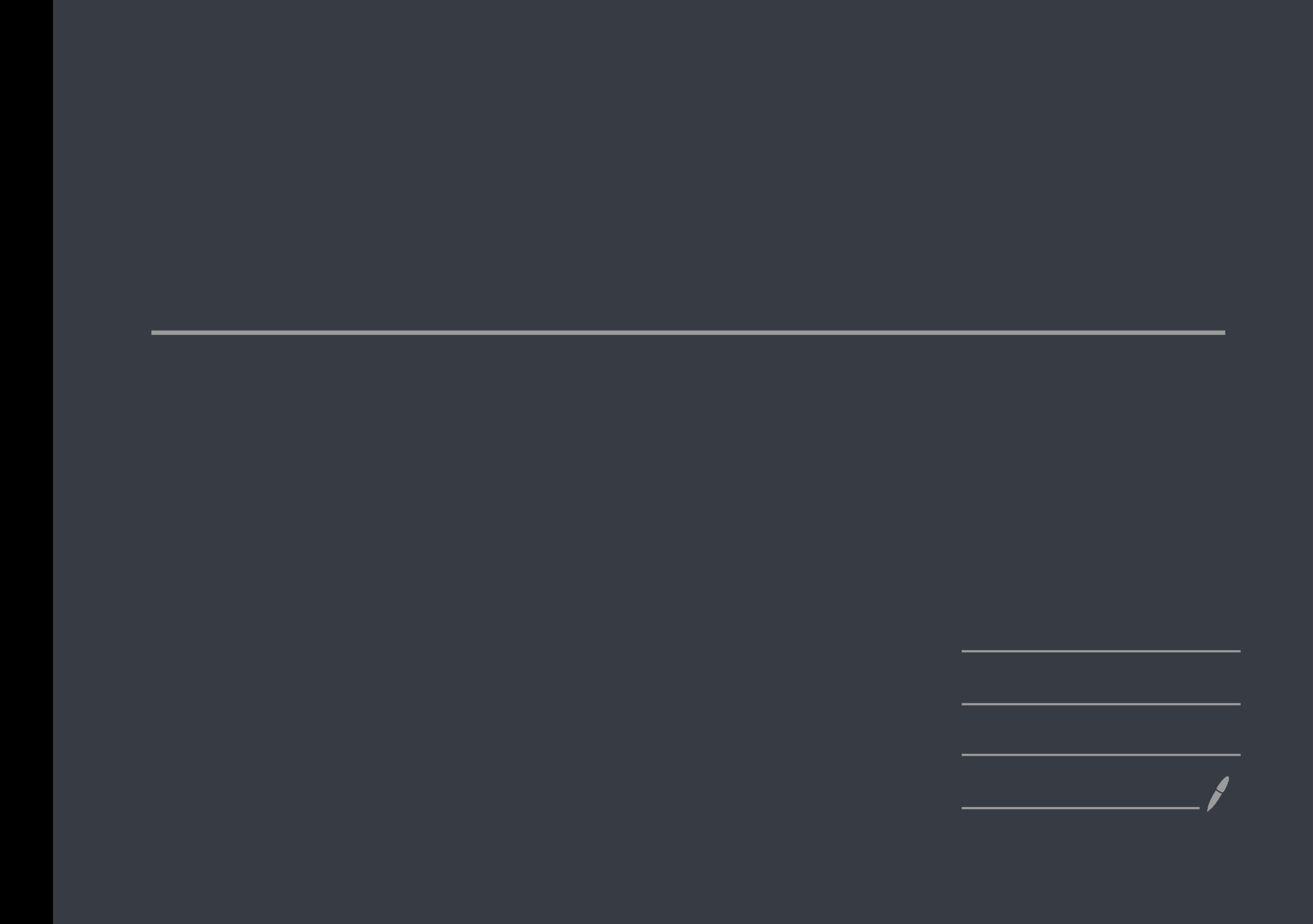


Week 1

High dimensional data: difficult to interpret, hard to analyse, difficult to visualise, storage can be expensive
↳ often overcomplete: many dimensions are redundant, can be explained by combination of other dimensions

Dimensionality reduction exploits structure and correlation
↳ work with more compact representation of the data, without losing info
↳ can think of as a compression technique (e.g. JPEG)

e.g. digit "8": 28×28 pixels, 784-D vector (pixels are structured, not random)
↳ many examples of 8's: differ slightly, can use D.R. to find lower dimensional representation of all 8's ("feature") (easier to work with)

PCA: linear D.R.

Mean Values

Use statistical properties to describe the data
Mean = average data point, not necessarily part of dataset (e.g. average "8" image)

$D = \{x_1, \ldots, x_N\}$ e.g. $D' = \{1, 2, 4, 6, 6\}$

$E[D] = \frac{1}{N} \sum_{n=1}^{N} x_N$ $E[D'] = 3.8$

Mean Values Quiz

1. $D = \{1, 2, 3\}$, $E[D] = \frac{1+2+3}{3} = 2$

2. $D = \left\{ \begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix}, \begin{bmatrix} 2 \\ 5 \\ 8 \end{bmatrix}, \begin{bmatrix} 3 \\ 6 \\ 9 \end{bmatrix} \right\}$, $E[D] = \begin{bmatrix} \frac{1+2+3}{3} \\ \frac{4+5+6}{3} \\ \frac{7+8+9}{3} \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 8 \end{bmatrix}$

3. $D = \left\{ 2\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, 2\begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, 2\begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix} \right\}$, $E[D] = \begin{bmatrix} \frac{2+6+10}{3} \\ \frac{4+8+6}{3} \\ \frac{6+10+2}{3} \end{bmatrix} = \begin{bmatrix} 6 \\ 6 \\ 6 \end{bmatrix}$

4. $D = \left\{ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \right\}$, $E[D] = \begin{bmatrix} \frac{2+4+6}{3} \\ \frac{4+6+5}{3} \\ \frac{6+8+4}{3} \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$

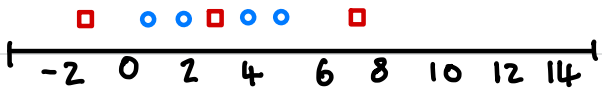
5. $\bar{x}_{n-1}$ of dataset D_{n-1} with $n-1$ data points
   collect new data point x_*
   new mean $\bar{x}_n$ of full dataset $D_n = D_{n-1} \cup \{x_*\}$

   ↳ $\bar{x}_n = \bar{x}_{n-1} + \frac{1}{n}(x_* - \bar{x}_{n-1})$

e.g.
$D_{n-1} = \{1, 2, 3\}$, $x_* = 4$, $D_n = \{1, 2, 3, 4\}$
$E[D_{n-1}] = 2$, $E[D_n] = 2.5$
$2.5 = 2 + \frac{1}{4}(4 - 2)$

6. 2D array of 28×28 image : reshape to vector of length 784 (x.flatten())

Variance of 1D datasets



$E[D_1] = E[D_2] = 3$ (same mean, but data points in D_2 less concentrated around mean compared to D_1)

Variance = used to characterise variability / spread of points in data

$D_1 = \{1, 2, 4, 5\}$, $E[D_1] = 3$

$D_2 = \{-1, 3, 7\}$, $E[D_2] = 3$

↙ Sum of Sq. distances

D_1 : $\frac{(1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2}{4} = 2.5$ (avg. Squared distance from mean value)

D_2 : $\frac{(-1-3)^2 + (3-3)^2 + (7-3)^2}{3} = \frac{32}{3}$ ↗ larger (Spread is higher in D_2)

$\{x_1, \ldots, x_N\} =: X$

$\text{Var}[X] = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$

- can never be -ve!
- expressed in squared units

$\mu = E[X]$

Standard deviation $= \sqrt{\text{Var}[X]}$

↖ Same units as mean value

(usually use st. deviations when talking about spread of data)

Variance of 1D datasets quiz

1. $D = \{1,2,3,2\}$, $\text{Var}[D] = \frac{(1-2)^2+(2-2)^2+(3-2)^2+(2-2)^2}{4} = 0.5$

   $E[D] = 2$

2. Standard deviation $= \sqrt{0.5} =$

3. Add 1 to each element in D: $\text{Var}[D] = \frac{(2-3)^2+(3-3)^2+(4-3)^2+(3-3)^2}{4} = 0.5$ (adding a constant doesn't change the variance)

   $E[D] = 3$

4. Multiply each sample in D by 2: $D = \{2,4,6,4\}$, $\text{Var}[D] = \frac{(2-4)^2+(4-4)^2+(6-4)^2+(4-4)^2}{4} = 2$, $\therefore$ 4 times variance of D

   $E[D] = 4$

5. $\bar{x}_{n-1}$, σ^2_{n-1} for D_{n-1} with $n-1$ samples (↙ variance)

What is variance σ^2_n if we add new element x_*:

$$\sigma^2_n = \frac{n-1}{n}\sigma^2_{n-1} + \frac{1}{n}(x_* - \bar{x}_{n-1})(x_* - \bar{x}_n)$$

e.g. $D_{n-1} = \{1,3,5\}$, $E[D_{n-1}] = 3$, $\sigma^2_{n-1} = \frac{(1-3)^2+(3-3)^2+(5-3)^2}{3} = 2.\dot{6}$

$x_* = 4$: $D_n = \{1,3,5,4\}$, $E[D_n] = \frac{13}{4}$, $\sigma^2_n = \frac{(1-\frac{13}{4})^2+(3-\frac{13}{4})^2+(5-\frac{13}{4})^2+(4-\frac{13}{4})^2}{4}$

$= 2.1875$

$$2.1875 = \frac{3}{4}(2.\dot{6}) + \frac{1}{4}(4-3)(4-\frac{13}{4})$$

Symmetric, positive definite matrices

Matrix M is symmetric if $M = M^T$

Symmetric matrix M is positive-definite if $z^T M z > 0$:

$$\begin{bmatrix} z_1 & z_2 & \cdots & z_n \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ M_{21} & M_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ M_{n1} & \cdots & \cdots & M_{nn} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} > 0 \quad \text{for a real column vector } z \neq 0$$

(Symmetric and positive-definite) (e.g. $\begin{bmatrix} 7 & 1 \\ 1 & 2 \end{bmatrix}$)

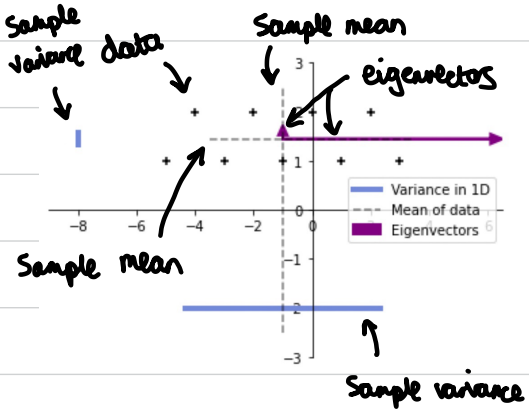
The eigenvalues of Symmetric positive-definite matrices are +ve

↳ e.g. M is Symmetric positive-definite matrix, λ is eigenvalue of M, so $Mv = \lambda v$, where v is the corresponding eigenvector

↳ when M is positive-definite, we have $v^T M v = \lambda v^T v > 0$

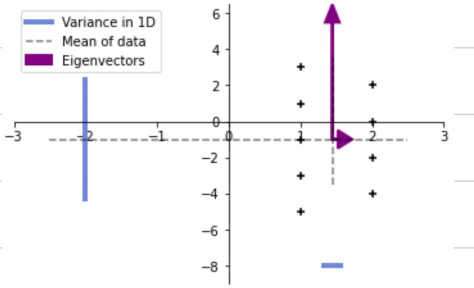
↳ since $v^T v > 0$, λ must be +ve

Covariance matrix: Symmetric and positive-definite



Covariance matrix for dataset of 2 indep. r.v. x and y

$$\begin{bmatrix} \text{cov}(x,x) & \text{cov}(x,y) \\ \text{cov}(y,x) & \text{cov}(y,y) \end{bmatrix} = \begin{bmatrix} 7.5 & 0 \\ 0 & 1.1 \end{bmatrix}$$



$$\begin{bmatrix} \text{cov}(x,x) & \text{cov}(x,y) \\ \text{cov}(y,x) & \text{cov}(y,y) \end{bmatrix} = \begin{bmatrix} 1.11 & 0 \\ 0 & 7.5 \end{bmatrix}$$

In both covariance matrices, 2 variables are independent since $\text{cov}(x,y) = \text{cov}(y,x) = 0$ ($\therefore$ matrices are diagonal)

↳ diagonal entries: variances for each of the variables, always +ve

↳ eigenvalues of matrix = variances

Eigenvectors are orthogonal to one another due to matrix symmetry

↳ represent different directions in which the data varies

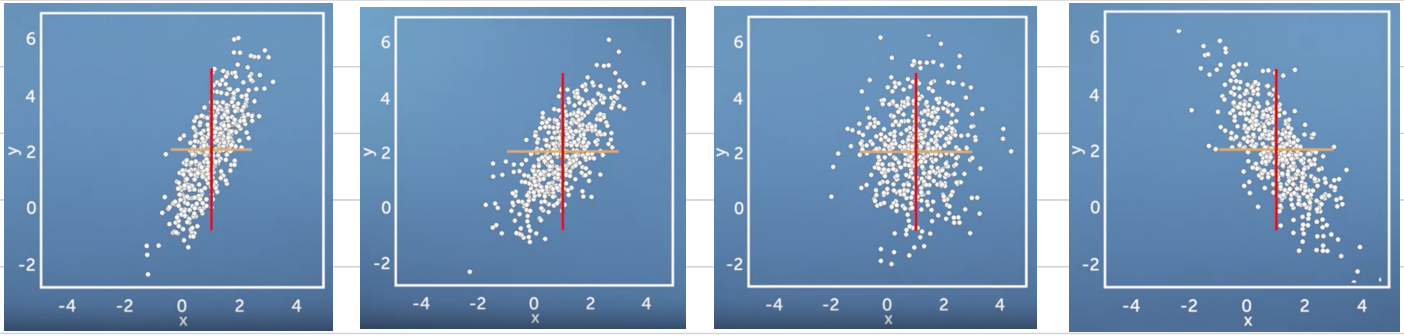
↳ directions in which variables vary the most coincide with the eigenvectors (in examples: along x and y axes)

$\text{cov}(x,y) = \text{cov}(y,x)$

Matrix of eigenvalues D: obtained by diagonalisation → $D = C^{-1}MC$, C: matrix of eigenvectors

↳ covariance matrix similar to diagonal matrix with +ve, real entries (variances along the eigenvectors)

Variance of higher dimensional datasets



Same variances in x and y, same means, but different shapes

↳ if we look at horizontal and vertical spread of data: can't explain any correlation between x and y

$Cov[x,y] = E[(x-\mu_x)(y-\mu_y)]$

$\mu_x = E[x]$

$\mu_y = E[y]$

$Var[x]$

$Var[y]$

$Cov[x,y]$

$Cov[y,x]$

$$\begin{bmatrix} Var[x] & Cov[x,y] \\ Cov[y,x] & Var[y] \end{bmatrix}$$

(covariance matrix)

always symmetric, positive definite matrix

Cov. between x and y is +ve: on avg. y increases if we increase x

Cov. between x and y is -ve: on avg. y decreases if we increase x

Cov. between x and y is 0: x and y are uncorrelated

Covariance matrix is every element is 3D vector with (x, y, z) components:

$$\begin{bmatrix} \text{Var}[x] & \text{Cov}[x,y] & \text{Cov}[x,z] \\ \text{Cov}[y,x] & \text{Var}[y] & \text{Cov}[y,z] \\ \text{Cov}[z,x] & \text{Cov}[z,y] & \text{Var}[z] \end{bmatrix}$$

(variances on diagonal, cross-covariances on off-diagonal)

Dataset of D-dimensional vectors $x = \begin{bmatrix} x_1 \\ \vdots \\ x_D \end{bmatrix}$, covariance matrix: $D \times D$

$$D = \{x_1, \ldots, x_N\}, \quad x_i \in \mathbb{R}^D$$

$$\text{Var}[D] = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T$$

↑ $D \times D$ matrix

Covariance matrix of a 2D dataset quiz

1. 2D dataset $D = \{x_i\}_{i=1}^{N}$ with N samples
   ↳ each sample x_i: 2D vector of x, y

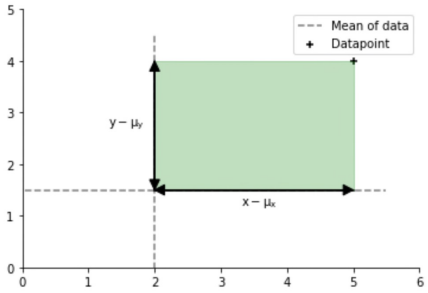
$$\begin{bmatrix} 0 & 1 \\ 1 & 2 \\ 0 & 1 \\ 1 & 2 \end{bmatrix} \quad \begin{matrix} x & y \\ \downarrow & \\ \end{matrix} \begin{bmatrix} -0.5 & -0.5 \\ 0.5 & 0.5 \\ -0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix} \begin{matrix} x \\ \downarrow \\ y \end{matrix} \begin{bmatrix} -0.5 & 0.5 & -0.5 & 0.5 \\ -0.5 & 0.5 & -0.5 & 0.5 \end{bmatrix}$$

(4×2) (2×4)

(other way around)

Covariance between 2 scalar random variables: $\text{Cov}[x,y] = E[(x - \mu_x)(y - \mu_y)] \approx \frac{1}{N} \sum_{i=1}^{N} (x - \mu_x)(y - \mu_y)$

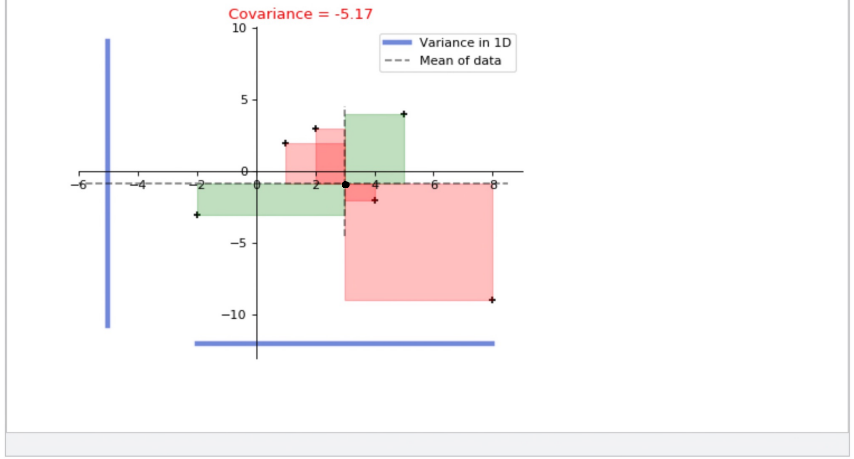
In the formula for covariance, we can think of each individual multiplication as the calculation of an area, a rectangle with sides $\mathrm{x} - \mu_x$ and $\mathrm{y} - \mu_y$.



as x increases from μ_x, y also increases

For this datapoint, an increase in x from the mean is linked to an increase in y. Where $\mathrm{x} - \mu_x$ and $\mathrm{y} - \mu_y$ have the same sign, the contribution to the covariance is positive and in green, while if the signs are opposite it will be negative and in red. In other words, green means that x and y are positively correlated, while red means they're negatively correlated.

The total sum of areas, divided by the number of points n, will be the value of the covariance.

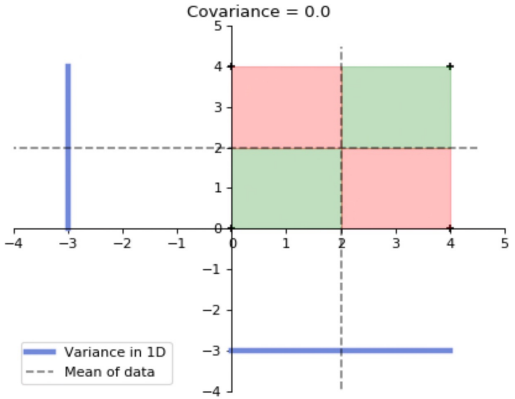


The dashed lines meet at the mean of the dataset. The blue lines represent the magnitude of the variance of the x (horizontal) and y (vertical) components of the dataset.

If red and green balance out, the covariance will be 0. Otherwise the sign of the covariance will give a direction in which the points appear to correlate.

What is $\mathrm{cov}(x, y)$ for the dataset in the array labelled "Q1: square"? Is it what you would expect from the plot?

↳ covariance is 0



points are evenly distributed around the mean, so they balance out, so no way to determine direction of correlation

2. $\begin{bmatrix} cov(x,x) & cov(x,y) \\ cov(y,x) & cov(y,y) \end{bmatrix} = \begin{bmatrix} Var(x) & cov(x,y) \\ cov(y,x) & Var(y) \end{bmatrix}$

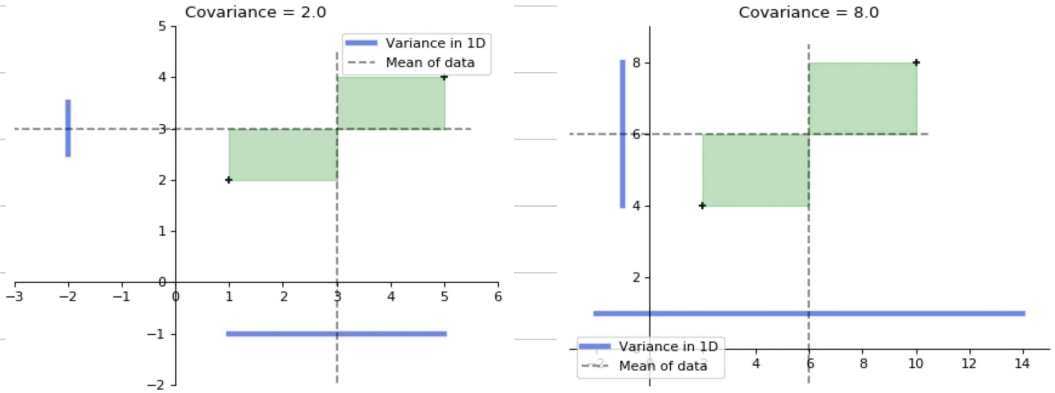
$D = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 5 \\ 4 \end{bmatrix} \right\}$, $Cov[x,y] = \frac{1}{N} \sum_{i=1}^{N} (x - \mu_x)(y - \mu_y) = \frac{1}{2}\left[(1-3)(2-3) + (5-3)(4-3)\right] = 2$

$\mu_x = 3, \ \mu_y = 3$

$Var[x] = \frac{1}{2}\left[(1-3)^2 + (5-3)^2\right] = 4$

$Var[y] = \frac{1}{2}\left[(2-3)^2 + (4-3)^2\right] = 1$

Covariance matrix $= \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix}$



* each element in D by 2

$Cov[x,y] \uparrow \times 4$

$Var[x] \uparrow \times 4$

$Var[y] \uparrow \times 4$

3. Dataset with covariance matrix $\begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix}$

Multiply each vector in D by 2: $Var[x'] = Var[x] \times 2^2 = 12$

$Var[y'] = Var[y] \times 2^2 = 16$ $\Rightarrow \begin{bmatrix} 12 & 8 \\ 8 & 16 \end{bmatrix}$

$Cov[x,y] = 2 \times 2^2 = 8$

4. $D = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 7 \\ 4 \end{bmatrix} \right\}$, covariance matrix $\begin{bmatrix} 9 & 3 \\ 3 & 1 \end{bmatrix}$

↳ add $\begin{bmatrix} 2 \\ 2 \end{bmatrix}$ to each element in D : $D = \left\{ \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \begin{bmatrix} 9 \\ 6 \end{bmatrix} \right\}$

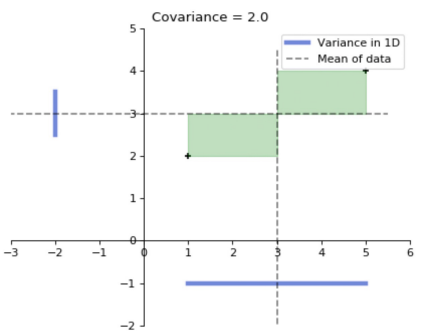
$Cov[x,y] = Cov[y,x] = \frac{1}{2}[(3-6)(4-5)+(9-6)(6-5)] = 3$

$\mu_x = 6, \quad \mu_y = 5$

$Var[x] = \frac{1}{2}[(3-6)^2+(9-6)^2] = 9$

$Var[y] = \frac{1}{2}[(4-5)^2+(6-5)^2] = 1$

$\Rightarrow \begin{bmatrix} 9 & 3 \\ 3 & 1 \end{bmatrix}$



(covariance matrix doesn't change)

5. Dataset D, every element in D has x and y coordinate

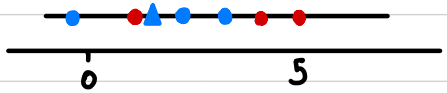
↳ covariance matrix = $\begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$

↳ x and y are positively correlated - when x increases then y increases on average and vice versa

Effect on the mean

Linear transformation : Shift data around or stretch it

$D = \{-1, 2, 3\}$, $E[D] = \frac{4}{3}$



↑ Shift blue data to right by 2 : mean also shifts by 2

$D' = \{1, 4, 5\} = D+2$, $E[D'] = \frac{10}{3} = \frac{4}{3} + 2$

(mean + shift)

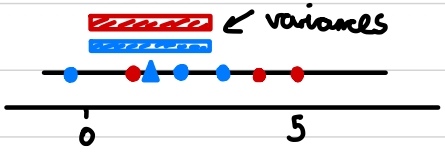
$E[D+a] = a + E[D]$ (Shift by a)

$D'' = \{-2, 4, 6\}$
$E[D''] = \frac{8}{3} = \frac{4}{3} \cdot 2$ ← Scaling factor

$E[\alpha D] = \alpha E[D]$ (Scale by α)

$E[\alpha D + a] = \alpha E[D] + a$ ← Shift (linear transformations on mean of data)
↑
Scaling factor

Effect on the covariance



red dots: data shifted by 2

Shifting data: no effect on variance

$Var[D] = Var[D+a]$
↑ offset applied to each element of D

$Var[\alpha D] = \alpha^2 Var[D]$ (e.g. Scale data by 2, squared distance from μ scaled by 4, so Var. 4x ↑ as before)
↑ Scales each element, real number

$D = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^p$

Variance of data in D: given by covariance matrix

Apply linear transformation: $Ax_i + b$

$$Var[AD+b] = A\,Var[D]\,A^T$$

↳ $Var[D] = E[(D-\mu)(D-\mu)^T]$

↳ multiply D by A: $Var[AD] = E[(AD - A\mu)(AD - A\mu)^T]$

$$= E[A(D-\mu)(D-\mu)^T A^T]$$
$$= A\,E[(D-\mu)(D-\mu)^T]\,A^T$$
$$= A\,Var[D]\,A^T$$

Linear transformation of data: Shifting only affects mean, scaling affects mean and variance

Week 1 Lab

Means and (co)variances of a dataset

Vectorized code significantly improves runtime on larger datasets

e.g. data matrix X of size (N, D)

↳ what is mean and covariance when we apply affine transformation $Ax_i + b$ for each x_i in X

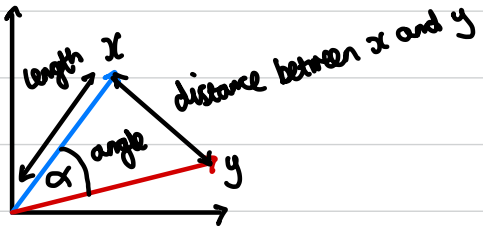
$m' = \text{affine\_mean}(m, A, b) = A \,@\, \text{mean} + b$

$S' = \text{affine\_covariance}(S, A, b) = A \,@\, S \,@\, A.T$ (covariance not affected by b)

(m: old mean, S: old covariance)

Week 2: Inner Products

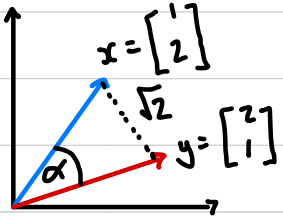
DR: find compact representations of data that live in a lower dimensional space (but similar to original data)



$$\cos\alpha = \frac{x^T y}{\|x\|\|y\|} = \frac{4}{\sqrt{5}\times\sqrt{5}} = \frac{4}{5}, \therefore \alpha \approx 0.64 \text{ rad}$$

$$x^T y = \sum_{i=1}^{N} x_i y_i, \quad x, y \in \mathbb{R}^N$$

$$\|x\| = \sqrt{x^T x} = \sqrt{\sum_{i=1}^{2} x_i^2}$$



$$d(x-y) = \left\| \begin{bmatrix}1\\2\end{bmatrix} - \begin{bmatrix}2\\1\end{bmatrix} \right\| = \left\| \begin{bmatrix}-1\\1\end{bmatrix} \right\| = \sqrt{1+1} = \sqrt{2}$$

e.g. $\|x\| = \sqrt{1^2+2^2} = \sqrt{5}$, $\|y\| = \sqrt{5}$

$$d(x,y) = \|x-y\| = \sqrt{(x-y)^T(x-y)}$$

Inner product : generalisation of the dot product

↳ want to express geometric properties (lengths, angles) between vectors

Def: $x, y \in V$

$\langle \cdot, \cdot \rangle : V \times V \to \mathbb{R}$ (mapping from $V \times V$ to real numbers)

- symmetric : $\langle x, y \rangle = \langle y, x \rangle$
- positive definite : $\langle x, x \rangle \geq 0$, $\langle x, x \rangle = 0 \iff x = 0$ ← zero vector (if and only if)
- bilinear : $x, y, z \in V$, $\lambda \in \mathbb{R}$, inner product between $\lambda x + z$ and y : $\langle \underbrace{\lambda x + z}_{\text{linearity}}, y \rangle = \lambda \langle x, y \rangle + \langle z, y \rangle$

$$\langle x, \underbrace{\lambda y + z}_{\text{linearity}} \rangle = \lambda \langle x, y \rangle + \langle x, z \rangle$$

Inner product : • $\langle x, y \rangle = x^T I y$ (dot product)

(bilinear : linearity in both arguments of function)

- $\langle x, y \rangle = x^T A y$

  e.g. $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \to 2x_1y_1 + x_2y_1 + x_1y_2 + 2x_2y_2$

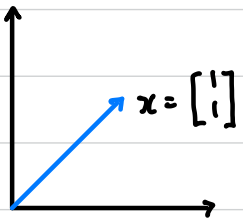
  ↑ any symmetric positive-definite matrix defines a valid inner product

Lengths and distances

$\|x\| = \sqrt{\langle x, x \rangle}$ ← inner product is positive-definite (i.e. ≥ 0, so can sqrt)

↑ length / norm of x

e.g. $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$



$\langle x, y \rangle = x^T y \Rightarrow \|x\| = \sqrt{2}$ (v1) ← length of vector (longer than v2)

or $\langle x, y \rangle = x^T \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} y = x_1 y_1 - \frac{1}{2}(x_1 y_2 + x_2 y_1) + x_2 y_2$ (v2)

$\Rightarrow \|x\| = \sqrt{x_1^2 - \frac{1}{2}(x_1 x_2 + x_2 x_1) + x_2^2} = \sqrt{x_1^2 - x_1 x_2 + x_2^2}$

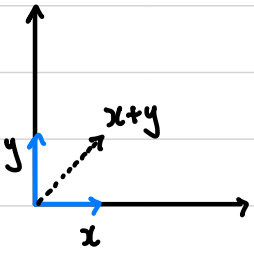
$\|x\|^2 = \langle x, x \rangle = 1 + 1 - 1 = 1, \Rightarrow \|x\| = 1$ (length of vector using v2 of inner product)

(Squared norm : inner product of x with itself)

Stretch vector by α

$\|\alpha x\| = |\alpha| \, \|x\|$ ← absolute

Triangle inequality: $\|x+y\| \leq \|x\| + \|y\|$



$x = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad y = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$\|x\| = 1 = \|y\|$

$\|x+y\| = \sqrt{2}$

$\sqrt{2} \leq 2$

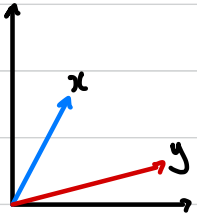
Cauchy - Schwarz inequality: $|\langle x, y \rangle| \leq \|x\| \|y\|$

$d(x,y) = \|x-y\| = \sqrt{\langle x-y, x-y \rangle}$ (length of difference vector)

↳ use a dot product : distance = Euclidean distance

e.g. $x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \; y = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$

$x - y = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$



Dot product : $\sqrt{8}$

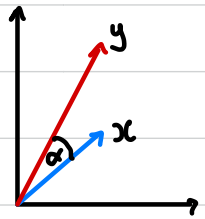
or: $\langle x, y \rangle = x^T \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} y$ (different inner product)

$\|x-y\| = \sqrt{12}$

Angles and Orthogonality

$$\cos\alpha = \frac{\langle x, y \rangle}{\|x\|\|y\|}$$

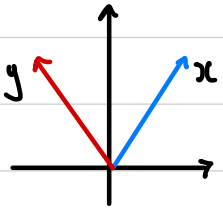
, e.g. $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $y = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$



$$\cos\alpha = \frac{x^T y}{\sqrt{x^T x \, y^T y}} = \frac{3}{\sqrt{10}}$$

$$\therefore \alpha \approx 0.32 \text{ rad} \approx 18^\circ$$

e.g. $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$



$\cos\alpha = 0$, $\therefore \alpha = \frac{\pi}{2}$ rad $= 90^\circ$

(orthogonal vectors)

Inner product allows us to characterise orthogonality: vectors which are most dissimilar, nothing in common besides origin

2 vectors x and y, where x and y are non-zero vectors are orthogonal iff their inner product $= 0$

Vectors orthogonal w.r.t. 1 inner product don't have to be orthogonal w.r.t another inner product

e.g. different inner product: $\langle x, y \rangle = x^T \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} y \Rightarrow \langle x, y \rangle = -1$ (vectors not orthogonal w.r.t. this inner product)

Can find basis of vector space so basis are orthogonal to each other

↳ $\langle b_i, b_j \rangle = 0$, if $i \neq j$

↳ use inner product to normalise basis vectors: $\|b_i\| = 1$ (orthonormal basis)

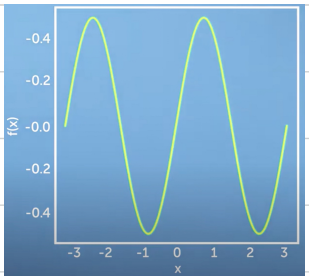
Unconventional Inner Products

Inner product between 2 functions: $\langle u, v \rangle = \int_a^b u(x)v(x)\,dx$ (if $= 0$, functions u and v are orthogonal)

e.g. $u(x) = \sin(x)$

$v(x) = \cos(x)$

$f(x) = u(x)v(x)$ →



odd function, as $f(-x) = -f(x)$

if we set integral limit to $-\pi$ and π, integral of $f(x) = 0$, so sin and cos are orthogonal

e.g. Set of functions $\{1, \cos x, \cos 2x, \cos 3x, \ldots\}$ are all orthogonal if we integrate from $-\pi$ to π

2 uncorrelated r.v. : $\mathrm{Var}[x+y] = \mathrm{Var}[x] + \mathrm{Var}[y]$ (r.v. under each term)

Pythagorean theorem: $c^2 = a^2 + b^2$

r.v. can be considered vectors in a vector space

Symmetric, positive-definite, linear

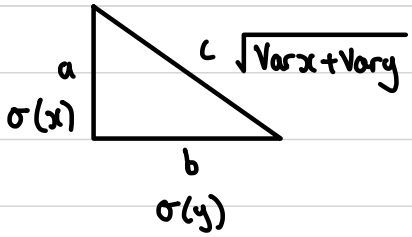
Inner product between r.v. : $\langle x, y \rangle = \mathrm{Cov}[x, y]$

$\mathrm{Cov}(\alpha x + y, z) = \alpha \mathrm{Cov}[x, z] + \mathrm{Cov}[y, z]$ (linearity)

$\|x\|$ = length of r.v. : $\sqrt{\mathrm{Cov}[x,x]} = \sqrt{\mathrm{Var}[x]} = \sigma(x)$

↳ 0 vector has no uncertainty, as σ is 0

Angle between 2 r.v. : $\cos\theta = \dfrac{\langle x, y \rangle}{\|x\| \|y\|} = \dfrac{\mathrm{Cov}[x, y]}{\sqrt{\mathrm{Var}[x]\mathrm{Var}[y]}}$ (= 0 iff $\mathrm{Cov}[x,y] = 0$, is the case when x and y are uncorrelated)



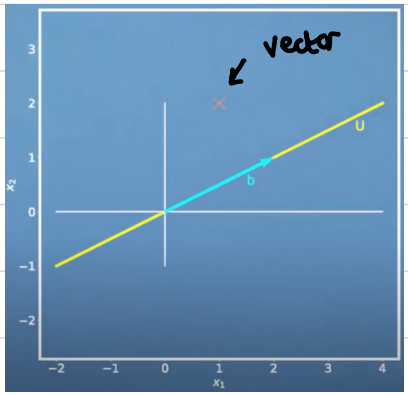
(geometric interpretation of r.v.)

Week 3: Orthogonal Projections

Compress data: loss of information

↳ want to keep informative dimensions, ignore irrelevant dimensions

Projection onto 1D Subspaces

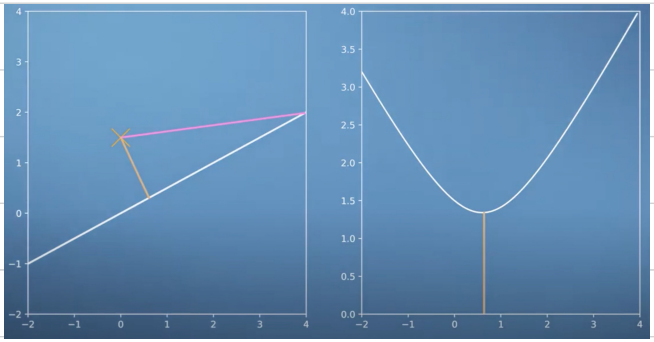


Vector can be represented as linear combination of basis vectors

1D subspace U with basis vector b

↳ all vectors in U: represented as λb for some lambda

Want a vector in U that is closest to x



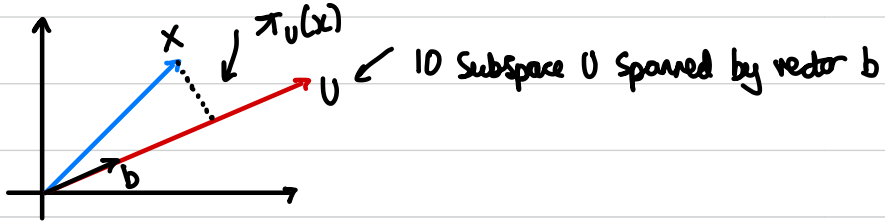
find vector in U closest to x by an orthogonal projection of x onto U (difference vector of x and projection is orthogonal to U)

↙ projection of x onto U (↘: coordinate of projection w.r.t. b of subspace U)

$\pi_U(x)$ ↙ multiple of basis vector that spans U

1. $\pi_U(x) \in U \Rightarrow \exists \lambda \in \mathbb{R} : \pi_U(x) = \lambda b$ (as $\pi_U(x) \in U$)
2. $\langle b, \pi_U(x) - x \rangle = 0$ (orthogonality condition: inner product of basis vector and difference vector is 0)

(both properties generally hold for any x in $\mathbb{R}^D$ and 1D subspaces U)



↙ 1D subspace U spanned by vector b

$\langle b, \pi_U(x) - x \rangle = 0$

$\Leftrightarrow \langle b, \pi_U(x) \rangle - \langle b, x \rangle = 0$ (exploit linearity of inner product)

$\Leftrightarrow \langle b, \lambda b \rangle - \langle b, x \rangle = 0$

$\Leftrightarrow \lambda \|b\|^2 - \langle b, x \rangle = 0$

$\Leftrightarrow \lambda = \frac{\langle b, x \rangle}{\|b\|^2}$

$\Rightarrow \pi_U(x) = \lambda b = \frac{\langle b, x \rangle b}{\|b\|^2}$

↳ choose dot product as inner product: $\frac{b^T x\, b}{\|b\|^2} = \frac{bb^T}{\|b\|^2} x = \pi_U(x)$

projection matrix: projects 2D point onto 1D subspace

coordinate of projected point w.r.t. basis b by using dot product of b with x

if $\|b\| = 1 \Rightarrow \lambda = b^T x$, $\pi_U(x) = \underbrace{bb^T}_{\text{projection matrix}} x$

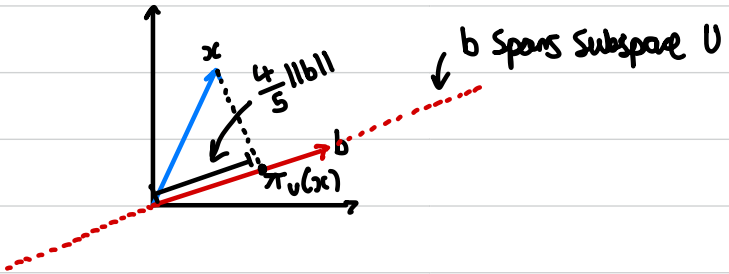
Projection $\pi_U(x)$ still a vector in $\mathbb{R}^D$

↳ but don't need D coordinates to represent it, only need λ

Example 1D Projection

$$\pi_U(x) = \frac{x^T b}{\|b\|^2} b$$

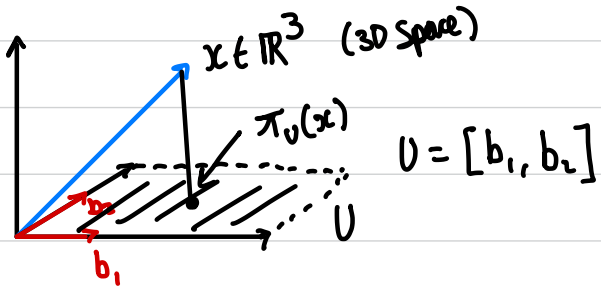
e.g. $b = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$, $x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$



want orthogonal projection of x onto U

$$\pi_U(x) = \frac{4}{5}\begin{bmatrix} 2 \\ 1 \end{bmatrix} \text{ (orthogonal projection)}$$

N-dimensional Projections



$U = [b_1, b_2]$

$\pi_U(x)$ is element of U, so can represent as linear combination of basis vectors of U

$$\pi_U(x) = \lambda_1 b_1 + \lambda_2 b_2$$

$$\langle x - \pi_U(x), b_1 \rangle = 0$$

$$\langle x - \pi_U(x), b_2 \rangle = 0$$

represent projection as L.C. of basis of subspace

1. $\pi_U(x) = \sum_{i=1}^{M} \lambda_i b_i$ $\qquad \lambda = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_M \end{bmatrix}$, $B = [b_1 | \dots | b_M]$

λ: $M \times 1$, B: $D \times M$

2. $\langle \pi_U(x) - x, b_i \rangle = 0, \quad i = 1, \dots, M$

difference vector between x and projection is orthogonal to subspace

$\pi_U(x) = B\lambda$

use dot product as inner product: $\langle \pi_U(x) - x, b_i \rangle = \langle B\lambda - x, b_i \rangle \overset{?}{=} 0$ ← needs to be 0

$\iff \langle B\lambda, b_i \rangle - \langle x, b_i \rangle = 0, \quad i = 1 \dots M$

$\iff \lambda^T B^T b_i - x^T b_i = 0, \quad i = 1 \dots M$

$\iff \lambda^T B^T B - x^T B = 0$

$\iff \lambda^T = x^T B (B^T B)^{-1}$

$\iff \lambda = (B^T B)^{-1} B^T x$

$\implies \pi_U(x) = B\lambda = \underbrace{B(B^T B)^{-1} B^T}_{\text{projection matrix}} x$

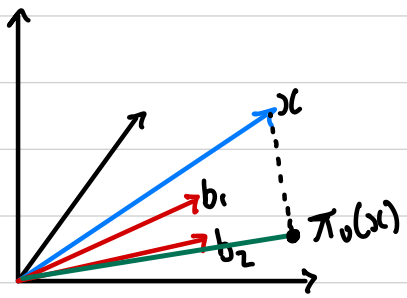
Special case of orthonormal basis: $B^T B = I$

$\pi_U(x) = BB^T x$

↑ projected vector in D, but only require M coordinates (λ vector) to be represented as linear combination of basis vectors of subspace U

1D case: $\pi_U(x) = \frac{b^T x}{b^T b} b, \quad \lambda = \frac{b^T x}{b^T b}$

Example N-dimensional Projections



$$x = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}, \quad b_1 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}, \quad b_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$U = [b_1, b_2]$$

Projected vector can be represented as L.C. of basis of subspace, and vector that connects data point and its projection must be orthogonal to the subspace

Orthogonal projection: $\pi_U(x) = B\lambda$

$$B = [b_1 | b_2] = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\lambda = (B^T B)^{-1} B^T x$$

$$B^T x = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$B^T B = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix}$$

$$B^T B \lambda = B^T x$$

(Gaussian elimination) $\Rightarrow \lambda = \begin{bmatrix} -1 \\ 3 \end{bmatrix}$

$\Rightarrow \pi_U(x) = -1b_1 + 3b_2 = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}$

↙ projected point has 3rd component of 0, subspace requires 3rd component is always 0

↖ 3D vector, but can represent using 2 coordinates if we use basis defined by b_1 and b_2, ∴ is compact representation of projection of x onto lower dimensional subspace

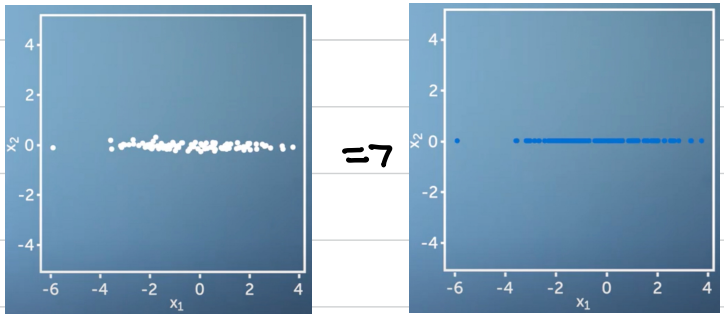
Week 4: Principal Component Analysis (PCA)

Algorithm for linear dimensionality reduction

↳ used for data compression + visualisation

High dimensional data (e.g. images) often lie in lower dimensional subspace

↳ many dimensions are highly correlated



$\Rightarrow$

PCA: use orthogonal projections to find lower dimensional representations of data that retain as much info as possible expressed using fewer basis

$X = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^D$ ← D-dimensional vectors

↑ data

↑ vector

Use dot product for inner product

assume orthonormal basis of $\mathbb{R}^D$

① $x_n = \sum_{i=1}^{D} \beta_{in} b_i$ (every vector in $\mathbb{R}^D$ can be represented as L.C. of basis vectors)

② $\beta_{in} = x_n^T b$

↖ orthogonal projection of x_n onto 1D subspace spanned by i'th basis vector

③ $B = (b_1, \dots, b_M)$ (matrix of orthonormal basis vectors)

$\tilde{x} = BB^T x$ (orthogonal projection of x onto subspace spanned by M basis vectors)

↑

coordinates of $\tilde{x}$ w.r.t. basis vectors in B ("code")

1. Centered data: $E[X] = 0$
2. ONB $b_1, \dots, b_D$

in PCA: ignore this

$$\tilde{x}_n = \sum_{i=1}^{M} \beta_{in} b_i + \sum_{i=M+1}^{D} \beta_{in} b_i \in \mathbb{R}^D$$

M-dimensional subspace (D-M)-dimensional subspace (orthogonal complement to first subspace)

(2 sums span property ①)

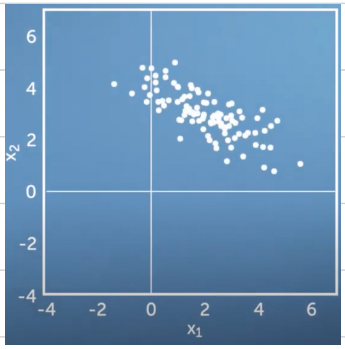
: principal subspace ($b_1, \dots, b_M$ span this)

$\tilde{x}_N$ still D-dimensional vector, but lies in M-dimensional subspace of $\mathbb{R}^D$

↳ only M coordinates $(\beta_{n1} \dots \beta_{nM})$ necessary to represent it

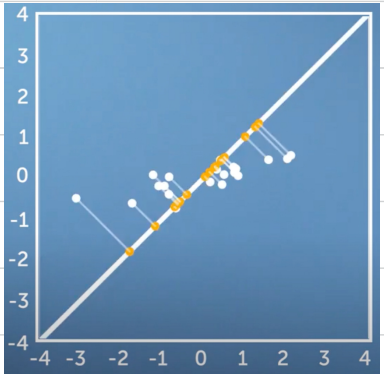
Want to find parameters β_{in}, orthonormal basis vectors b_i such that avg. squared reconstruction error is minimised

↳ $J = \frac{1}{N} \sum_{n=1}^{N} \| x_n - \tilde{x}_n \|^2$ (loss function)

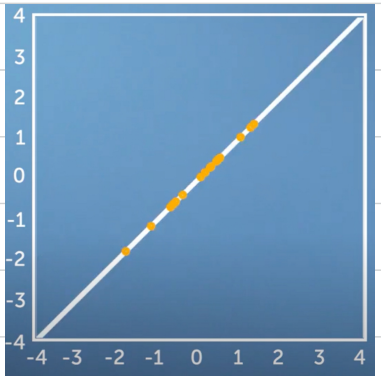


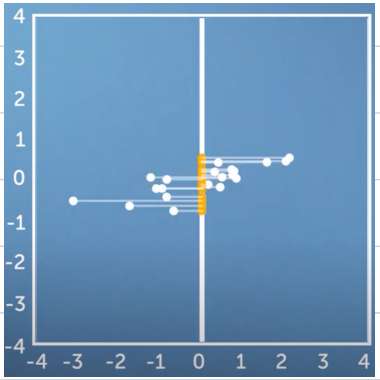
want to find good 1D subspace such that avg. squared reconstruction error of data and projections is minimised

(options of subspaces)

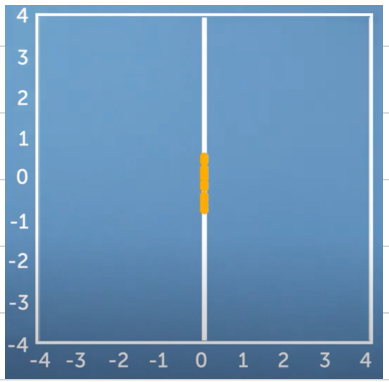


=?





=?



Some projections more informative than others: PCA finds the best one

↳ partial deriv. of J w.r.t. parameters β_{in} and b_i, set to 0, solve for optimal parameters

$$\frac{\partial J}{\partial \{\beta_{in}, b_i\}} = \frac{\partial J}{\partial \tilde{x}_n} \frac{\partial \tilde{x}_n}{\partial \{\beta_{in}, b_i\}}$$

$$\frac{\partial J}{\partial \tilde{x}_n} = -\frac{2}{N}(x_n - \tilde{x}_n)^T$$

$$\frac{\partial J}{\partial \beta_{in}} = \frac{\partial J}{\partial \tilde{x}_n} \frac{\partial \tilde{x}_n}{\partial \beta_{in}}, \qquad \frac{\partial \tilde{x}_n}{\partial \beta_{in}} = b_i \;,\; i = 1 \dots M$$

optimal coordinates of $\tilde{x}_n$ w.r.t. basis are orthogonal projections of original data point onto i'th basis vector that spans principal subspace

$$= -\frac{2}{N}(x_n - \tilde{x}_n)^T b_i$$

$$= -\frac{2}{N}\left(x_n - \sum_{j=1}^{M} \beta_{jn} b_j\right)^T b_i \overset{\text{ONB}}{=} -\frac{2}{N}\left(x_n^T b_i - \beta_{in} \underbrace{b_i^T b_i}_{=1}\right) = -\frac{2}{N}(x_n^T b_i - \beta_{in}) = 0 \iff \beta_{in} = x_n^T b_i$$

Ⓐ $\tilde{x}_n = \sum_{j=1}^{M} \beta_{jn} b_j$ (projected data point)

Ⓑ $J = \frac{1}{N} \sum_{n=1}^{N} \|x_n - \tilde{x}_n\|^2$ (loss function)

Ⓒ $\frac{\partial J}{\partial \tilde{x}_n} = -\frac{2}{N}(x_n - \tilde{x}_n)^T$ (partial deriv. of loss w.r.t. $\tilde{x}_n$)

Ⓓ $\beta_{jn} = x_n^T b_j$, $j = 1, \dots, M$ (optimal coordinates)

$$\tilde{x}_n \overset{A}{=} \sum_{j=1}^{M} \beta_{jn} b_j$$

$$\overset{B}{=} \sum_{j=1}^{M} (x_n^T b_j) b_j \qquad \text{(dot product is symmetric)}$$

$$= \sum_{j=1}^{M} b_j (b_j^T x_n) = \underbrace{\left(\sum_{j=1}^{M} b_j b_j^T\right)}_{\text{projection matrix}} x_n \qquad \text{Q} \quad (\tilde{x}_n: \text{orthogonal projection of } x_n \text{ onto subspace spanned by M basis vectors } b_j \text{ from } j=1 \dots M)$$

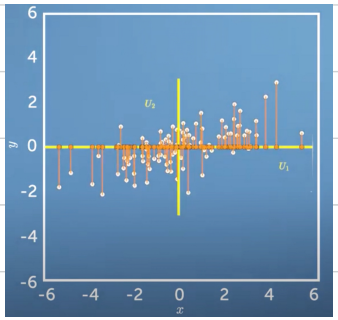
$$x_n = \left(\sum_{j=1}^{M} b_j b_j^T\right) x_n \qquad \text{(projection onto principal subspace)}$$

$$+\left(\sum_{j=M+1}^{D} b_j b_j^T\right) x_n \qquad \text{(projection onto orthogonal complement : missing from } \tilde{x}_n \text{, } \therefore \text{ approximation)}$$

$$x_n - \tilde{x}_n = \left(\sum_{j=M+1}^{D} b_j b_j^T\right) x_n \qquad \text{(displacement vector lies exclusively in subspace we ignore : orthogonal complement to principal subspace)}$$

difference between x_n and projection

$$= \sum_{j=M+1}^{D} (b_j^T x_n) b_j \quad \text{E}$$



project data onto U_1 subspace

vertical lines : difference between data and projection (difference vector)

↳ have no variation in x, only has component that lines in U_2 subspace which is orthogonal complement to U_1, which is subspace we projected onto

↙ avg. squared reconstruction error

Reformulate loss function: $J = \frac{1}{N}\sum_{n=1}^{N} \| x_n - \tilde{x}_n \|^2$

(E) $= \frac{1}{N}\sum_{n=1}^{N} \| \sum_{j=M+1}^{D} (b_j^T x_n) b_j \|^2$

ONB $= \frac{1}{N}\sum_{n=1}^{N}\sum_{j=M+1}^{D} (b_j^T x_n)^2$

$= \frac{1}{N}\sum_{n}\sum_{j} b_j^T x_n x_n^T b_j$

$= \sum_{j=M+1}^{D} b_j^T \underbrace{\left(\frac{1}{N}\sum_{n=1}^{N} x_n x_n^T\right)}_{\text{data covariance matrix } S \text{ (we assume centered data)}} b_j$

$J = \sum_{j=M+1}^{D} b_j^T S b_j = \text{trace}\left(\underbrace{\left(\sum_{j=M+1}^{D} b_j b_j^T\right)}_{\text{projection matrix}} S\right)$

(F)

↑ projects data covariance matrix onto orthogonal complement of principal subspace

↳ can reformulate loss function as variance of data projected onto subspace we ignore

↳ ∴ minimising (B) = minimising variance of data that lies in subspace orthogonal to principal subspace

↳ interested in retaining as much variance after projection as possible

Minimising avg. squared reconstruction error = minimising projection of variance of data by projecting onto subspace we will ignore in PCA

$J = \sum_{j=M+1}^{D} b_j^T S b_j$ (minimising this: need orthonormal basis that spans subspace we will ignore)

↳ then take orthogonal complement as basis of principal subspace

$b_1, b_2 \quad b_i^T b_j = \delta_{ij}$ (1 if $i=j$, 0 otherwise)

$J = b_2^T S b_2, \quad b_2^T b_2 = 1$ (constraint)

Solve optimisation using Lagrangian (Lagrange multiplier λ): $L = b_2^T S b_2 + \lambda(1 - b_2^T b_2)$

$$\frac{\partial L}{\partial \lambda} = 1 - b_2^T b_2 = 0 \iff b_2^T b_2 = 1$$

$$\frac{\partial L}{\partial b_2} = 2 b_2^T S - 2\lambda b_2^T = 0 \iff S b_2 = \lambda b_2$$

(b_2: eigenvector of covariance matrix; λ: eigenvalue; eigenvalue problem)

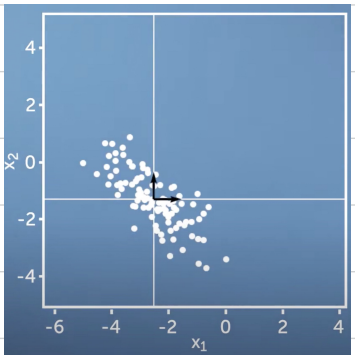
need to choose b_2 as corresponding eigenvector which will span subspace we will ignore

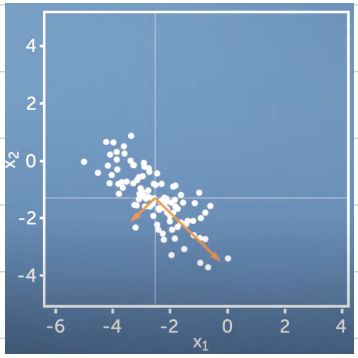
$$J = b_2^T S b_2 = \underbrace{b_2^T b_2}_{\text{orthonormal basis}} \lambda = \lambda$$

b_1 spans principal subspace: eigenvector corresponding to largest λ of covariance matrix

$\therefore$ avg. squared reconstruction error is minimised if λ is smallest eigenvalue of data covariance matrix

eigenvectors of covariance matrix already orthogonal due to matrix symmetry





best projection we can get which retains most info.

projects onto subspace spanned by eigenvector of covariance matrix which belongs to largest eigenvalue (which points in direction of largest variance)

↑ variance

↳ eigenvector belonging to 2nd largest eigenvalue points in direction of 2nd largest variance

General case: find M-dimensional principal subspace of D-dimensional dataset

b_j, $j = M+1, \dots, D$ (optimise these)

$Sb_j = \lambda_j b_j$, $j = M+1, \dots, D$

$$J = \sum_{j=M+1}^{D} \lambda_j$$

PCA

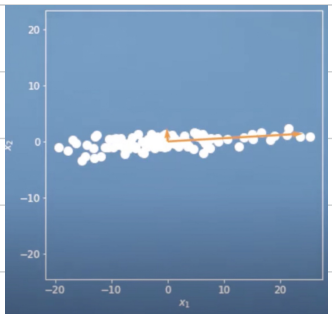
Subtracting the mean can avoid numerical difficulties

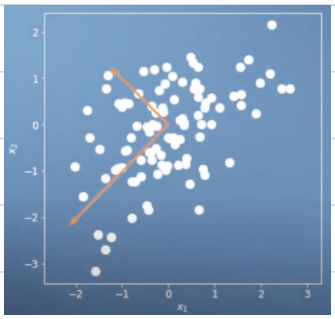
↳ e.g. values centered around 10^8 : covariance matrix requires multiplication of large nums, numerical instabilities

↓

then recommended to ÷ each dimension by corresponding σ : makes data unit free, variance in each dimension = 1

↳ does leave correlations intact





(unit variance)

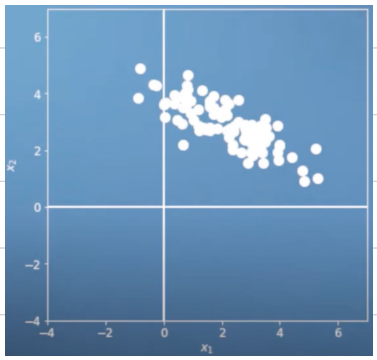
principal subspace of normalised dataset

strong correlation between 2 dimensions

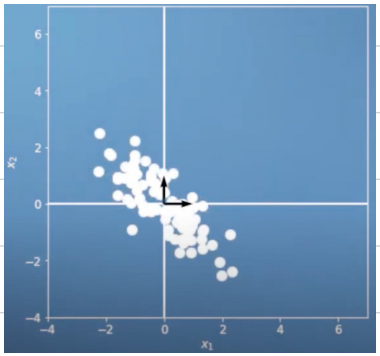
x_1 in cm, x_2 in m

(so naturally varies more)

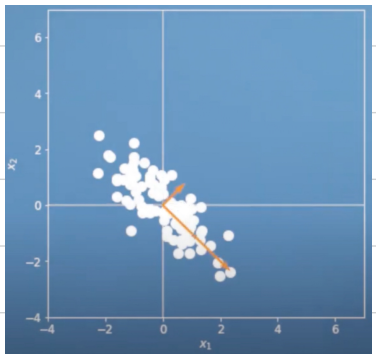
example:



(want to project onto 1D subspace)



(center and $\div \sigma$, data is unit free with $\sigma^2 = 1$ along each axis - black arrows)



(compute data covariance matrix and its eigenvalues and corresponding eigenvectors)

eigenvectors are scaled by magnitude of corresponding eigenvalue

longer vector spans principal subspace (U)

can then project any point x_* onto principal subspace

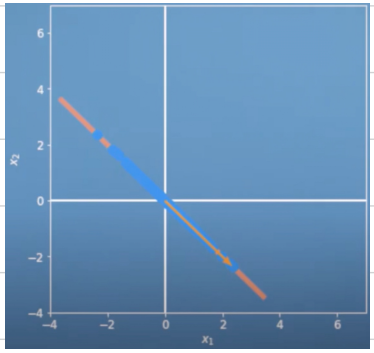
$$x_* \leftarrow \frac{x_*^{(d)} - \mu^{(d)}}{\sigma^{(d)}} \quad \text{(for every dimension in } x_*\text{)}$$

$$\tilde{x}_* = \pi_U(x_*) = BB^T x_*$$

$B^T x_*$: coordinates of projection w.r.t. basis of principal subspace

B: matrix of eigenvectors belonging to largest eigenvalues as columns

$\tilde{x}_*$: projection of x_* onto principal subspace U spanned by eigenvectors that belong to largest eigenvalues



projection of x^*

PCA in high dimensions

D-dimensions: covariance matrix is D×D matrix

↳ high D: computing eigenvectors + eigenvalues is expensive (scales cubically in no. of dimensions)

May have fewer data points than dimensions

$x_1, \dots, x_N \in \mathbb{R}^D$ (assume centered data) ← i.e. some dependencies

$S = \frac{1}{N} X^T X \Rightarrow \text{rank}(S) = N$, $D - N + 1$ eigenvalues $= 0$ (matrix not full rank, rows and cols. are linearly dependent)

$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix} \in \mathbb{R}^{N \times D}$

Turn D×D covariance matrix S to full rank N×N covariance matrix without eigenvalues = 0

$Sb_i = \lambda_i b_i$ ← basis vector of orthogonal complement of principal subspace

$\frac{1}{N} X^T X b_i = \lambda_i b_i$ (recover eigenvectors of S; b_i: eigenvector of S belonging to eigenvalue λ_i)

S

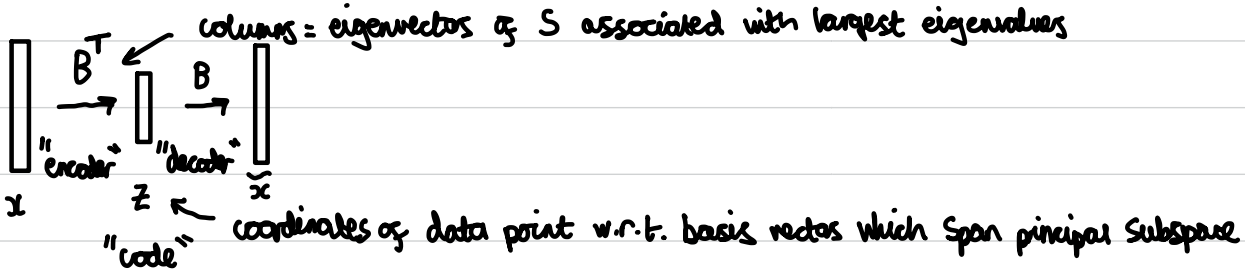
(multiply X) ⇒ $\frac{1}{N} X X^T X b_i = \lambda_i X b_i$ ⇒ $\frac{1}{N} X^T X X^T c_i = \lambda_i X^T c_i$

S

$\in \mathbb{R}^{N \times N}$; c_i; c_i

c_i: eigenvectors of matrix (has same non-zero eigenvalues as S, but now N×N, so can compute eigenvectors + eigenvalues much quicker than for original S)

Perspectives of PCA



Found PCA parameters s.t. reconstruction error between x and $\tilde{x}$ is minimised

Can think of PCA as linear autoencoder (i.e. encode x and try to decode to something similar to x)

If encoder + decoder are linear mappings: get PCA solution by minimising squared autoencoding loss

↳ replace with non-linear mapping: non-linear autoencoder

↳ e.g. deep autoencoder where linear functions of encoder + decoder replaced with DNN's

Information theory perspective: code can be seen as smaller, compressed version of x

↳ when reconstructing original data using code, don't get exact data back (slightly distorted / noisy version)

↳ lossy compression

↳ maximise correlation between original data and lower-D code

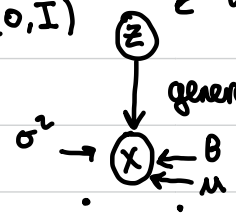
↳ related to mutual information (maximise)

Minimising variance = maximising variance of data when projected onto principal subspace

Interpret variance as info contained within data: PCA retains as much info as possible

$p(z) = N(0, I)$ ← unknown, lower-D code

Can look at PCA as latent variable model



(assume linear relationship)

$x = Bz + \mu + \epsilon$

$\epsilon \sim N(0, \sigma^2 I)$

Likelihood: $p(x|z) = N(x|Bz + \mu, \sigma^2 I)$

Marginal likelihood: $\int p(x|z)\,p(z)\,dz$

$= N(x|\mu, BB^T + \sigma^2 I)$

Use maximum likelihood estimation to find parameters:

- μ: mean of data
- B: matrix of eigenvectors corresponding to largest eigenvalues

To find z of data point: apply Bayes theorem to invert L.R. between z and x

↳ $p(z|x) = \dfrac{p(x|z)p(z)}{p(x)}$

5 different perspectives leading to different objectives:

- Minimising squared reconstruction error
- Minimising autoencoder loss
- Maximising mutual information
- Maximising variance of projected data
- Maximising likelihood in latent variable model

(all give same solution to PCA problem)