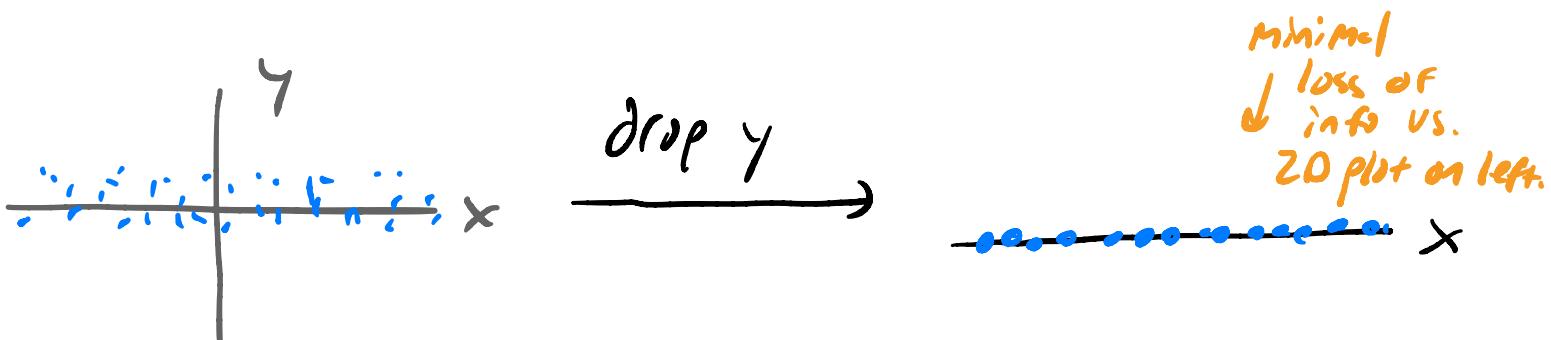
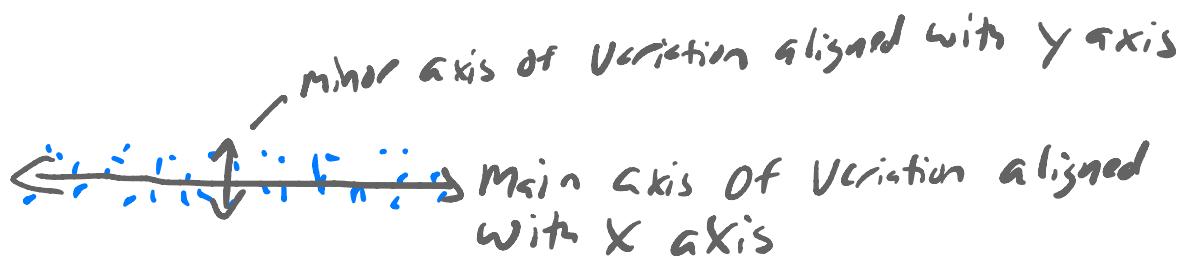


Principal Component Analysis (PCA)

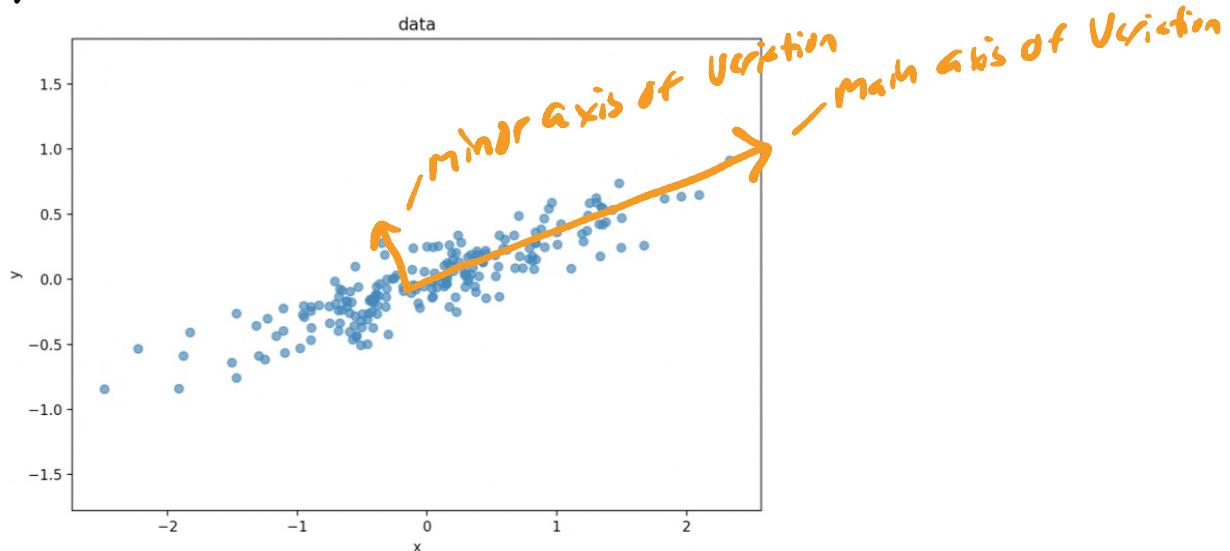
Recall the example where we could toss out a Variable (y) without lossing much information:



In this case, the main axes of variation of data are aligned with the coordinate axes



\Rightarrow clearly this is not true in general! Axes of variation do not need to correspond to variables in dataset
— e.g. $x, y, \text{etc.}$:

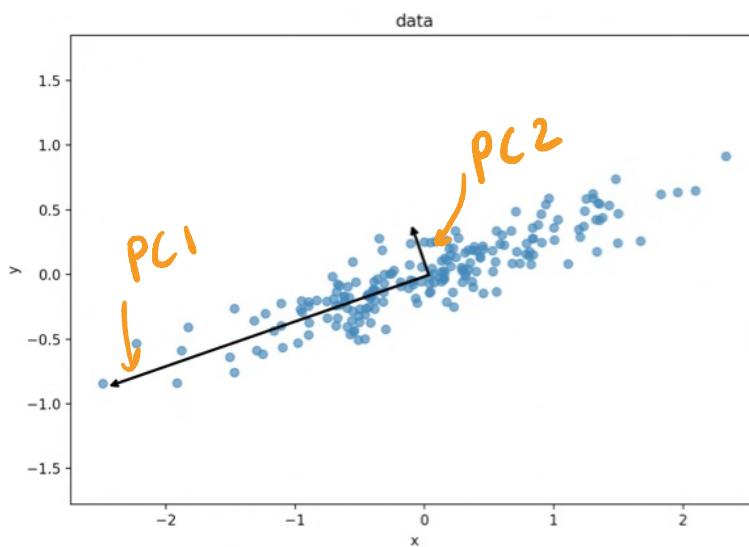


Above Main Variation Axis != x axis!
minor Variation axis != y axis!

We call these intrinsic axes of variation in data
principal components (PCs)

⇒ We number them in order based on the amount
of variation along each intrinsic axis

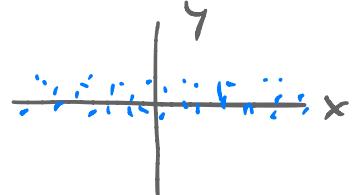
⇒ Larger variation → Smaller number.



Goal of PCA

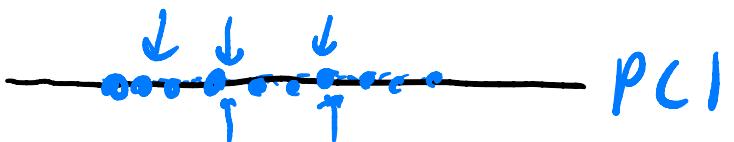
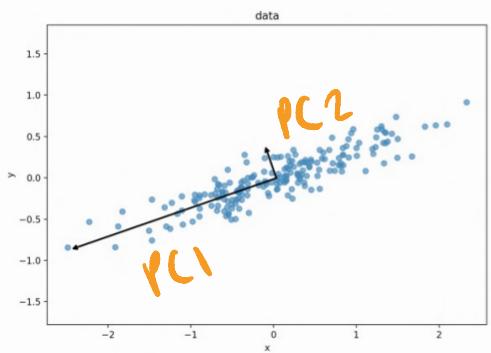
Reduce dimensionality of dataset
(e.g. $M=500 \rightarrow 30$) by dropping
intrinsic axes with little variation
(least amount of info lost)

e.g. PC2 above, y variable in



As we do not want to drop data variables

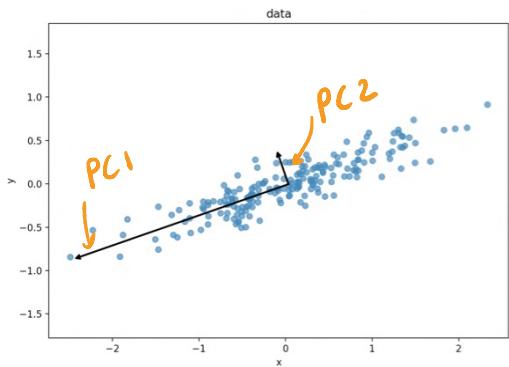
e.g. example drop PC 2.



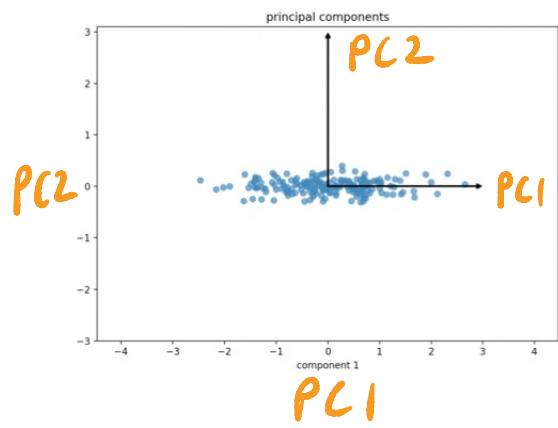
How do we "access" the PCs? — they can be oriented in any direction (but they are assumed to be orthogonal to each other).

⇒ Rotate the data so that the PCs become coordinate axes:

New coordinate axes; The axes of primary variation in the data



Rotate
→



How is this done? A translation then a rotation matrix applied:

See plots above with centered data at $(0,0)$

[Rotated, centered data]

rotation matrix

centered data matrix: $A_c = A - \bar{\mu}$

translation

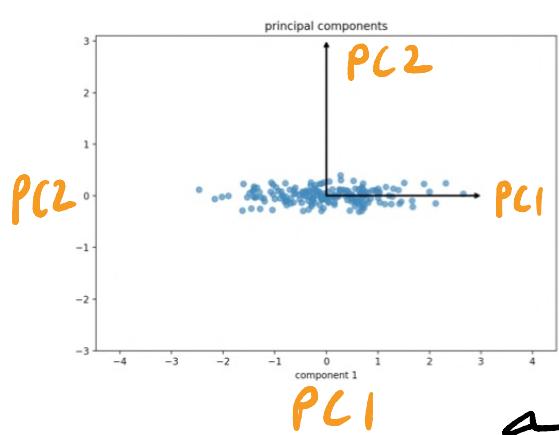
$\hat{A}_c = (P \cdot T @ A_c) \cdot T$

or more concisely:

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

Same thing

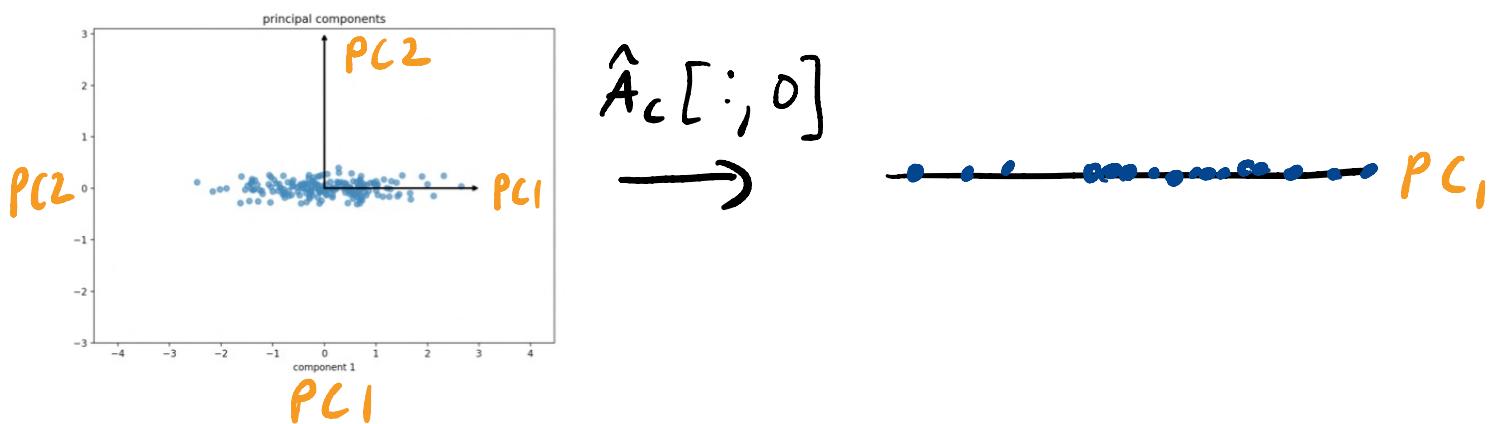
\hat{A}_c



\hat{A}_c is centered data in PCA Space: coordinate axes are PC1 & PC2, not X, Y

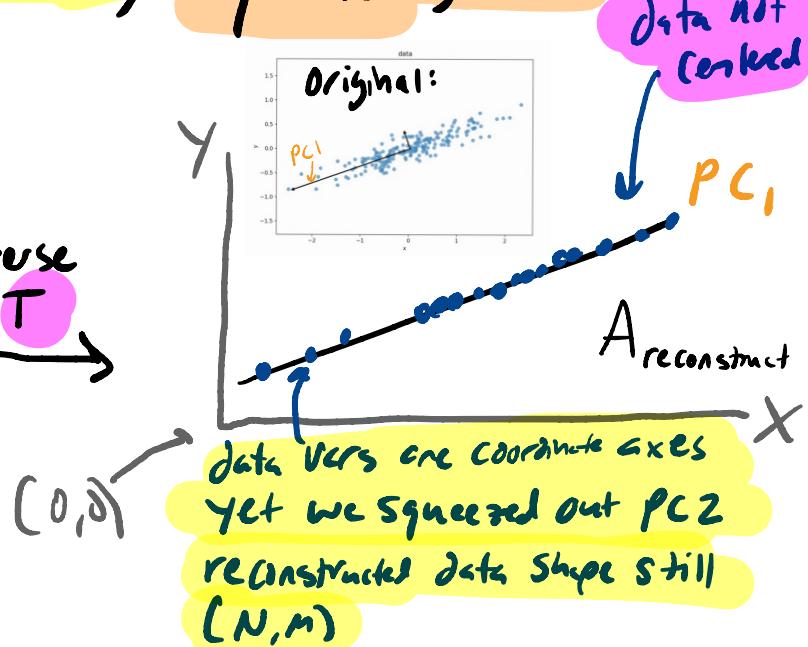
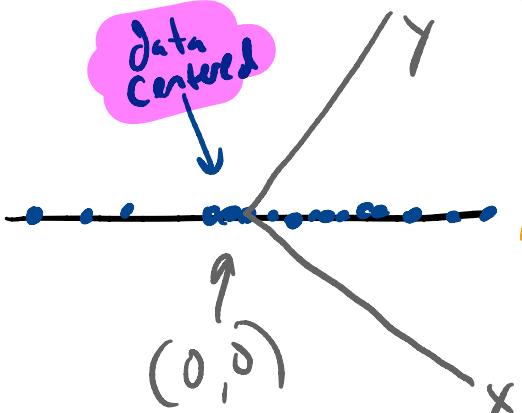
... and to drop the insignificant PCs, we can project onto the large meaningful ones (e.g. PC1)

Remember: project = index vers — e.g. $\hat{A}_c[:, 0]$



This achieves what we wanted, but the coordinate axes are PCs — not data variables, which might not be what we want.

We can express this result in terms of the original data variables (reconstruct the data) by undoing the rotation + translation!



To reverse R & T and reconstruct data from \hat{A}_c :

rotation matrix rotated data with
 insignificant PCs dropped
 e.g. PC2

uncenters data by adding back
each Var's mean
ie. $A_c \rightarrow A$

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

gives us A_c

Same — use either equation

Q: What is the rotation matrix P ?

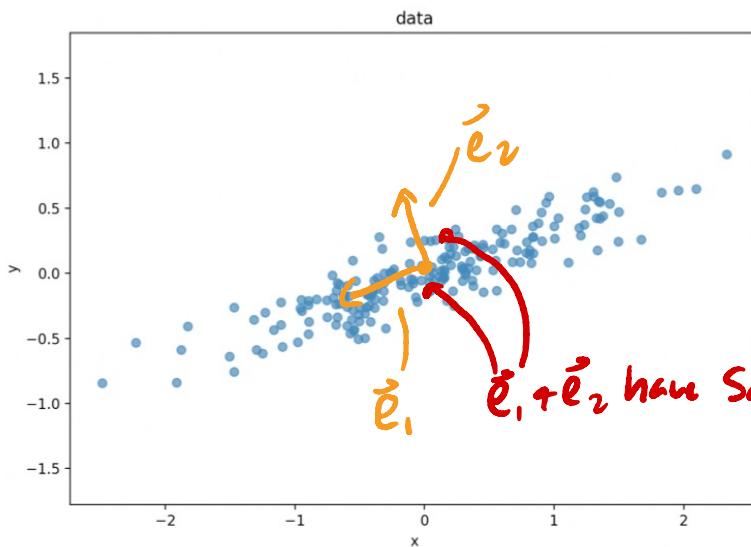
P is the eigenvectors of the Covariance matrix Σ of the data.

Recall: $\Sigma = \frac{1}{N-1} A_c @ A_c^T$

Shape: (M, M)

M of them — one per data variable

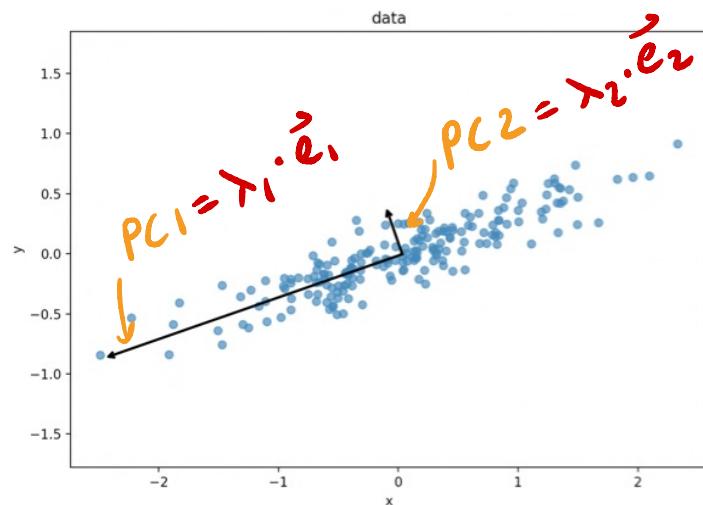
Eigen Vectors $P = [\vec{e}_1, \vec{e}_2, \dots, \vec{e}_n]$ tell us the direction but not amount of the corresponding PCs.



As rotating data by P aligns coordinate axes with PCs

How do we know the amount of Variation in each PC

Direction? Eigenvalues = $[\lambda_1, \lambda_2, \dots, \lambda_n]$



M Scales/floats
One per PC/eigen vector / data Variable.

$\leftarrow \lambda_1$ is much bigger than λ_2 !

IV Computing eigenvectors and eigenvalues

In Numpy:

$$e_vals, e_vecs = \text{np. linalg. eig}(\Sigma)$$

Computes the Spectral decomposition of Σ

P is an orthogonal matrix (forms orthogonal basis)

V Strategy for removing insignificant PCs:

e_vals tell us amount of variation in a PC
 \Rightarrow if small, we can remove.

e.g. $e_vals = [0.1, 3, 0.7, 5]$

\Rightarrow Sort high \rightarrow low:

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

These account for small part of overall data variation, might be good candidate to remove PC 3 & PC 4 here.

★ How many PCs to drop / what is significant is up to us to decide.

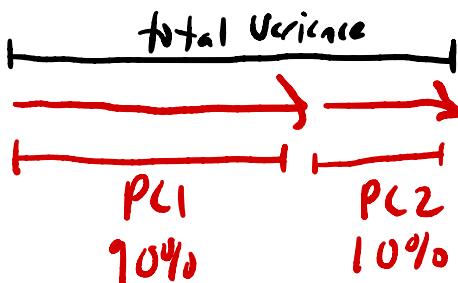
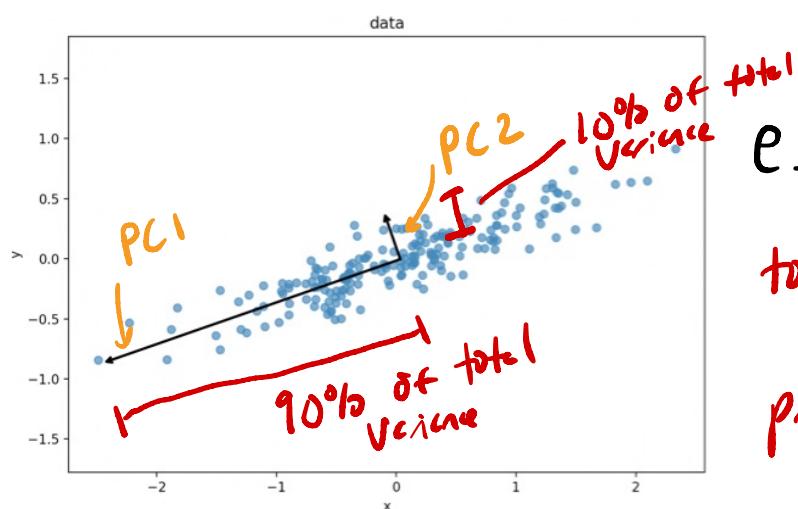
⇒ useful to determine the

proportion of total Variance Accounted for by each PC

$$e\text{-Vals} = [5, 3, 0.7, 0.1]$$

(expressing each as a fraction of the total Variance "pot" more helpful)

these numbers have little meaning on their own



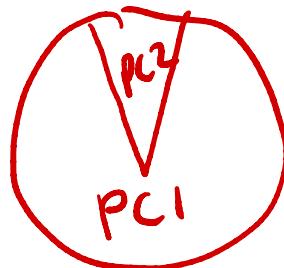
★ prop-Var must sum to 100% (or 1).

$$e\text{-Vals} = [5, 3, 0.7, 0.1]$$

$$\begin{aligned} \text{total Var} &= 5 + 3 + 0.7 + 0.1 \\ &= 8.8 \end{aligned}$$

$$\begin{aligned} \text{prop-Var} &= \left[\underbrace{\frac{5}{8.8}}, \underbrace{\frac{3}{8.8}}, \underbrace{\frac{0.7}{8.8}}, \underbrace{\frac{0.1}{8.8}} \right] \\ &\quad \text{57\%} \quad \text{34\%} \quad \text{8\%} \quad \text{1\%} \end{aligned}$$

$$\text{Sum} = 100\%$$



Often helpful to define a **Variance - kept Cut-off (threshold)** — e.g. toss out PCs to keep at least 90% of the total Variance in the data.

⇒ Compute cumulative Variance accounted for :

Cum-Vcr =

$$\begin{aligned} & \left[\frac{5}{8.8}, \frac{3}{8.8} + \frac{5}{8.8}, \frac{0.7}{8.8} + \frac{3}{8.8} + \frac{5}{8.8}, \frac{0.1}{8.8} + \frac{0.7}{8.8} + \frac{3}{8.8} + \frac{5}{8.8} \right] \\ &= \left[0.57, 0.9, \underbrace{0.99}_{\text{Keep}}, \underbrace{1.0}_{\text{Discard}} \right] \end{aligned}$$

after accounting for all PCs including this smallest one, total Vcr must be 1.0 i.e. 100%.

To keep 90% of the information in data/Variance, only keep top 2 PCs.