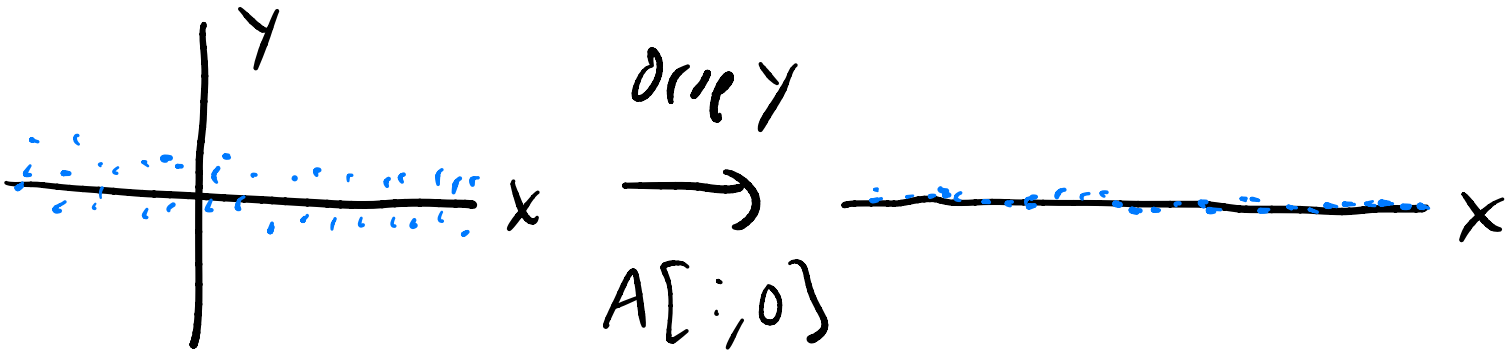
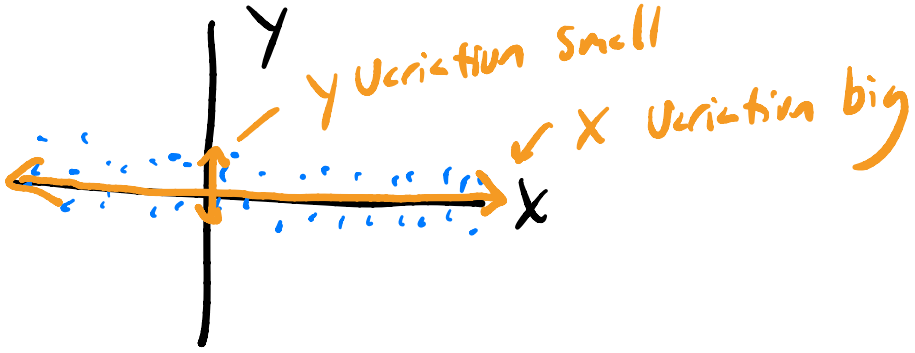
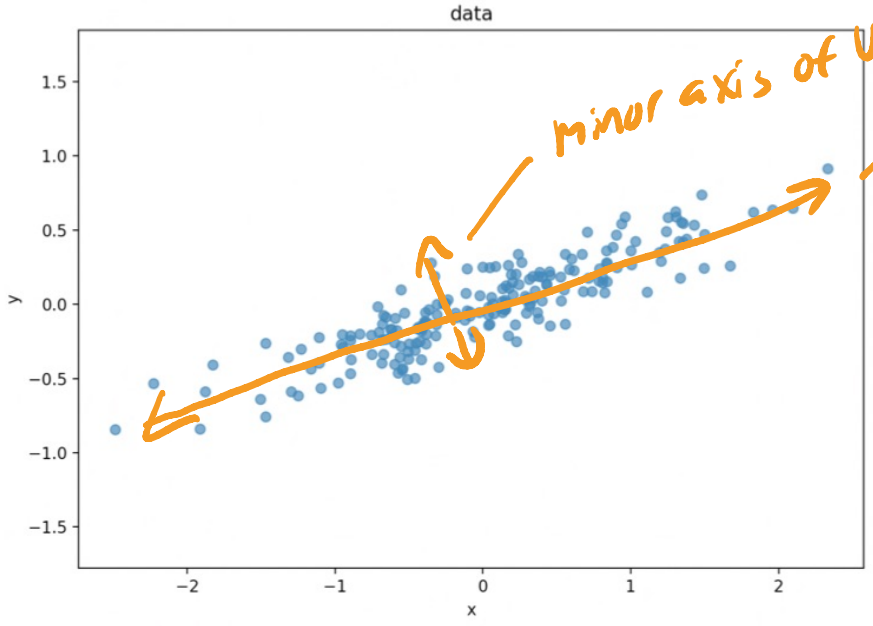
Principal component analysis (PCA)





y variation small

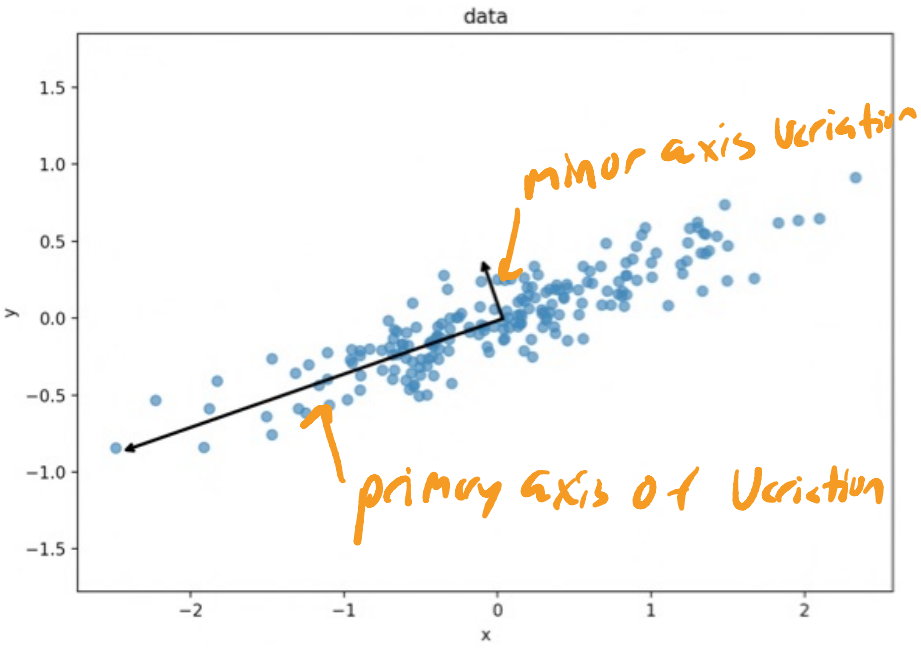
x variation big



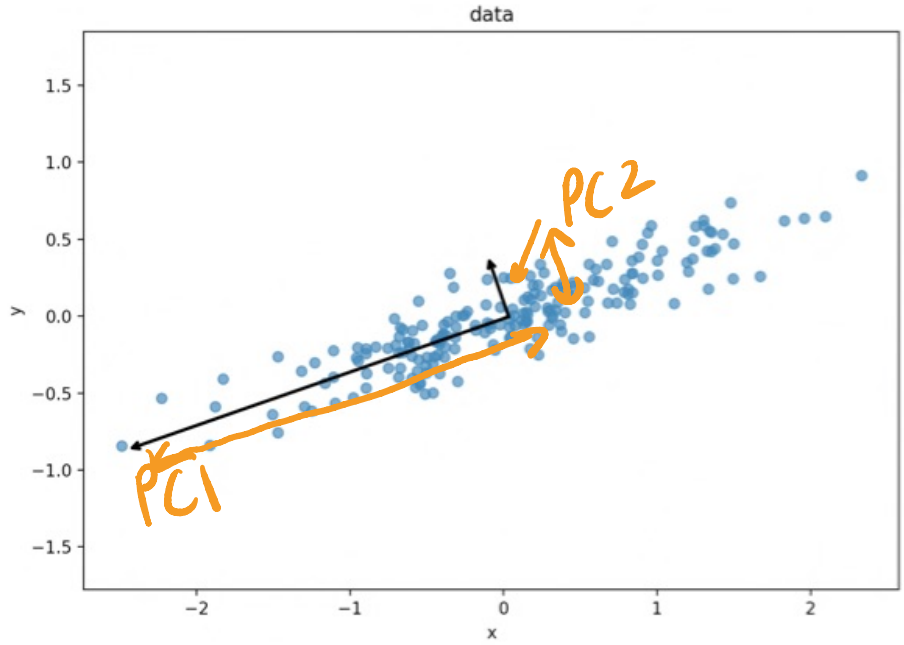
minor axis of variation] != Y axis

primary axis of variation

!= X axis



Principal components (PCs): intrinsic axes of variation in data

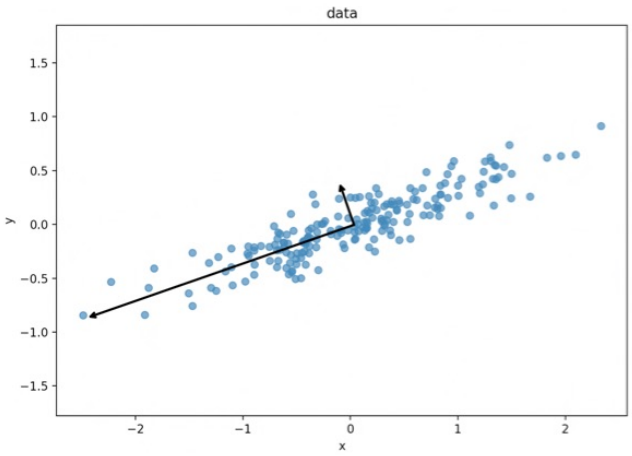


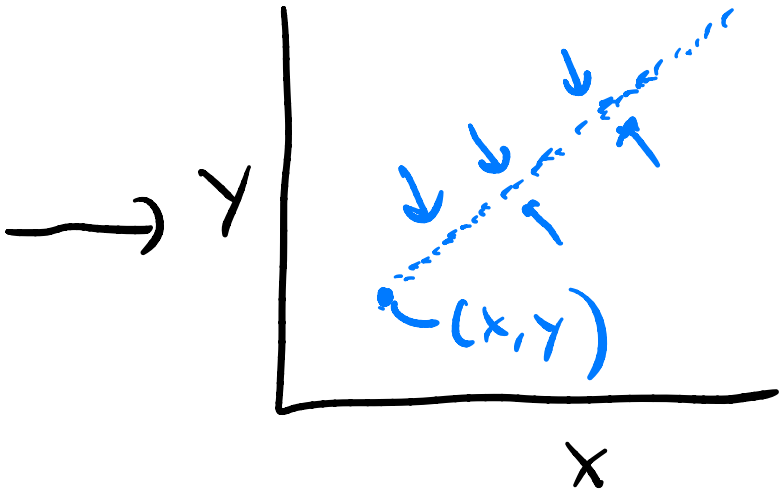
Smaller the PC number bigger the axis of variation.

Goal of PCA

We want to drop PCs intrinsic axes of variation that are insignificant / small

⇒ Do <u>not</u> want to drop data variables





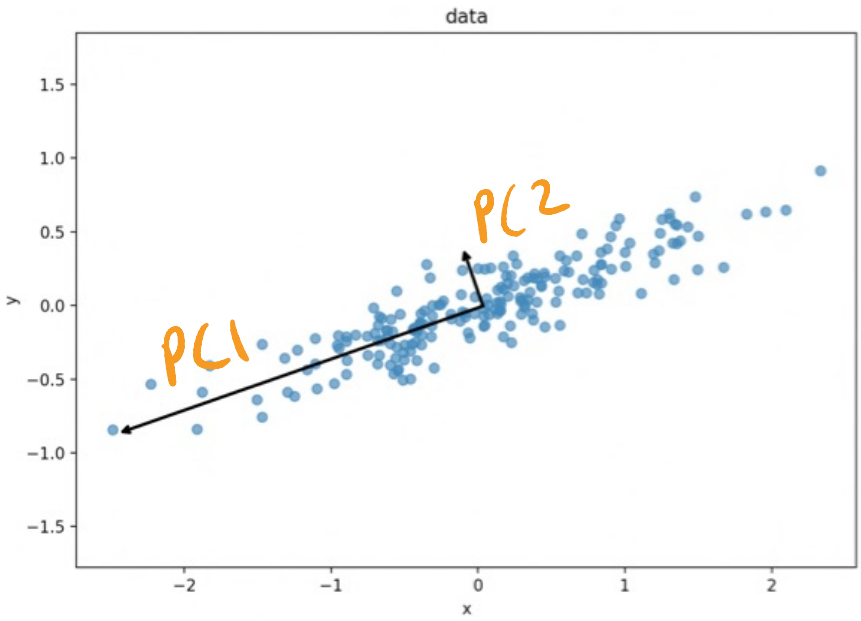
Lecture 18 | PCA (Continued)

principal component

data variable

PC1 != X

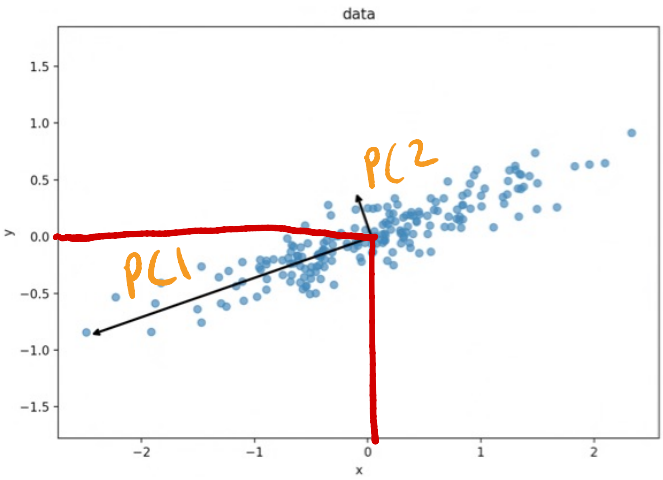
PC2 != Y



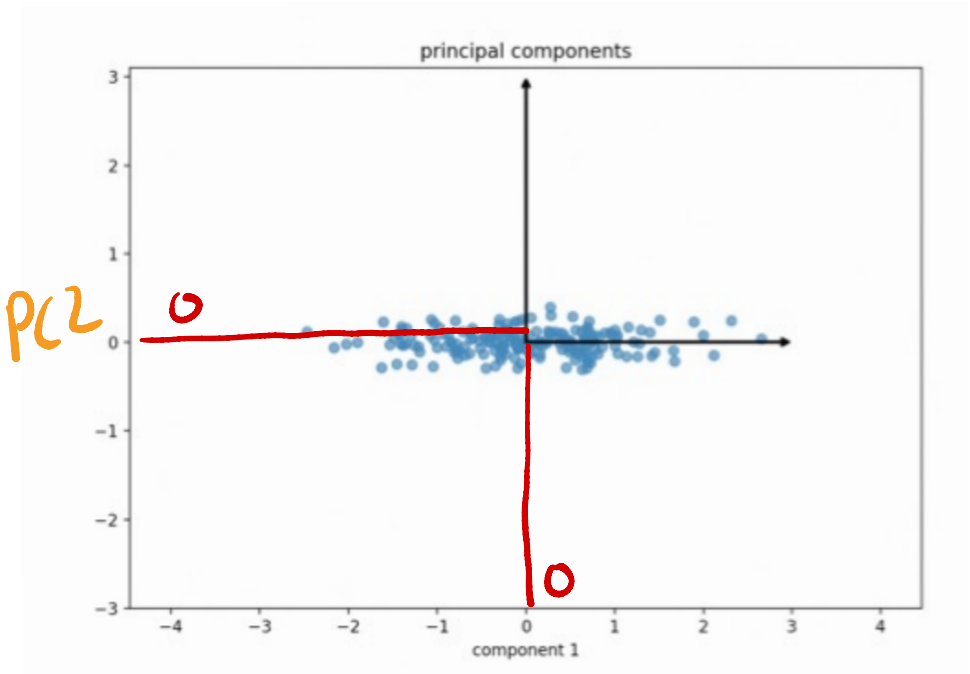
★ Want to rotate data so that PC1 becomes "X axis" and PC2 becomes "y axis"

Do that with a <u>translation</u> and multiplying by a <u>rotation matrix</u>.

A_c



Rotate



rotated data

$$\hat{A}_c = (P.T \,@\, A_c).T$$

$$\hat{A}_c = A_c \,@\, P$$

rotation matrix

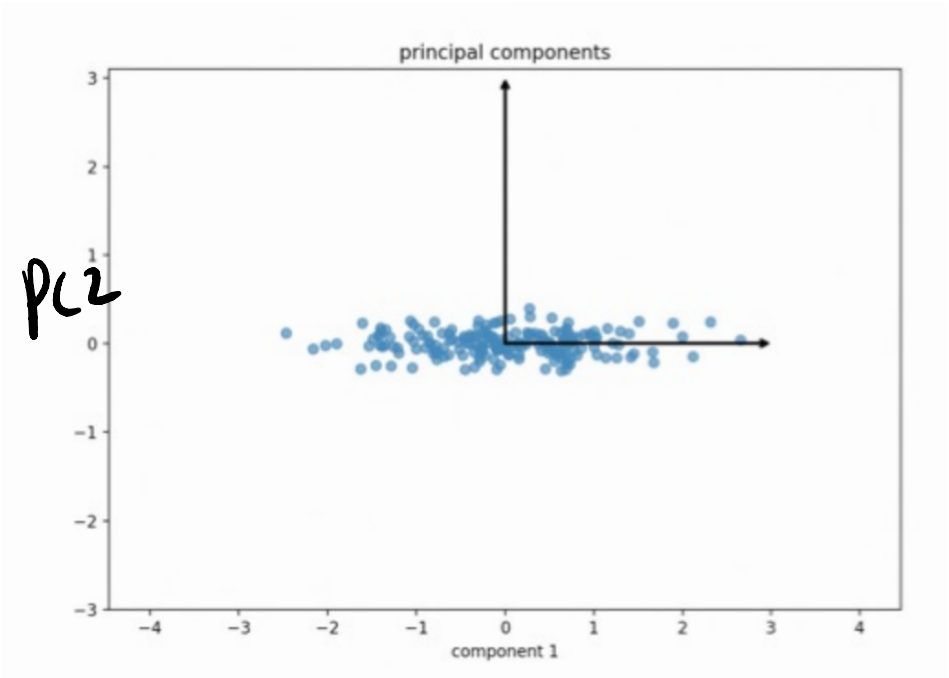
Centered data matrix

$A_c = A - \vec{\mu}$

translation

equal

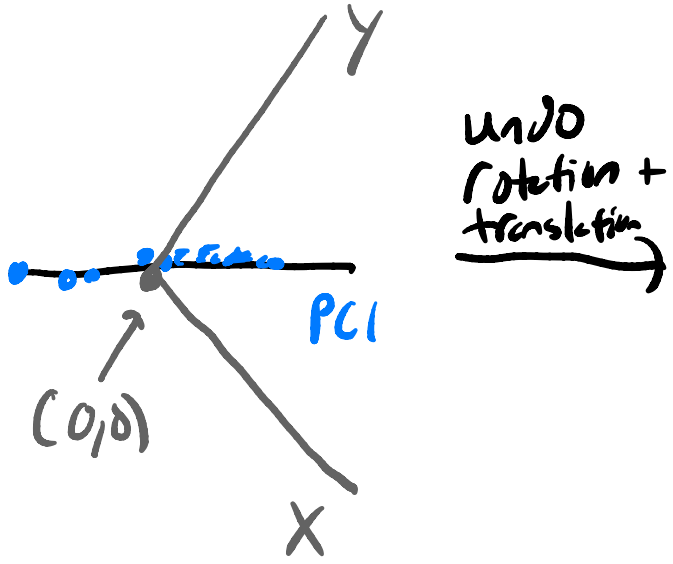
$\hat{A}_c$:



drop PC2

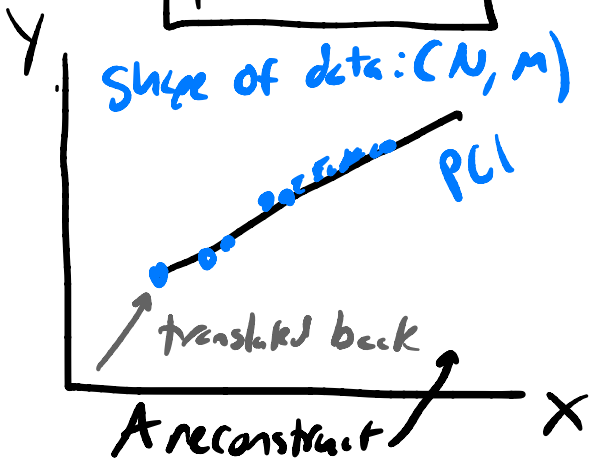
$\hat{A}_c[:, 0]$

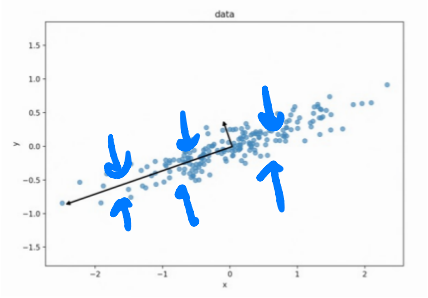
⇒ Can get back to original data variables — reConstruct data — get our "axes" to be original data axes again



undo rotation + translation

Final result





rotated data after PC2 removed

original data means

$$A_{reconstruct} = (\hat{P} @ \hat{A}_c.T).T + \vec{\mu}$$

Same

$$= \hat{A}_c @ \hat{P}.T + \vec{\mu}$$

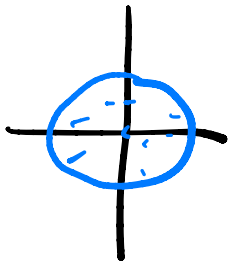
uncenter data

$A_c \rightarrow A$

What is the rotation matrix P?

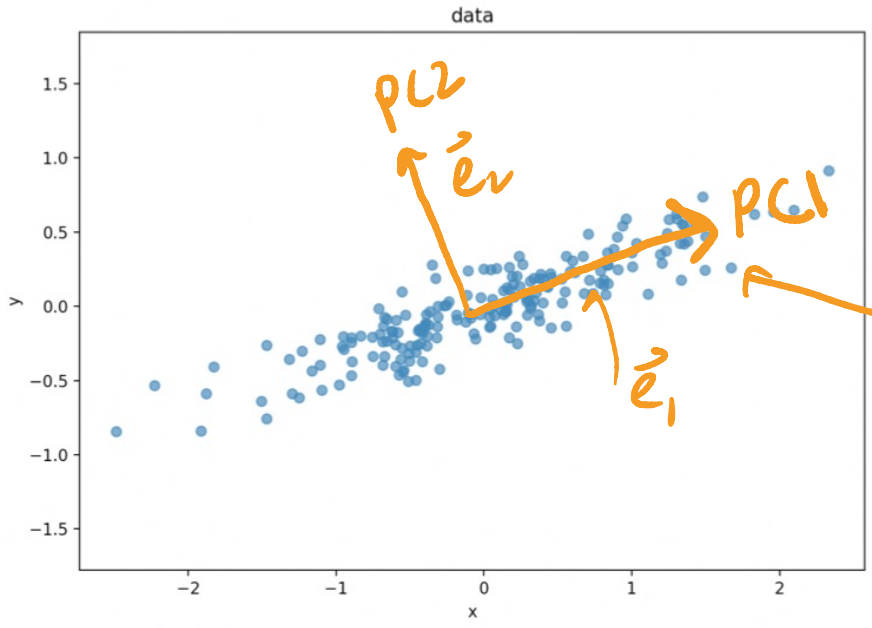
If we have Σ covariance matrix.

shape here: (2,2)



$\Sigma = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$

shape(M, M)



Eigenvectors of Σ: Vectors that tell us the direction of the PCs but not the amount of variation in the PCs.

e.g. # eigenvectors = # data vars = m

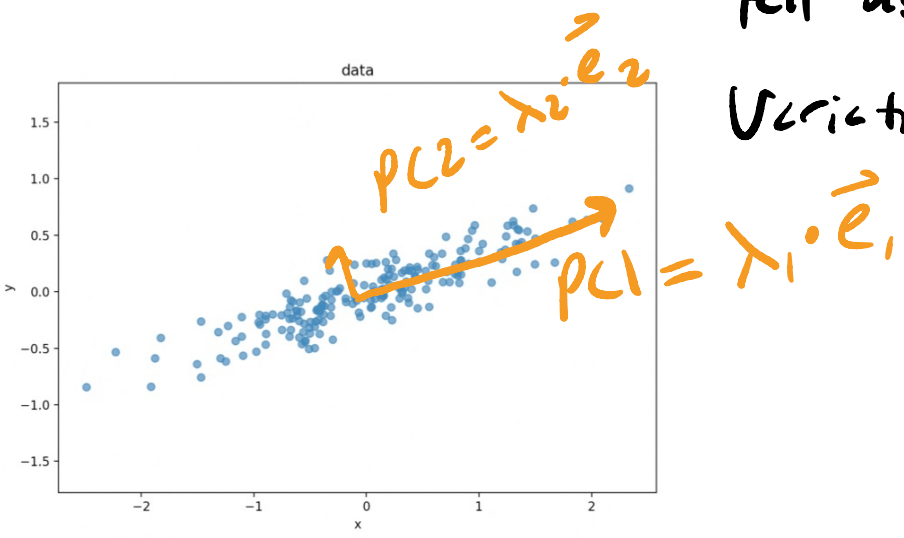
$\Rightarrow$ we have 2.

$\vec{e}_1, \vec{e}_2$

Eigenvalues of Σ: Scalars/floats — one number per eigenvector/data variable

$$e\_vals = [\underbrace{\lambda_1}_{2.5}, \underbrace{\lambda_2}_{1.2}]$$

tell us the amount of variation in each PC direction



In numpy:

```
e_vals, e_vecs = np.linalg.eig(Σ)
```

(Σ: covariance matrix)

e_vecs → P

P: cols of P

$$P = [\vec{e}_1, \vec{e}_2, \vec{e}_3, \ldots, \vec{e}_m]$$

Strategy for removing insignificant PCs:

$m = 4$

```
e_vals = [0.1, 3, 0.7, 5]
```

0.1: not much variation in this PC ← maybe safe to toss out

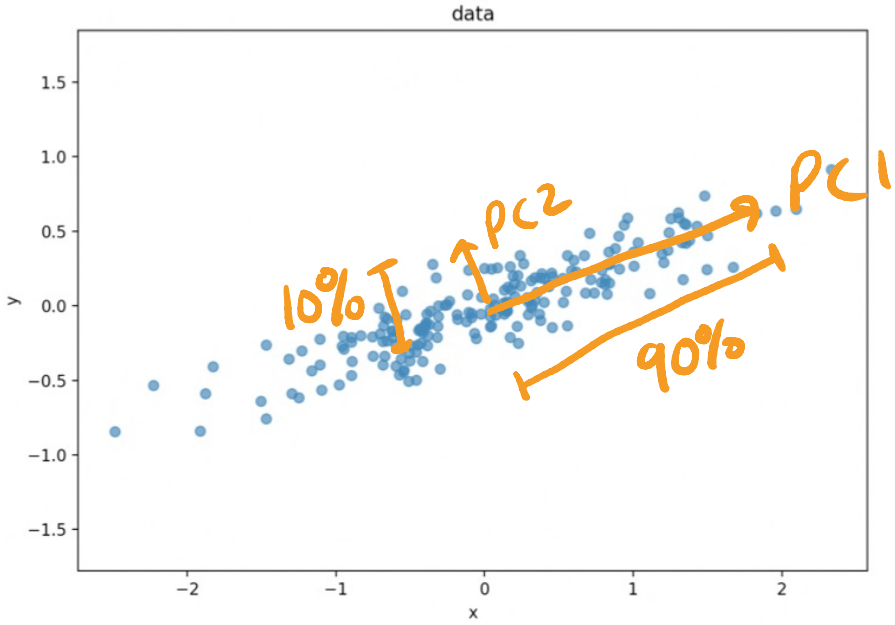
3, 0.7, 5: How much variation in this PC

→ Sort e_vals high → low:

```
e_vals = [5, 3, 0.7, 0.1]
          PC1  PC2  PC3  PC4
```

maybe good to drop PC3 & PC4.

proportion variance accounted for by each PC:



100% of data variance

PC1 + PC2

$$\text{total var} = \sum_{i=1}^{M} \lambda_i = 5 + 3 + 0.7 + 0.1 = 8.8$$

PC3 PC4

$$\text{prop-var} = \left[5/8.8,\ 3/8.8,\ 0.7/8.8,\ 0.1/8.8 \right]$$

57% 34% 8% 1%

Of total variance.

Sums to 100% in total

Define Variance Cut off threshold:

Want to keep 90% of the total variance.

⇒ helpful to Compute Cumulative Variance accounted for by PCs.

Drop these →

Cum_prop_Var = [5/8.8 , (5+3)/8.8 | (5+3+0.7)/8.8 , (5+3+0.7+0.1)/8.8]

PC1 | PC1-2 | PC1-3 | PC1-4

0.57 | 0.9 | 0.99 | 1.0

57% | 90% | 99% | 100%

Goal: keep at least 90% : keep PC1, PC2

Drop PC3, PC4