\ \hline 0 & 0 & \\ \vdots & \vdots & X_2 \\ 0 & 0 & \end{array}\right).$$

- Repeat the above process $t = \min(m-1, n)$ times and obtain $R = Q_t Q_{t-1} \cdots Q_2 Q_1 X$. By orthogonality $Q^{-1} = Q^\top$, yielding

$$X = Q_1^\top \cdots Q_t^\top R = QR.$$

Application to the SVD of X:

1. Write $X = QR$
2. Perform SVD on $R = U_1 \Sigma V^\top$
3. Obtain the SVD for

$$X = QU_1 \Sigma V^\top = U \Sigma V^\top$$

PCA and SVD

Let X be an $m \times n$ matrix representing a dataset.

$$\left.\underbrace{\qquad X \qquad}_{n\text{ attributes}}\right\}\, m \text{ cases}$$

PCA requires to determine the eigenvalues of the covariance matrix $C = X^\top X$ (slide 4.12). Now observe that using the SVD decomposition $X = U\Sigma V^\top$,

$$\begin{aligned} X^\top X &= \left(U\Sigma V^\top\right)^\top \left(U\Sigma V^\top\right) \\ &= V\Sigma^\top U^\top U\Sigma V^\top \\ &= V\Sigma^2 V^\top. \end{aligned}$$

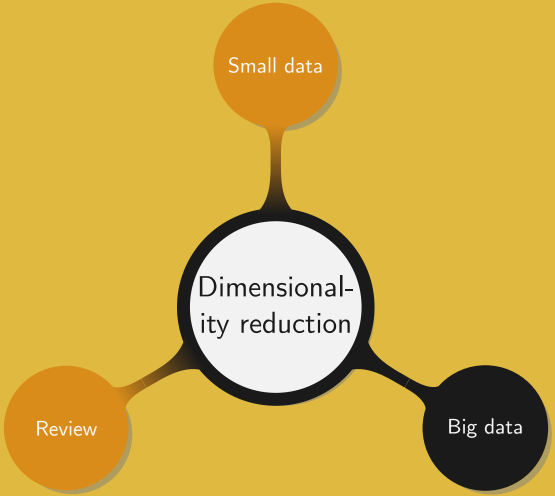
Principal components are given by $XV = U\Sigma V^\top V = U\Sigma$ and the singular values are the square of the eigenvalues.

Dimensionality reduction

Given a dataset one wants to:

- Retain a maximum of information in a minimum amount of space
- Run PCA using:
    - The covariance matrix:
        - Might be a bit faster but is unstable
        - Eigenvalues correspond to the variances of the principal components
    - SVD:
        - Might be a bit slower but stable
        - The square root of the singular values corresponds to the variances of the principal components
- Only keep the *k* largest principal components by truncating the matrices

Chapter organisation



Distributed systems

An $m \times n$ matrix X can be:

- Dense
- Sparse
- Square
- Tall and skinny
- Short and fat

Ways to represent a matrix over many nodes:

- By row or column
- By elements
- By blocks

Remarks.

- The matrix does not fit anymore in memory
- The shape and representation of the matrix impacts the speed

Tall and skinny

Setup:

- X does not fit in memory
- X has many more rows than columns ($m \gg n$)
- n^2 fits in memory on a single machine

When running PCA:

- Complexity of a full SVD: $\mathcal{O}(mn^2)$
- We only need the k most significant singular values and vectors
- The work needed is $\mathcal{O}(mk^2)$

Can we take advantage of the shape of X to be faster?

Tall and skinny

From slide 4.22, we know that

$$C = X^T X = V\Sigma^2 V^\top. \tag{4.2}$$

Also C can fit in memory since it has dimension $n \times n$ and by assumption a single machine has n^2 memory. Thus we *hope to compute* $X^\top X$, without any dependence on m. Then using equation 4.2 we can find the eigenvalues of C and retrieve V and Σ. The main challenge is to efficiently compute $X^\top X$, while ensuring the singular values of C are not significantly altered in the process.

Setup preparation:

- Store matrices row-by-row on disk
- Ensure all elements are in $[-1, 1]$, i.e. divided by the largest element

Tall and skinny

Simple MapReduce approach to compute $X^T X$:

- Mapper: for all pairs (x_{ij}, x_{ik}) on row i, return the $(key, value)$ pair
$$\left((c_j, c_k), x_{ij}x_{ik}\right),$$
where c_j and c_k correspond to columns j and k, respectively.
- Reducer: consider all the $(key, value)$ pairs $\left((c_i, c_j), \langle v_1, \cdots, v_R\rangle\right)$, where each of the v_k corresponds to a product of elements in X and R is the number of non-zero products, and compute $\sum_{i=1}^{R} v_i$.

Considerations on the complexity:

- Communication: *shuffle size* $\mathcal{O}(mL^2)$, with L the maximum number of non-zero elements on a row
- Overload on a single machine: *reduce-key complexity* $\mathcal{O}(m)$

Measuring similarity

Measuring the distance between several documents:

- Count the number of occurrence for all the words
- Compare the scores
- If documents feature many common words, then they are similar

Limitation:

- Take three documents d_1, d_2, and d_3, where d_1, d_2 are long and d_3 is an exert of d_1
- What is likely to happen?

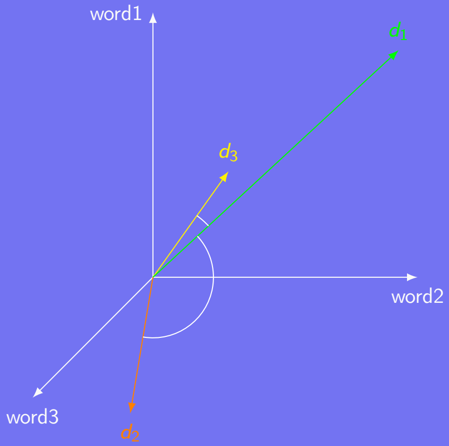
Cosine similarity

For two vectors d_i and d_j we define their *cosine similarity* as

$$\cos(d_i, d_j) = \frac{\langle d_i, d_j \rangle}{\|d_i\| \|d_j\|}.$$

Basic idea:

- The closer two documents, the smaller the angle
- The smaller the angle, the larger the cosine



More refined strategy:

- Classify words by meaning
- Consider the cosine similarity based on word semantic

Tall and skinny with cosine similarity

We fix γ, a parameter that can be adjusted when running MapReduce.

MapReduce using cosine similarity to compute $X^T X$:

- Mapper: for all pairs (x_{ij}, x_{ik}) on row i, return the $(key, value)$ pair

  $$\left((c_j, c_k), x_{ij} x_{ik}\right), \text{ with probability } \min\left(1, \frac{\gamma}{\|c_j\|\|c_k\|}\right),$$

  where c_j and c_k correspond to columns j and k, respectively.

- Reducer: consider all the $(key, value)$ pairs $\left((c_i, c_j), \langle v_1, \cdots, v_R\rangle\right)$, where each of the v_k corresponds to a product of elements in X and R is the number of non-zero products and proceed as follows.
  - If $\frac{\gamma}{\|c_j\|\|c_k\|} > 1$, then return $\frac{1}{\|c_j\|\|c_k\|} \sum_{i=1}^{R} v_i$
  - Otherwise, return $\frac{1}{\gamma} \sum_{i=1}^{R} v_i$

Tall and skinny with cosine similarity

What the reducer returns is in fact not $X^\top X$, but a matrix X' which contains the cosine similarities between the columns of X. Defining $\bar{v}_i$ to be v_i, if a $(key, value)$ pair is returned and 0 otherwise, this can be seen by looking at the expectation of an output

$$E\left(\frac{1}{\gamma}\sum_{i=1}^{R}\bar{v}_i\right) = \frac{1}{\gamma}P(\bar{v}_i = v_i)\sum_{i=1}^{R} v_i = \frac{1}{\|v_j\|\|v_k\|}\sum_{i=1}^{R} v_i.$$

To recover $X^\top X$ from the matrix X', it suffices to define a diagonal matrix D whose elements d_{ii} are exactly $\|c_i\|$, and then compute $X^\top X \approx DX'D$.

Using Latala's theorem it is possible to prove that when using cosine similarities to sample columns, singular values are preserved with "high enough" probability.

Tall and skinny with cosine similarity

Remarks.

- Since $\|c_i\|$ is used in the mapper it means the norm of all the columns must be pre-computed. This requires an all-to-all communication.
- The parameter γ can be adjusted depending on what is expected:
  - Preserve similar entries in $X^\top X$: $\gamma = \Omega\left(\frac{\log n}{s}\right)$, where s is the lowest cosine similarity
  - Preserve singular values of $X^\top X$: $\gamma = \Omega\left(\frac{n}{\varepsilon^2}\right)$, where ε is the relative error

Complexity, with h the smallest non-zero value after normalization:

- Shuffle size: $\mathcal{O}(nL\gamma/h^2)$
- Reduce key complexity: $\mathcal{O}(\gamma/h^2)$

Distributed QR

As explained in slide 4.21 first decomposing X into $X = QR$ can speed up the SVD of X. The basic idea related to Householder reflection theorem can in fact be extended into a “tiled QR decomposition” algorithm, which is applicable when the matrix X does not fit in memory.

In this algorithm, the tiles correspond to matrix blocks which can be stored and processed on different nodes.

Represent X in tall and skinny blocks, then start process:

- Obtain the QR decomposition of block $X_{k,k}$
- Adjust all the blocks $X_{k,j}$ on the right of $X_{k,k}$
- Adjust all the blocks $X_{i,k}$ below $X_{k,k}$ and the bocks $X_{i,j}$ on their right
- Repeat for all blocks on the diagonal

Looking at the big picture

Brief summary:

- Data has many variables
- Apply PCA to find a better perspective
- Ignore low contributions, only keep the most “prominent” dimensions
- Apply SVD in order to compute PCA
- Run further tasks based on the PCA “approximation”

Comments on PCA:

- It works well on small data as well as on tall and skinny big data
- It searches for a “best” solution

How much better is the best solution compared to a random solution?

Random projections

Construct a map preserving the pairwise distance between points

Setup and goals:

- Map m points $(u_i)_{1 \leq i \leq n}$ in $\mathbb{R}^n$ into m points in $\mathbb{R}^d$, with $d \ll n$
- Ensure that for all i, j,

$$\|v_i\|_2 \approx \|u_i\|_2 \text{ and } \|v_i - v_j\|_2 \approx \|u_i - u_j\|_2 \tag{4.3}$$

Johnson-Lindenstrauss lemma states that after a random projection, with high probability the distance between any two points is “distorted” by a factor at most $1 \pm \varepsilon$.

If $X' = \frac{1}{\sqrt{d}} XR$, where $R \in \mathbb{R}^{n \times d}$ is random with independent identically distributed entries and zero mean, then X' is much smaller and preserves all pairwise distances in expectation.

Random projections

Feeling behind the random projection idea:

- Project a random vector onto a fixed subspace:
    - Consider m points in $\mathbb{R}^n$ and fix k coordinates uniformly at random
    - Two vectors differing by only few coordinates can see their distance totally changed after projection
- Project a fixed vector onto a random subspace:
    - Consider m points in $\mathbb{R}^n$ and project them in a k-dimensional subspace
    - Two vectors differing by only few coordinates will "spread out" all coordinates and prevent missing out on "important" coordinates

Generation of the matrix $R = (r_{i,j})$:

- Different $r_{i,j}$ can impact the variance and the error tail bounds
- The $r_{i,j}$ are independent identically distributed with zero mean
- The $r_{i,j}$ are often selected following a symmetric distribution about zero with unit variance

Random projections

Remarks.

- A random projection is likely not a projection in the algebraic sense:
    - Most often: $R^2 \neq R$
    - Eigenvalues of R are not necessarily in $\{0, 1\}$
    - It is ε-close to a projection
- Requirement (4.3) does not extend to other norms
- The norm of a projected vector is $\sqrt{\frac{d}{n}}$ in expectation, with an error exponentially small in d
- Time complexity: $\mathcal{O}(mnd)$
- The projection can be made very sparse with little loss of accuracy and a major speedup

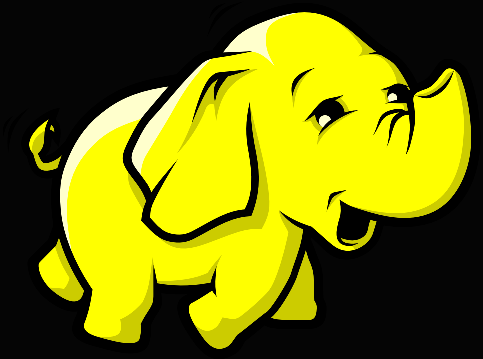
Dimensionality reduction for big data:

Preserving important structural properties of the data:

- Tall and skinny can be handle using SVD
- For other cases random projections are a good alternative
- PCA will ensure a good result, random projections might not
- Randomized PCA:
    - Randomly project X to obtain X'
    - Perform a QR decomposition on $X' = QR$
    - Compute $B = Q^\top X$
    - Run SVD on $B = U_1 \Sigma V^\top$
    - Approximate the SVD of X by $QQ^\top X = Q\left(U_1 \Sigma V^\top\right) = U\Sigma V^\top$
    - Complete PCA the usual way

Key points

- How to perform dimensionality reduction?
- What is the relationship between SVD and PCA?
- What is cosine similarity?
- How to deal with tall and skinny big data?
- What are random projections?



Thank you!