Singular Value Decomposition (SVD)

$X = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_m \\ | & | & & | \end{bmatrix}_{n\times m} = U \Sigma V^T = \begin{bmatrix} | & & | \\ u_1 & \cdots & u_n \\ | & & | \end{bmatrix}_{n\times n} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_m \\ \hline 0 & & \end{bmatrix}_{n\times m} \begin{bmatrix} | & & | \\ v_1 & \cdots & v_m \\ | & & | \end{bmatrix}^T_{m\times m}$

$\forall$ $(n\times m)$ matrices

x_i reshaped: n dim eg. pixels, $x_m \in \mathbb{R}^n$

t; Singular values; L.S.vectors; R.S.vectors

Mixture of all cols U to make a col X

"eigen faces | flow field"

"eigen time series"

$\sigma_1 \geq \cdots \sigma_n \geq 0$

Orthonormal basis for col(X) ; O.B. for row(X)

All ordered by impt, mag of σ_i & Unique SVD

Outerprod

$= \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_m u_m v_m^T + 0$

$= \hat{U}\hat{\Sigma}V^T$

"Economy SVD" $(n \gg m)$

Rank-1 matrix

$X = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_m u_m v_m^T$

If $\sigma_{r+1} \cdots \sigma_m$ all negligibly small, we truncate at rank r

$\therefore X = \begin{bmatrix} | & & | & | \\ u_1 & \cdots & u_r & \cdots u_n \\ | & & | & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_m \\ \hline 0 & & \end{bmatrix} \begin{bmatrix} - & v_1^T & - \\ & \vdots & \\ - & v_m^T & - \end{bmatrix} \approx \tilde{U}\tilde{\Sigma}\tilde{V}^T$

rank theorem: $r \leq m$ cols and assume $n \gg m$, $\therefore r < n$

best rank, r approximation

Now after truncation, no longer symmetrical U & V.

$\tilde{U}^T\tilde{U} = I_{r\times r} \neq \tilde{U}\tilde{U}^T$

ortho vectors

$\tilde{U}^T$ $r\times n$ $\cdot$ $\tilde{U}$ $n\times r$ $= I_{r\times r}$

BUT $\begin{bmatrix} | & & | \\ u_1 & \cdots & u_r \\ | & & | \end{bmatrix} \begin{bmatrix} - & u_1^T & - \\ & \vdots & \\ - & u_r^T & - \end{bmatrix} \neq I_{n\times r}$

$\tilde{U}$ $n\times r$

Eckard-Young Theorem [1936]

$\underset{\tilde{X} \text{ s.t } \text{rank}(\tilde{X}) = r}{\arg\min} \| X - \tilde{X} \|_F = \tilde{U}\tilde{\Sigma}\tilde{V}^T$

the best rank-r approx, $\tilde{X}$ that minimizes error

Frobenius Norm, F

eg. $\|A\|_F = \sqrt{\sum_{(i,j)} (A_{ij})^2}$, Take matrix A, reshape into 1 long vector, calc 2-norm of it

Unitary Transformation of U,V

$X = U\Sigma V^T = \hat{U}\hat{\Sigma}V^T$

where,

$UU^T = U^TU = I$

$VV^T = V^TV = I$

U, V unitary matrices

Unitary Transformation (Linear transformat^n)

↳ Preserve $\angle$s & $\|\vec{v}\|$, rotation

$\langle x, y \rangle = \langle Ux, Uy \rangle \quad \forall x, y \in \mathbb{R}^n$

* Orthogonal matrix:

$AA^T = A^TA = I$, real values

$\therefore A^{T*} = A^T$

Unitary matrix:

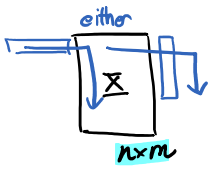
$AA^{T*} = A^{T*}A = I$

C Complex Conjugate Transpose

$a \pm bi \rightarrow a \mp bi$

If real, $bi = 0$

either



unit vectors, v

$\mathbb{R}^m$

* U & V is unitary, X isn't

XV Linear transformat^n (matrix multip.)

$\mathbb{R}^n$

unit vectors, u

u^X

$\mathbb{R}^m$

$\mathbb{R}^n$

Take, $X = U\Sigma V^T$ ← identity matrix here

· Orientat^n depends on U of X

· Axis length transformat^n d.o. $*\Sigma$ of X

Axis are the PC = Vol $\frac{X_c^T X_c}{n-1}$, if X centered already,

Vol X

· Compare to a transformat^n ↳ same mapping

$Av = R_2 R_y R_x \begin{bmatrix} \text{Scale}_z & 0 & 0 \\ 0 & \text{Scale}_y & 0 \\ 0 & 0 & \text{Scale}_x \end{bmatrix} I V$

rotat^n

SAME!

Eigendecomposition of Square Cross Pdt Matrices

$X^TX = V\Sigma U^T U \Sigma V^T$

$= V\Sigma^2 V^T$ ↳ I

$\Rightarrow (X^TX)V = V\Sigma^2$ $n\times n$ $n\times n$

↳ eigenvalues = Σ diagonal singular values^2

Similarly $(XX^T)U = U\Sigma^2$ $n\times n$ ↑ some eigenvalues for the nonzero.

$n\times n$!

· Manually do E.D of XX^T / X^TX or directly SVD to get U, Σ, V

BUT let's say if X is so large to store as a memory. (very rare)

Method of Snapshots (1987)

· Compute X^TX element by element,

compute → $V\hat{\Sigma}$ & X^TX m by m matrices

2 vectors @ a time

→ Solve $\hat{U}$ assuming $X = \hat{U}\hat{\Sigma}V^T$

where $\hat{U} = XV\hat{\Sigma}^{-1}$

Linear System, Pseudo A

$Ax = b$ linear system, solve x but now A^{-1} x exist $(n\times m)$

Using SVD: $Ax = b$

$\hat{U}\hat{\Sigma}V^Tx = b$ (Assume "E-SVD")

$V\hat{\Sigma}^{-1}\hat{U}^T\hat{U}\hat{\Sigma}V^Tx = V\hat{\Sigma}^{-1}\hat{U}^Tb$

$\tilde{x} = A^+b$

$\sim$:= approx

A^+ := moore-penrose (left) pseudo Inverse

So now $A\tilde{x} = b$ is true for underdetermined, but overdetermined it is not: $A\tilde{x} = \hat{U}\hat{\Sigma}V^T V\hat{\Sigma}^{-1}\hat{U}^Tb = \hat{U}\hat{U}^Tb$

$\neq I$ if not $A\tilde{x} = b$

$\hat{U}\hat{U}^Tb$:= project^n of b on the span$(\hat{U})$, ∵ cols $\hat{U}$ span cols A,

$\therefore$ project^n of b on the span(A)

a) Underdetermined (short fat A)

$A \quad x = b$

· * Generally, ∞ Sol^n

· $\tilde{x}$ results in $\min \|\tilde{x}\|_2$ st $A\tilde{x} = b$ (min 2-norm sol^n)

eg. represents energy ↳ cheapest way of $A \otimes x$

b) Overdetermined (tall skinny A)

$A \quad x = b$

* Generally, 0 Sol^n

· $\min \|A\tilde{x} - b\|_2$ (least square error sol^n)

least error btw $A\tilde{x}$ & b fit

· To solve a linear system of equation, we know that b must be in the span of the columns of A

eg. $\begin{bmatrix} 3 & \frac{1}{2} \\ 1 & 2 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \end{bmatrix}$

$= -1\begin{bmatrix} 3 \\ 1 \end{bmatrix} + 2\begin{bmatrix} \frac{1}{2} \\ 2 \end{bmatrix}$

Linear Algebra

If $A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$, there exists no solution

$A = \begin{bmatrix} \frac{1}{2} & 4 \\ 3 & 6 \end{bmatrix}$, $b = \begin{bmatrix} 2 \\ 6 \end{bmatrix}$, $x = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$, a solution!

$\text{col}(A) \times \ker(A^T) = \mathbb{R}^n$

↳ So it really depends on the col(A) = col$(\hat{U})$ ··· range ↳ make up the subspace of A

↳ Likewise if b exists in the orthogonal complement, ker(A^T), then NO sol^n

↳ null(A) / ker(A) := set of x that gets sent to $\vec{0}$, one trivial sol^n $= \vec{0}$

Hence there exist a sol^n iff b is in the col(A), but if dim(null(A)) $\neq 0$

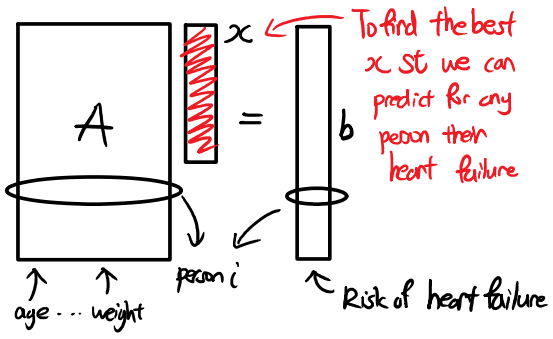
kernel is not trivial, means that there are some vectors that map to $\vec{0}$

If nullity$(A) \neq 0$, (dim) no. of vectors send to $\vec{0} \neq 0$, nullspace no longer trivial, large, then there exists some x_{null} vectors, $\therefore Ax_{sol^n} = A(x_{sol^n} + x_{null}) = b$

another x_{sol^n}

∴ Infinite Sol^n

Application: Linear Regression



To find the best x st we can predict for any person their heart failure

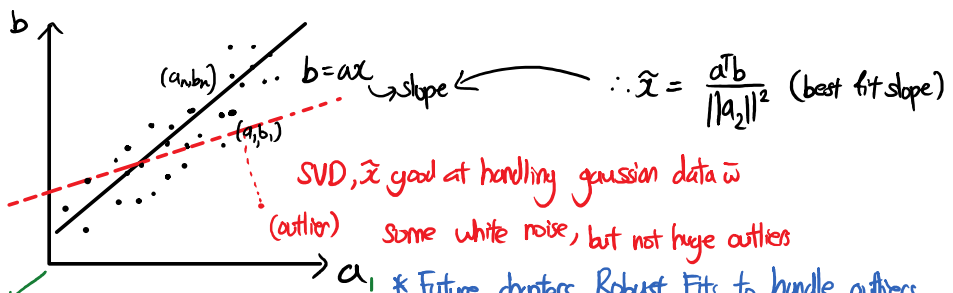
Asume general overdetermined case,

$\tilde{x} = A^+b = V\Sigma^{-1}U^Tb$

eg. Simplify to one factor a

$\begin{bmatrix} a_1 \\ \vdots \end{bmatrix} \begin{bmatrix} x_1 \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \end{bmatrix}$

$U = \frac{a}{\|a\|_2}$, $\Sigma = \|a\|_2$, $V = 1$



$\therefore \tilde{x} = \frac{a^Tb}{\|a_2\|^2}$ (best fit slope)

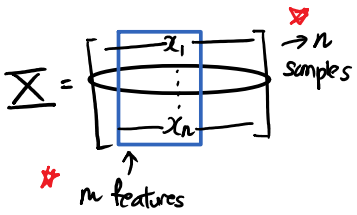
SVD, $\tilde{x}$ good at handling gaussian data w some white noise, but not huge outliers

* Future chapters, Robust Fits to handle outliers

So if $A = [a_1 a_2]$ then $\tilde{x}$:= best fit plane

Applicat^n : PCA

- Statistical Interpretation of SVD
- Data-driven hierarchical coordinate system that are based of principal components (orthogonal) uncorrelated axes that have maximal correlation to each measurement/ feature



Covariance Matrix of $\mathbb{X}$, $C_{m\times m} = \frac{1}{n-1}\mathbb{X}_c^T\mathbb{X}_c$

$\frac{1}{n}$ $\begin{bmatrix} x_{11}-\bar{x}_1 & \cdots & x_{n1}-\bar{x}_1 \\ \vdots & & \vdots \\ x_{1m}-\bar{x}_m & \cdots & x_{nm}-\bar{x}_m \end{bmatrix}$ $\begin{bmatrix} x_{11}-\bar{x}_1 & \cdots & x_{1m}-\bar{x}_m \\ \vdots & & \vdots \\ x_{n1}-\bar{x}_1 & \cdots & x_{nm}-\bar{x}_m \end{bmatrix}$, $\bar{x}_j$ = Col mean $= \frac{1}{n}\sum_{i=1}^{n} x_{ij}$

Centered Mean Data

↑ Col standardized

Correlation Matrix of $\mathbb{X}$, $S_{m\times m} = \frac{1}{n}\mathbb{X}_s^T\mathbb{X}_s$

$\frac{1}{n}$ $\begin{bmatrix} \frac{x_{11}-\bar{x}_1}{s_1} & \cdots & \frac{x_{n1}-\bar{x}_1}{s_1} \\ \vdots & & \vdots \\ \frac{x_{1m}-\bar{x}_m}{s_m} & \cdots & \frac{x_{nm}-\bar{x}_m}{s_m} \end{bmatrix}$ $\begin{bmatrix} \frac{x_{11}-\bar{x}_1}{s_1} & \cdots & \frac{x_{1m}-\bar{x}_m}{s_m} \\ \vdots & & \vdots \\ \frac{x_{n1}-\bar{x}_1}{s_1} & \cdots & \frac{x_{nm}-\bar{x}_m}{s_m} \end{bmatrix}$, $\bar{x}_j$ = Col mean, $s_\cdot$ = Col Stddeviat^n $= \frac{1}{n}\sum_{i=1}^{n} x_{ij}$

Std Normal Dist. Data

Why Standardize?

- If the cols are not of the same scale, some might have a larger variance Hence for the n datapoints, the spread is larger. ∴ PCA would be biased

Principal Component Matrix, $P = \mathbb{X}_c V$

(Project $\mathbb{X}$ on PC)

Principal Components

Right eigenvectors of $C = \frac{1}{n-1}\mathbb{X}_c^T\mathbb{X}_c$

same as the eigenvectors of $\mathbb{X}_c^T\mathbb{X}_c$

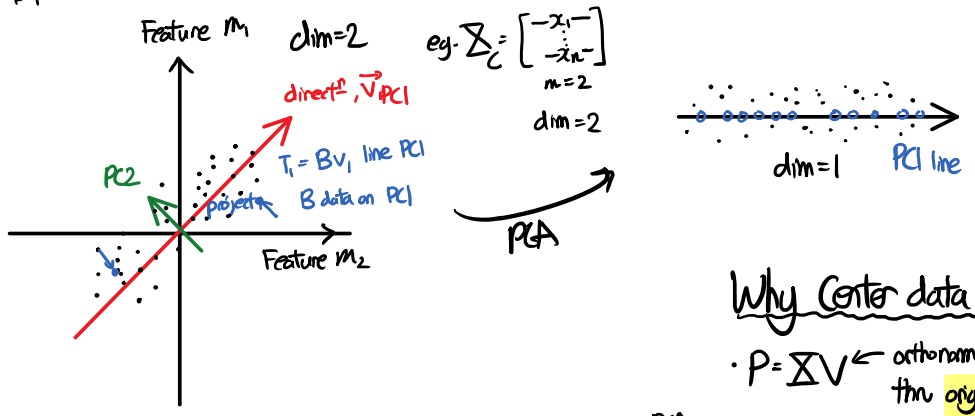
* Note eigendecomposit^n, eigenvalues of $C = \frac{1}{n-1}$ of that of $\mathbb{X}_c^T\mathbb{X}_c$

So we can do a SVD of $\mathbb{X}_c^T\mathbb{X}_c$ to get V

PC1: $P_1 = \mathbb{X}_c V_1$, reflects the direct^n of largest variance in data & has largest variance cyst all the normalized linear combi of cols of $\mathbb{X}_c$. Subsequent P_i, cols of P ortho to each other, decreasing variance captured

Since variance of data along principal component associated w eigenvalues of covariance matrix

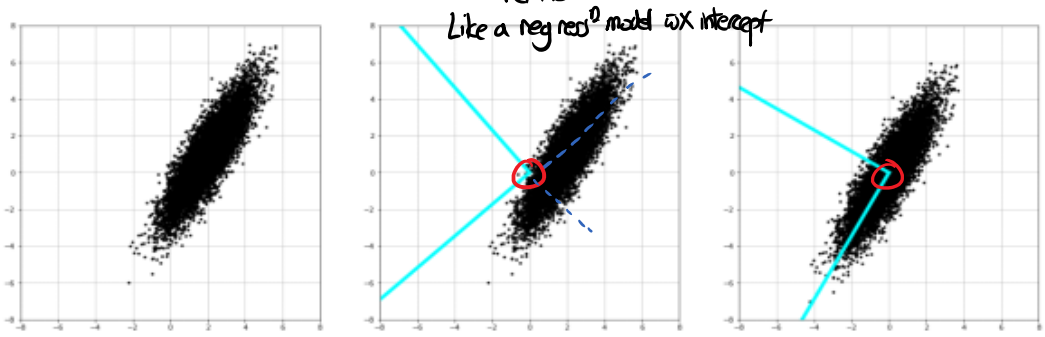
$\frac{\sum_{k=1}^{r}\lambda_k \text{ (up to r modes)}}{\sum_{k=1}^{n}\lambda_k \text{ (All modes)}}$ } Gives us the no. of principal components (cols of P) we need to cover a certain % variance of the data for instance 95% of the data



Why Center data is a must B4 PCA?

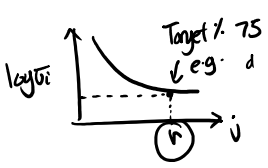
- $P = \mathbb{X}V$ ← orthonormal, Unitary, rotat^n, hence acts thru origin

PCA is Like a reg ress^n model w/ intercept



Truncation

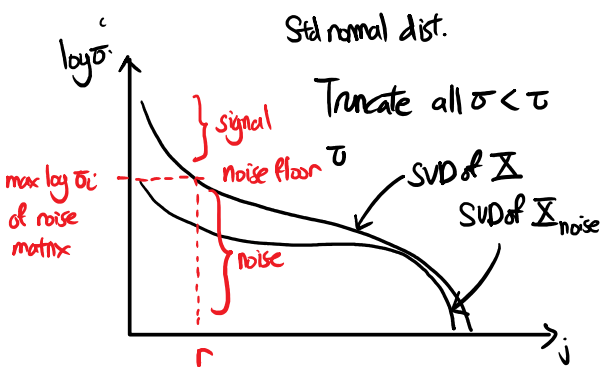
$\mathbb{X} = U\Sigma V^T \cong \tilde{U}\tilde{\Sigma}\tilde{V}^T$



↷ Gavish-Donoho

$\mathbb{X} = \mathbb{X}_{\text{"true"}} + \gamma\mathbb{X}_{\text{noise}}$ ← determines mag noise

Std normal dist.



i) For $\mathbb{X}_{n\times n}$, γ_{known}

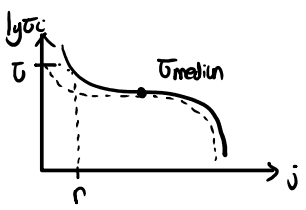
$\tau = \frac{4}{\sqrt{3}}\gamma\sqrt{n}$ ← If B=1

ii) For $\mathbb{X}_{n\times m}$, γ_{known}, $\beta = \frac{\text{smaller mn}}{\text{larger mn}}$

$\tau = \left(2(\beta+1) + \frac{8\beta}{(\beta+1)+(\beta^2+14\beta+1)^{1/2}}\right)^{1/2}\gamma\sqrt{n}$

↳ $\lambda(\beta)$

ii) For $\mathbb{X}_{n\times m}$, $\gamma_{unknown}$, assume $\sigma_{median} < \tau$



$\tau = \omega(\beta)\,\sigma_{med}$, $\omega(\beta) = \frac{\lambda(\beta)}{\mu_\beta}$

Solution of $\int_{(1+\sqrt{\beta})^2}^{u_\beta} \frac{[((1+\sqrt{\beta})^2-t)(t-(1-\sqrt{\beta})^2)]^{1/2}}{2\pi t}\,dt = 1/2$

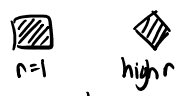
Alignment

$\mathbb{X} = \begin{bmatrix} | & & | \\ x & \cdots & x_m \\ | & & | \end{bmatrix}$ ← eye for example, ← nose

(Translat^n)

Align so if we dot prdt, features to features

- In addition data cannot handle (rotat^n)



BUT goes to show power of CNN to pool out features despite translat^n in data

Randomize SVD

- Data getting higher dim (eg. 1080p → 4K), Very heavy to compute $U_r\Sigma_r V_r^T$
- Assumpt^n : Low intrinsic rank exist for data*
- To improve the r capturing: Do oversampling

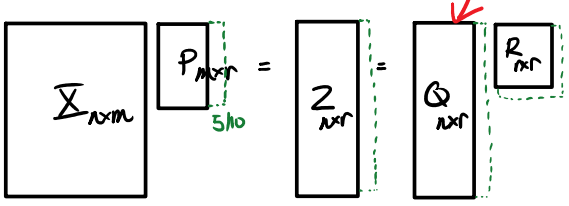
1) Take random Project^n Matrix $P \in \mathbb{R}^{m\times(r+5/10)}$

Project high rank data $\mathbb{X}$ onto P low rank, r matrix

$Z = \mathbb{X}P$

$= QR$ (QR factorizat^n)

Orthonormal basis for col(Z) & col($\mathbb{X}$) due to random vector & high dim geo. principals Col($\mathbb{X}$) sampled by P to give Z, where we can do a QR factorizat^n to obtain Q



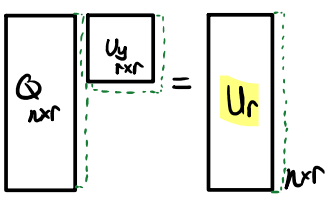
2) Project $\mathbb{X}$ down to the smaller dim Q

$Y = Q^T\mathbb{X} = U_y\Sigma_y V_y^T$

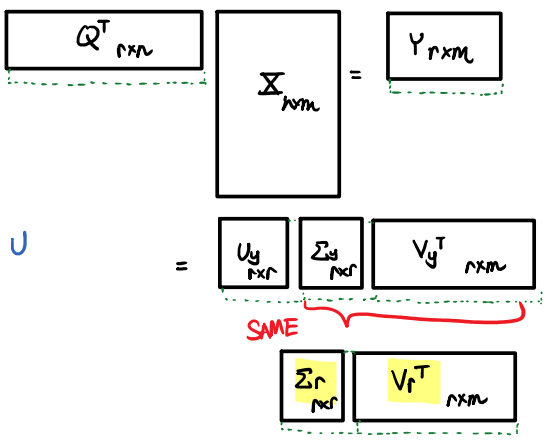
∵ U_y orthonormal basis for Y that would span col($\mathbb{X}$) $\Sigma_y V_y^T = \Sigma_x V_x^T$

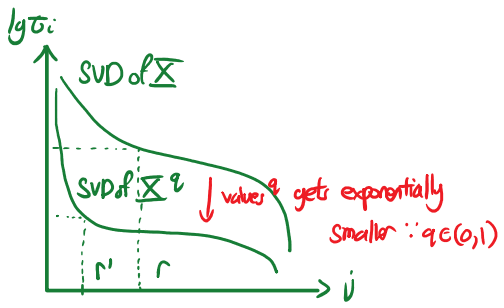
So we computed ΣV^T of the data $\mathbb{X}$, left U

$U_r = QU_y$



∵ If $\mathbb{X}$ low rank instinsic data not low do Power Iterat^n then do R-SVD $\mathbb{X}^q = (\mathbb{X}\mathbb{X}^q)^T\mathbb{X}$ (heavy computat^n)





Over determined Data (Cancer)

Cases

X = (mean centered) [x_1 ; x_n] 216 sample/person (Cancer / Noncancer)
40000 gene markers

$\cong$ Rank r approximation
216 × 4000

$X_{216\times r}$ = $U_{216\times 216}$ · $\Sigma_{216\times 216/\times 4000}$ · $V^T_{r\times 4000}$ = $U\hat{\Sigma}\hat{V}^T$ "Economy SVD"

- $U_{216\times 216}$: Eigenvectors of XX^T
- Σ: Singular Values
- $V^T_{4000\times 4000}$: Eigenvectors of X^TX
- Cols of U := orthonormal basis for Col(X) "Eigen genes"
- Rows of V^T := Orthonormal basis for row(X) "Eigen people"

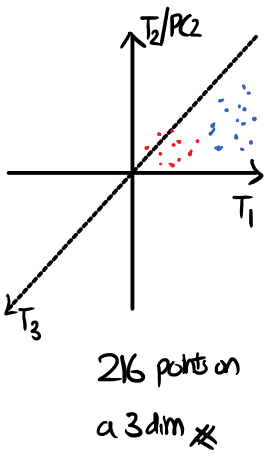
Now to represent the 216 people (rows) in a lower dim
↳ PCA analysis

For every person

x_i (in the row(X)) → Linear Combi of → span of rows(V^T) = row(V^T)

$T_1 = XV_1$ = (1st row X)$^T \cdot V_1$ = PC1 (216 × 1)

Plot PC1, 2, 3 every row in X



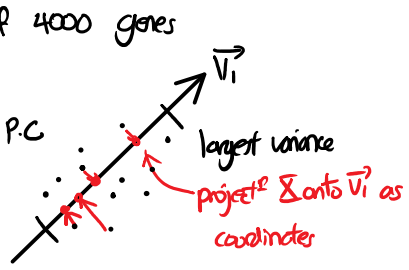
216 points on a 3 dim

Often in python, X^T

V_1^T @ X_i^T = For every i^{th} row, $V_1^T @ X_i^T$

PCA: $T = XV = U\Sigma$

where rows of V^T := Principal Components Direct^{e}/Loadings
V_1^T captures the most variance in the data of 4000 genes that makes up each 216 person
So to represent every 216 person onto the 1st P.C
$T_1 = XV_1$ (project^{e} of X on V_1)



Underdetermine Data (Faces)

X = (mean centered) [x_1 ⋯ x_m] 32000 n pixels = $U_{32000\times 32000}$ · $\Sigma_{2432\times 2432}$ · $V^T_{2432\times 2432}$ = "Economy SVD" = $\hat{U}\hat{\Sigma}\hat{V}^T$ $\cong$ Rank r approx (2432 × 32000)

m faces, 2432
Person 1 face

- $U_{32000\times 32000}$: EV of XX^T
- Σ
- V^T: EV of X^TX
- Cols of V, rows of V^T := ON basis for row(X) "Eigenfeatures"
- Cols U := Orthonormal basis for col(X) "Eigenfaces"

(cols)
Now to represent the 2432 faces in a lower dim
↳ PCA analysis

PCA: $T = X^TU$

For every face in X, col(X)

PC1 / T_1 (1st col of T) = X^TU = (every row X^T = col of X of X^T) · U_1

is a linear combi of $U_{\text{"eigenfaces"}}$ (32000 × 32000)

32000 pixels

"eigenface"

Then same as above. But then if we were to plot PC1 vs PC2, there is not much clustering due to how the 1st eigenface captures the whole general face, hence all X cols should not differ

Now how does this relate to machine learning?

$X = U\Sigma V^T$ (PCA)

$T = X^TU$ ← "eigenfaces"

Test image $\vec{x}$, not in data then,

"eigenfaces" $\tilde{U}_r^T$ (r × 32000) · test x (32000 × 1) = r × 1 α = "Amt of each mode/eigenfaces inside a test face"

· Innerpdt of cols $\tilde{U}$ w test x (U^Tx)

To see the high dim image again

Test $x \cong U_r$ (32000 × r) · α (r × 1) = 32000 × 1 Reconstructed Image using the truncated cols of U