

APS1070

Foundations of Data Analytics and
Machine Learning

Winter 2022

Week 7:

- *Projections*
- *Matrix Decompositions*
- *Principal Component Analysis (PCA)*



Slide Attribution

These slides contain materials from various sources. Special thanks to the following authors:

- Scott Sanner
- Ali Hadi Zadeh
- Jason Riordon

Last Time

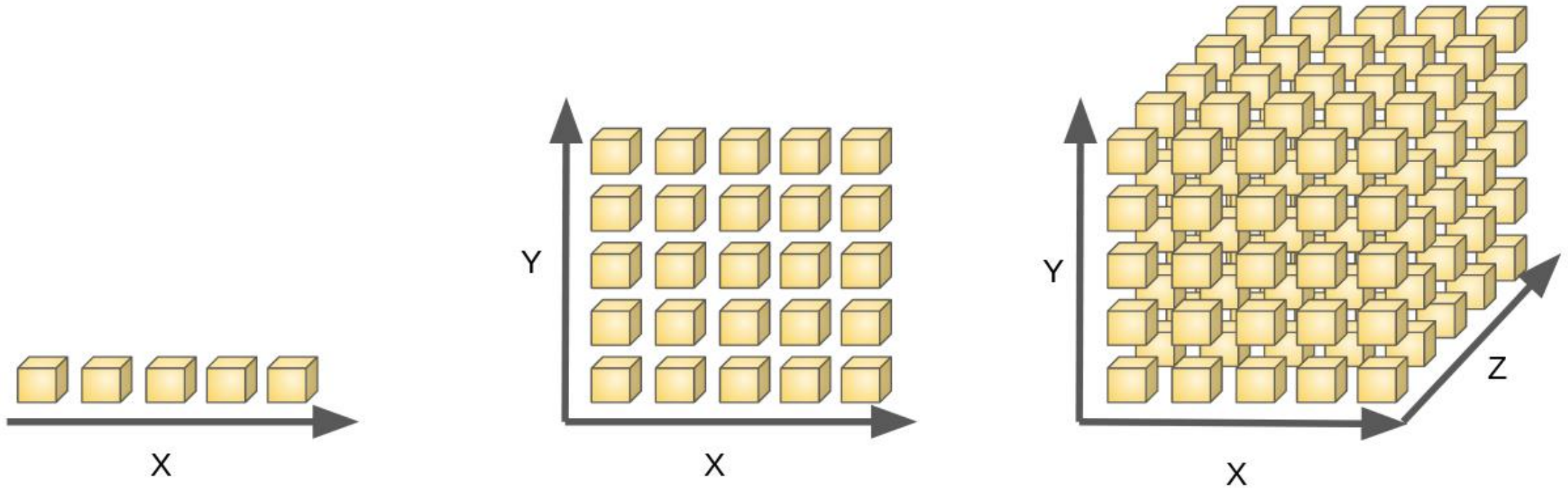
- Looked at linear algebra for data processing in the form of **Data Augmentation**
 - scalars, vectors and matrices
 - solving systems of equations
 - change of basis
 - analytical geometry
 - rotations and other transformations
- Today we will examine data processing in the form of **Dimensionality Reduction**

High-Dimensional Data



- Real world data is often high-dimensional!
- Challenge: difficult to **analyze**, **visualize** and **interpret**

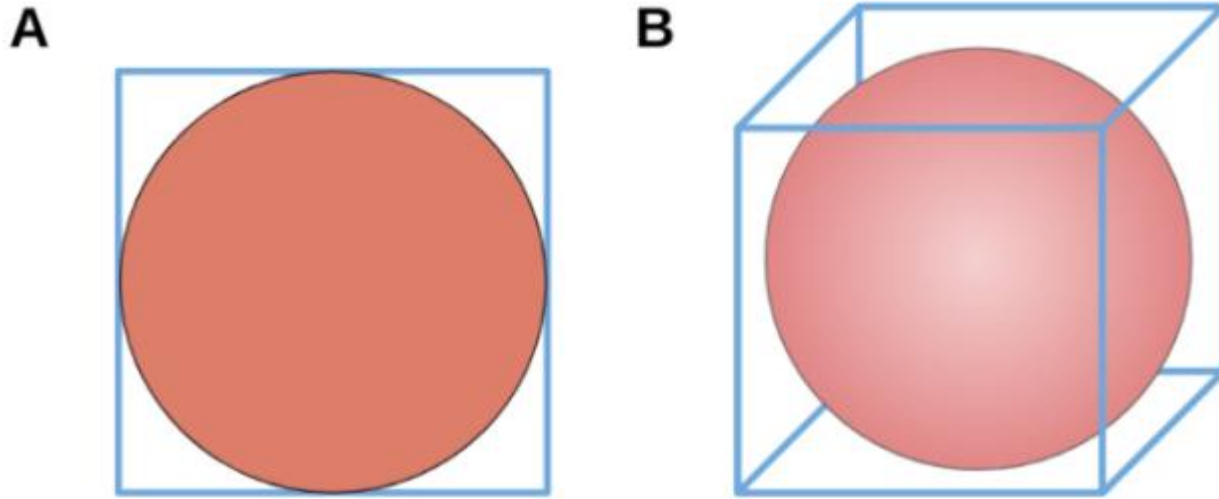
Curse of Dimensionality



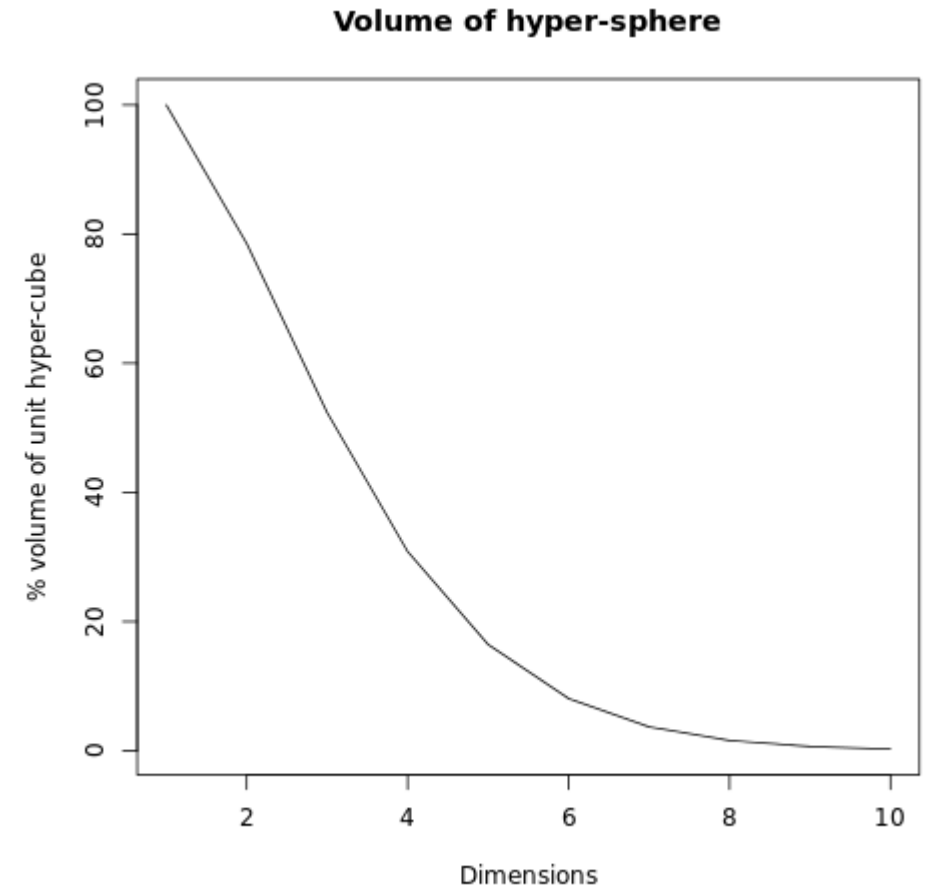
Source: [Peter Gleeson](#)

- As the number of feature or dimensions grows, the amount of data we need to generalize accurately grows exponentially.

Curse of Dimensionality



- Majority of the volume is outside of the sphere in higher dimensions



Curse of Dimensionality

- Many dimensions are unnecessary
- Data often lives on a low-dimensional manifold
- Dimensionality reduction finds the relevant dimensions

Agenda

- Projections
- Matrix Decompositions
 - Determinant
 - Trace
 - Eigenvalues and Eigenvectors
 - Eigendecomposition
- PCA
- Dimensionality Reduction



Theme:
**Dimensionality Reduction
Part 1**

Recap: Linear Algebra

- Given A and b , we want to solve for x :

$$Ax = b \quad \begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

- This can be given several interpretations:

- **By rows:** x is the intersection of hyper-planes:

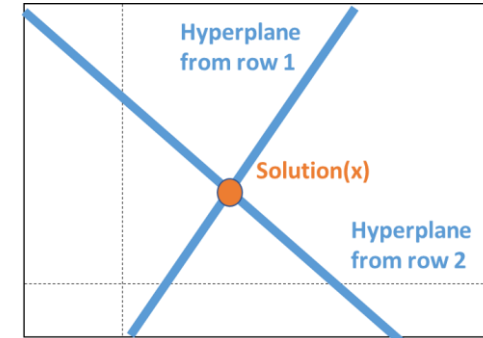
$$\begin{aligned} 2x - y &= 1 \\ x + y &= 5 \end{aligned}$$

- **By columns:** x is the linear combination that gives b :

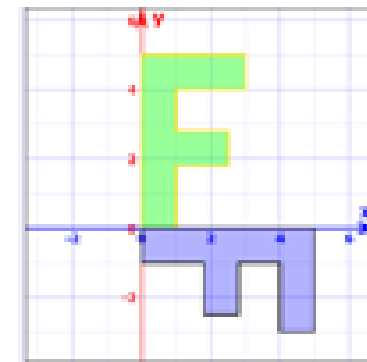
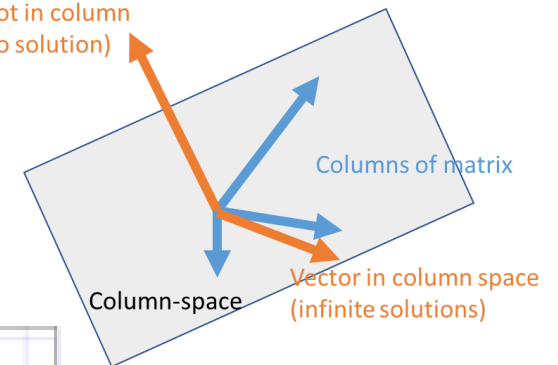
$$x \begin{bmatrix} 2 \\ 1 \end{bmatrix} + y \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

- **Transformation:** x is the vector transformed to b :

$$T(x) = b$$



Vector not in column space (no solution)



Recap: Analytical Geometry

- Projections are linear transformations that project to lower dimensional feature space

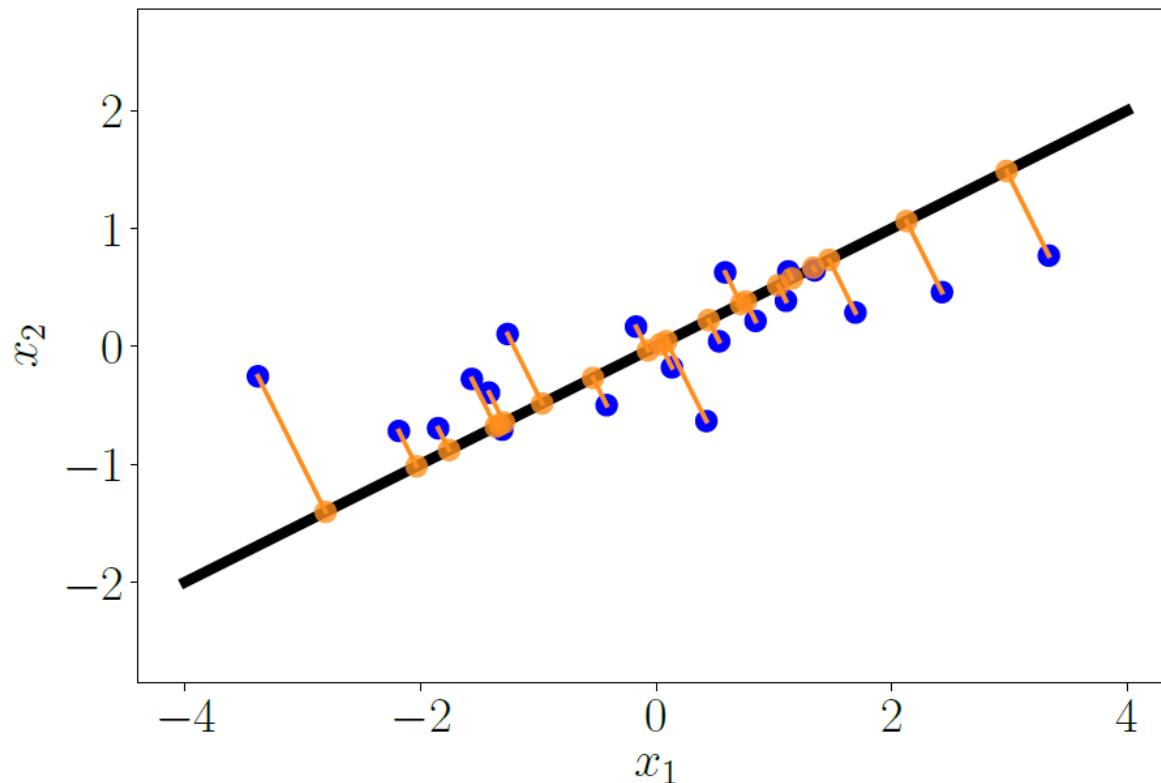
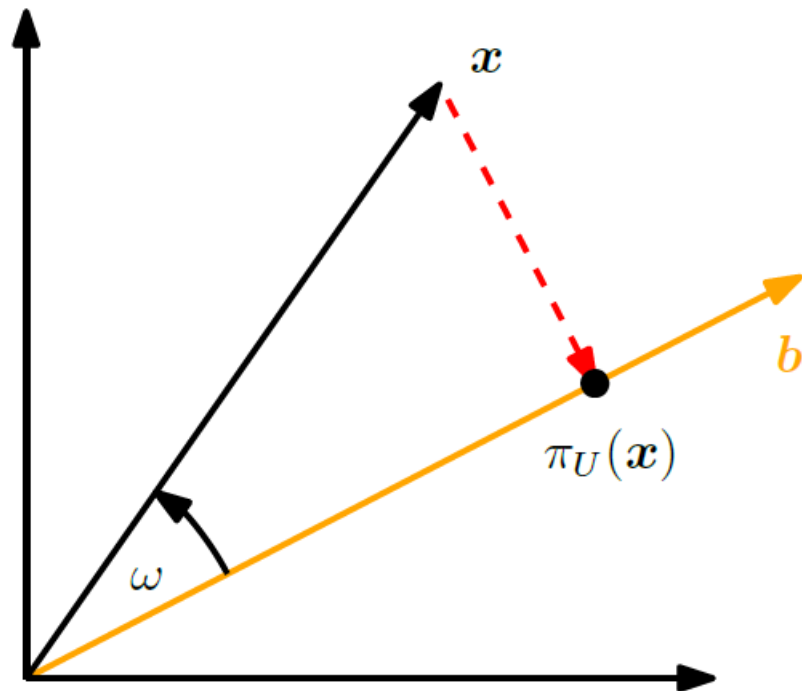


Figure 3.9
Orthogonal projection (orange dots) of a two-dimensional dataset (blue dots) onto a one-dimensional subspace (straight line).

Recap: Orthogonal Projections

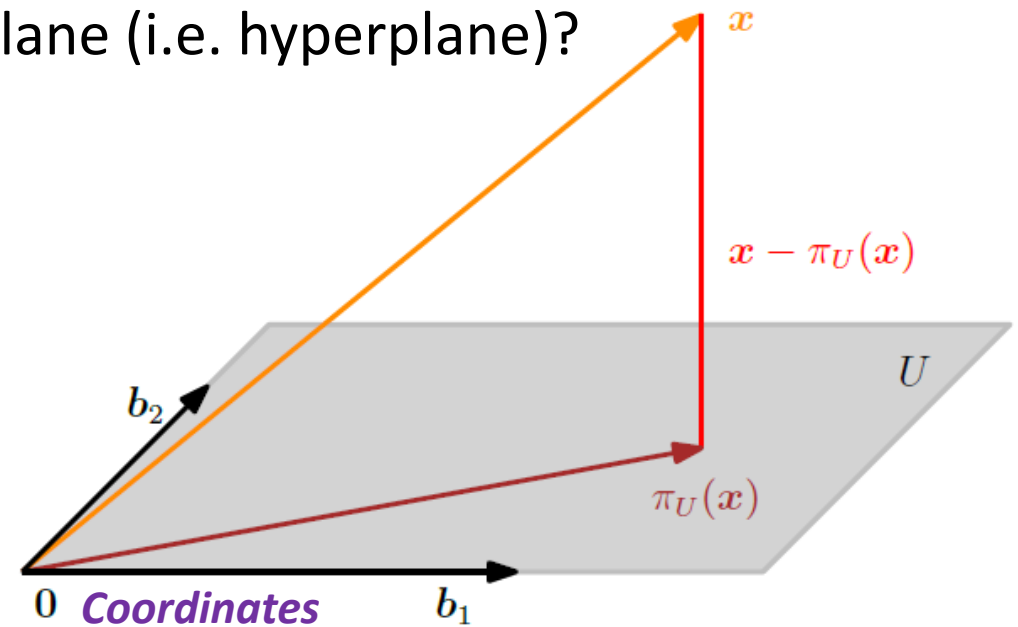
Project vector onto another vector



Coordinate

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

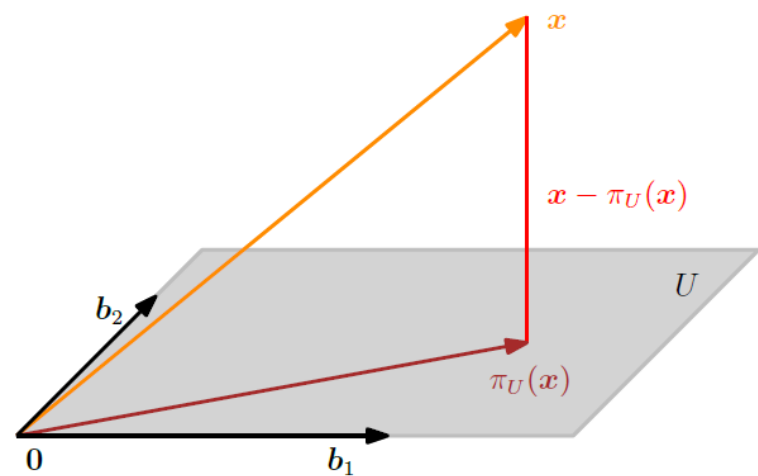
Now let's consider the case of projecting a vector onto a plane (i.e. hyperplane)?



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

Example

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



Example 3.11 (Projection onto a Two-dimensional Subspace)

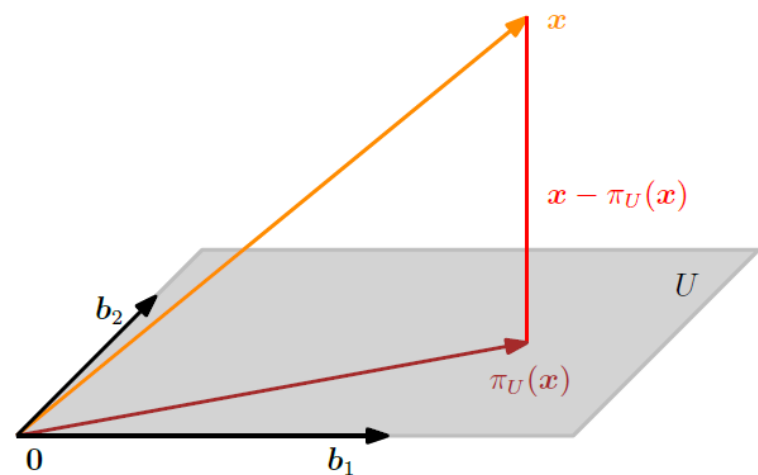
For a subspace $U = \text{span}\left[\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}\right] \subseteq \mathbb{R}^3$ and $x = \begin{bmatrix} 6 \\ 0 \\ 0 \end{bmatrix} \in \mathbb{R}^3$ find the coordinates λ of x in terms of the subspace U , the projection point $\pi_U(x)$ and the projection matrix P_π .

First, we see that the generating set of U is a basis (linear independence) and write the basis vectors of U into a matrix $B = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}$.

Second, we compute the matrix $B^T B$ and the vector $B^T x$ as

$$B^T B = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 3 & 5 \end{bmatrix}, \quad B^T x = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} 6 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \end{bmatrix}. \quad (3.60)$$

Example con't



Third, we solve the normal equation $B^\top B\lambda = B^\top x$ to find λ :

$$\begin{bmatrix} 3 & 3 \\ 3 & 5 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \end{bmatrix} \iff \lambda = \begin{bmatrix} 5 \\ -3 \end{bmatrix}. \quad (3.61)$$

Fourth, the projection $\pi_U(x)$ of x onto U , i.e., into the column space of B , can be directly computed via

$$\pi_U(x) = B\lambda = \begin{bmatrix} 5 \\ 2 \\ -1 \end{bmatrix}. \quad (3.62)$$

The corresponding *projection error* is the norm of the difference vector between the original vector and its projection onto U , i.e.,

$$\|x - \pi_U(x)\| = \left\| \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}^\top \right\| = \sqrt{6}. \quad (3.63)$$

Fifth, the projection matrix (for any $x \in \mathbb{R}^3$) is given by

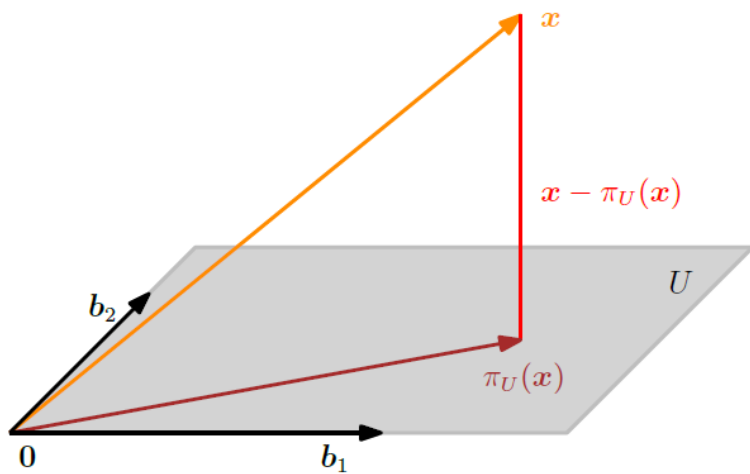
$$P_\pi = B(B^\top B)^{-1}B^\top = \frac{1}{6} \begin{bmatrix} 5 & 2 & -1 \\ 2 & 2 & 2 \\ -1 & 2 & 5 \end{bmatrix}. \quad (3.64)$$

To verify the results, we can (a) check whether the displacement vector $\pi_U(x) - x$ is orthogonal to all basis vectors of U , and (b) verify that $P_\pi = P_\pi^2$ (see Definition 3.10).

Projection onto a subspace with orthonormal bases

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

If we have orthonormal bases (B), then this simplifies the calculation of the inverse because $B^T = B^{-1}$



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

Google Colab Code

“how to summarize matrices, how matrices can be decomposed, and how these decompositions can be used for matrix approximations”

Matrix Decompositions

Readings:

- Chapter 4.1-5 MML Textbook

Decomposing a Matrix - 1

- There are many ways we can describe a matrix
- Example, take a look at the following transformation:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Notice that we could write the same thing in terms of several transformations:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 5 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Decomposing a Matrix - 2

- There are many ways we can describe a matrix
- Example, take a look at the following transformation:

$$A = \begin{bmatrix} 2 & -2 \\ 2 & 2 \end{bmatrix}$$

- Notice that we could write the same thing in terms of several transformations:

$$A = \begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Decomposing a Matrix

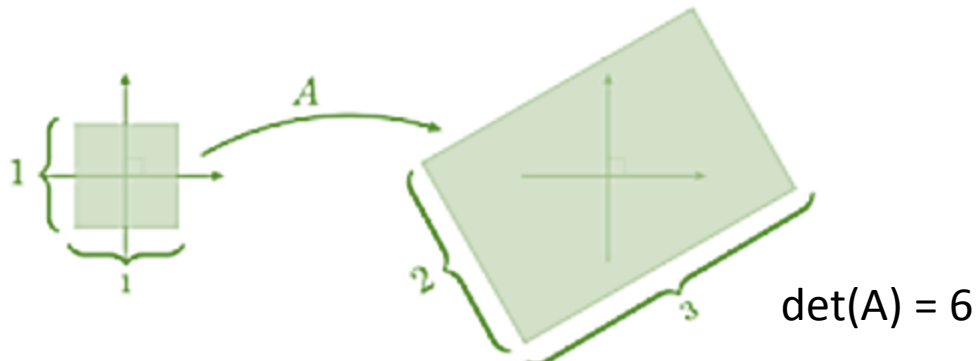
- This can be used for providing clarity on what the transformation is doing
- Simplifying and speeding up computations
- Dimensionality reduction as we'll see later

Key Concepts

- ☐ Determinant
- ☐ Trace
- ☐ Diagonalization
- ☐ Eigenvectors
- ☐ Eigenvalues

Determinant

- The determinant of a square matrix is a function that maps A onto a real number.
- Simple geometric meaning.
 - calculates how volume grows/shrinks under a linear transformation/mapping
 - for example:

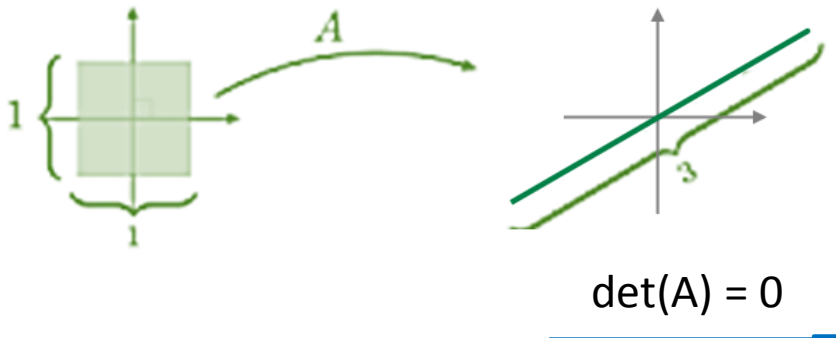


$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{vmatrix}$$

Two vertical bars indicate determinant of matrix enclosed inside

Determinant for Matrix Invertibility

- What does it mean when a matrix is not invertible?
- Simple geometric meaning.
 - transformation to a subspace
 - how does the volume grows/shrinks?
 - for example:



Note:

If $\det(A) = 0$, then matrix A is singular.

Determining the Determinant

➤ The determinant can only be calculated on square matrices (n x n):

➤ Determinant in 2d:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

➤ Determinant in 3d:

recursively take alternating
sums of sub-determinants

$$\begin{aligned} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} &= a_{11} \begin{bmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{bmatrix} - a_{12} \begin{bmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{bmatrix} + a_{13} \begin{bmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \\ &= a_{11}a_{22}a_{33} + a_{21}a_{32}a_{13} + a_{31}a_{12}a_{23} \\ &\quad - a_{31}a_{22}a_{13} - a_{11}a_{32}a_{23} - a_{21}a_{12}a_{33} \end{aligned}$$

Determining the Determinant

- Can you think of an easier way?
- Hint: what is the determinant of the following upper triangular matrix?

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \quad \det(A) = a_{11}a_{22}a_{33} + a_{21}a_{32}a_{13} + a_{31}a_{12}a_{23} \\ - a_{31}a_{22}a_{13} - a_{11}a_{32}a_{23} - a_{21}a_{12}a_{33} \\ = a_{11}a_{22}a_{33}$$

- Q: How can we obtain a triangular matrix?

Determining the Determinant

- Apply Gaussian Elimination to bring matrix into row Echelon form.
- Then determinant comes directly from the diagonal:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \longleftarrow \boxed{\det(A) = \prod_i^n a_{ii}}$$

for triangular matrix

- There are three things to consider:
 1. Row addition has no effect on determinant
 2. Each row switching operation changes sign of determinant
 3. Scaling a row scales the determinant

Trace

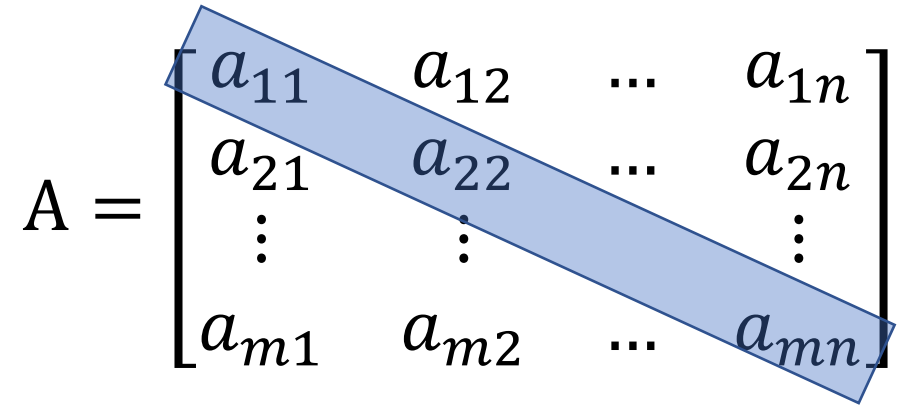
- The trace of a square matrix is the sum of the diagonal elements.
- It is defined as:

$$\mathbf{tr}(A) = \sum_{i=1}^n a_{ii}$$

- Has some interesting identities that can help simplify equations:

- $\mathbf{tr}(cA + dB) = c\mathbf{tr}(A) + d\mathbf{tr}(B)$
- $\mathbf{tr}(A^T B) = \mathbf{tr}(AB^T) = \mathbf{tr}(B^T A)$
- $\mathbf{tr}(ABC) = \mathbf{tr}(ACB) = \mathbf{tr}(BCA)$

Assuming that the dimensions work



A diagram of a square matrix A with dimensions $m \times n$. The matrix is shown as a grid of elements a_{ij} . The diagonal elements $a_{11}, a_{22}, \dots, a_{mn}$ are highlighted by a blue shaded parallelogram that runs from the top-left to the bottom-right. The matrix is enclosed in large square brackets.

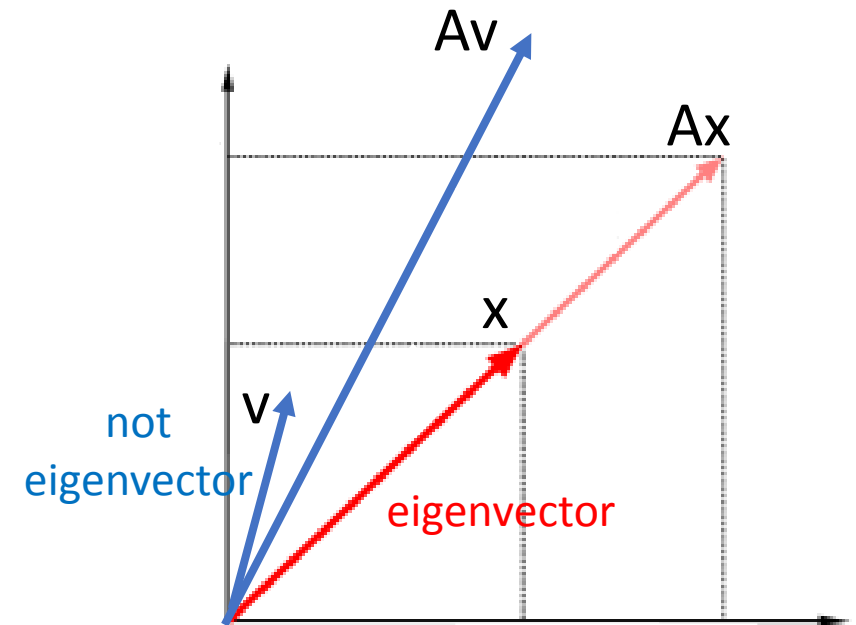
$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Eigenvectors

- Vectors that don't change their direction due to a transformation and can be represented as follows:

$$Ax = \lambda x$$

- Depending on the dimension there may be multiple independent vectors that satisfy the equation (eigen**vectors**).
- Where λ is a scalar referred to as eigen**value**.

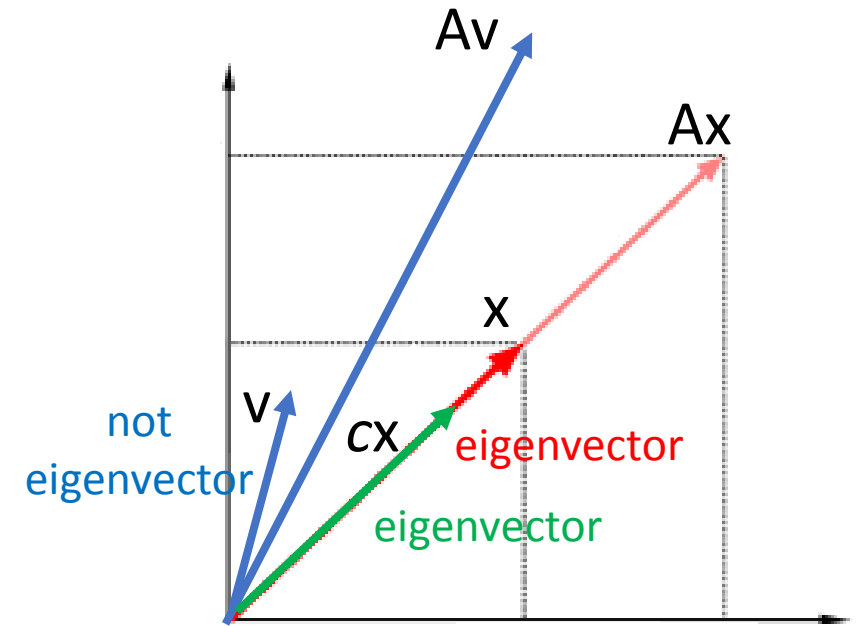


Non-Uniqueness of Eigenvectors

- If x is an eigenvector of A with corresponding eigenvalue λ then it holds that cx is also an eigenvector of A :

$$A(cx) = cAx = c\lambda x = \lambda(cx)$$

- All vectors that are collinear to x are also eigenvectors of A with the same eigenvalue λ



Finding Eigenvectors

- We can find the characteristic polynomial to find eigenvalues:

$$p_A(\lambda) = \mathbf{det}(A - \lambda I) \longleftarrow$$

We can describe
a matrix **A** as a
polynomial

$$= \det \begin{bmatrix} a_{11} - \lambda & a_{12} & a_{13} \\ a_{21} & a_{22} - \lambda & a_{23} \\ a_{31} & a_{32} & a_{33} - \lambda \end{bmatrix}$$

$$= c_0\lambda + c_1\lambda^2 + \dots + c_{n-1}\lambda^{n-1} + -1^n\lambda^n$$

solve for roots of the polynomial to obtain
eigenvalues and eigenvectors!

Aside:

$$Ax = \lambda x$$

$$Ax - \lambda x = 0$$

$$(A - \lambda I)x = 0$$

Theorems

➤ There are a few theorems worth mentioning:

1. λ_i is an eigenvalue if and only if λ_i is a root of a characteristic polynomial.
2. eigenvectors of a square matrix ($n \times n$) with **n distinct eigenvalues** are linearly independent and form a basis of \mathbb{R}^n .
3. The determinant of a square matrix ($n \times n$) is the product of its eigenvalues.

$$\det(A) = \prod_i^n \lambda_i$$

4. The trace of an $n \times n$ matrix is the sum of its eigenvalues.

$$\text{tr}(A) = \sum_{i=1}^n \lambda_i$$

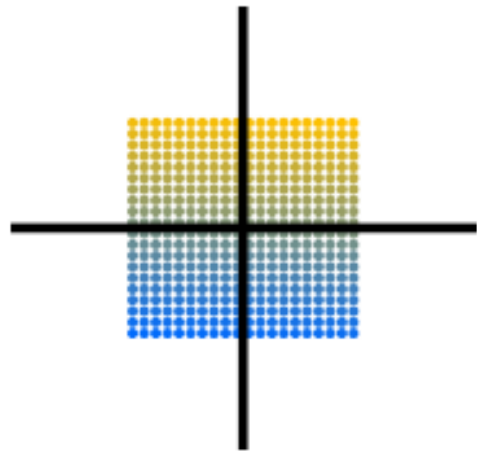
Example 4.5

$$A = \begin{bmatrix} 4 & 2 \\ 1 & 3 \end{bmatrix}$$

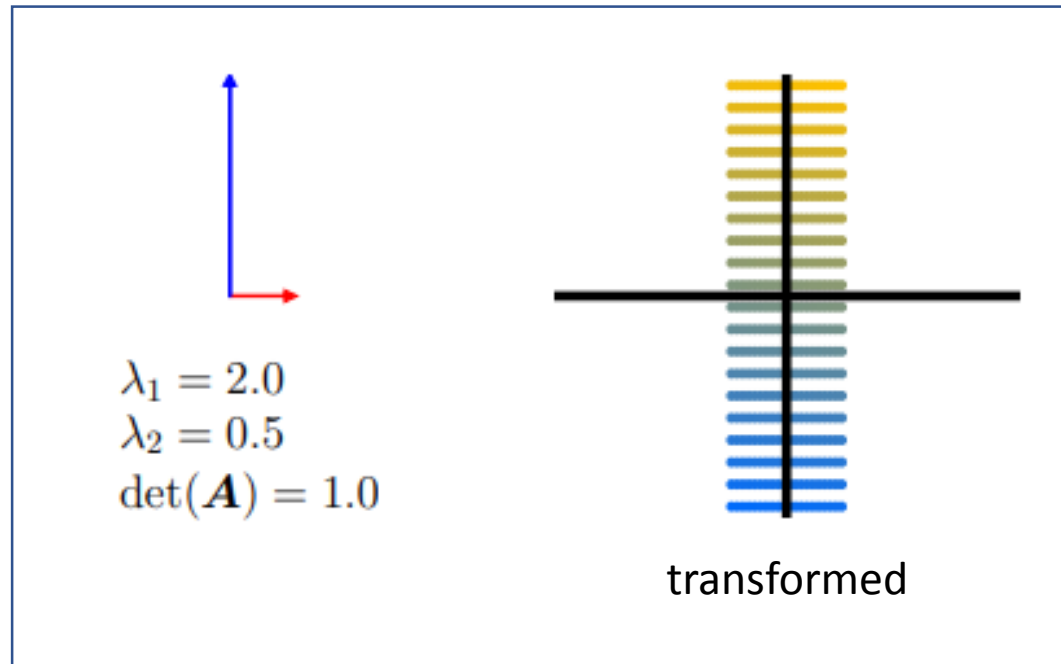
Eigenvalues?
Eigenvectors?

Graphical Intuition - Scaling

$$A = \begin{bmatrix} 1/2 & 0 \\ 0 & 2 \end{bmatrix}$$



original

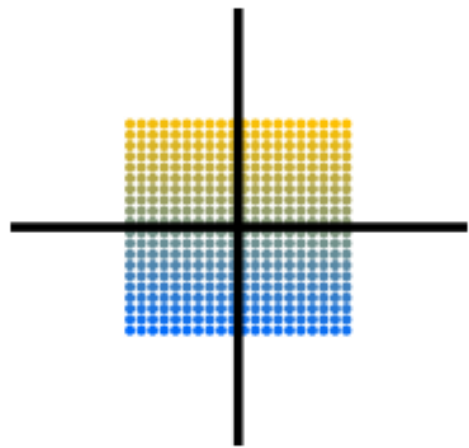


transformed

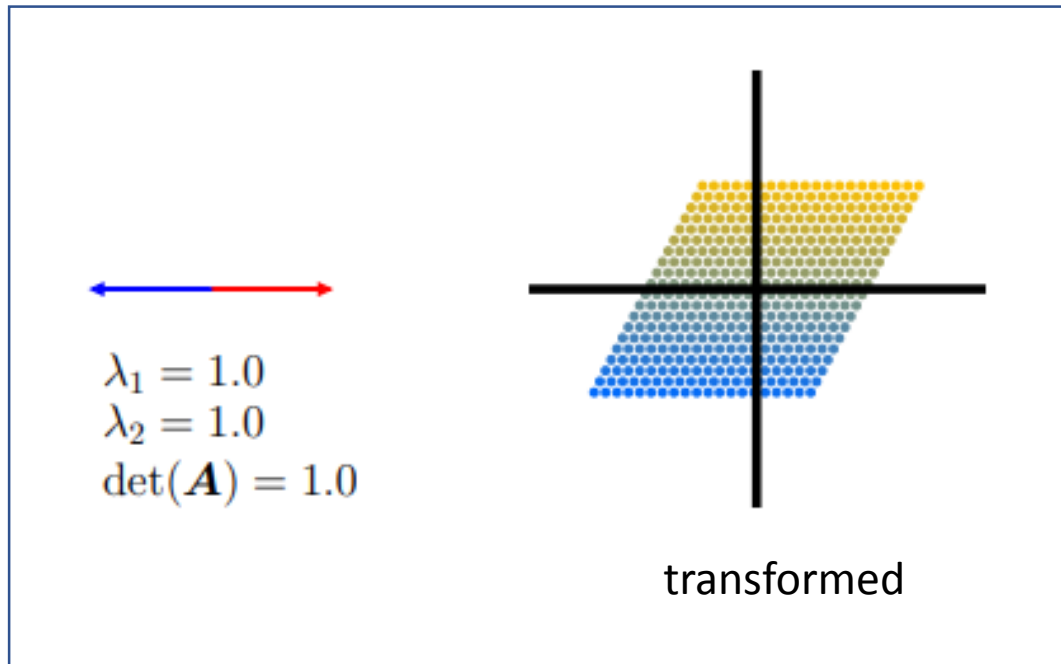
- The vertical axis stretched by a factor of 2 and the horizontal axis is compressed by a factor of 1/2. The mapping is area preserving.

Graphical Intuition - Shear

$$A = \begin{bmatrix} 1 & 1/2 \\ 0 & 1 \end{bmatrix}$$



original

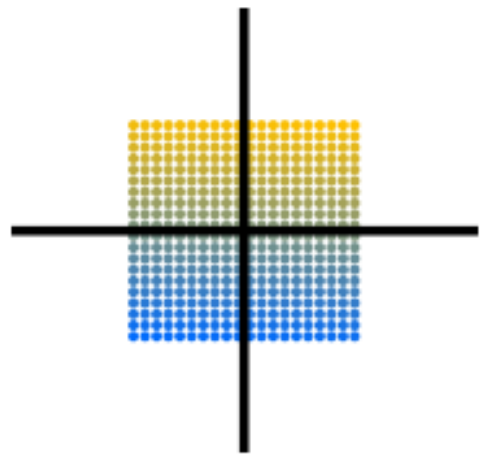


transformed

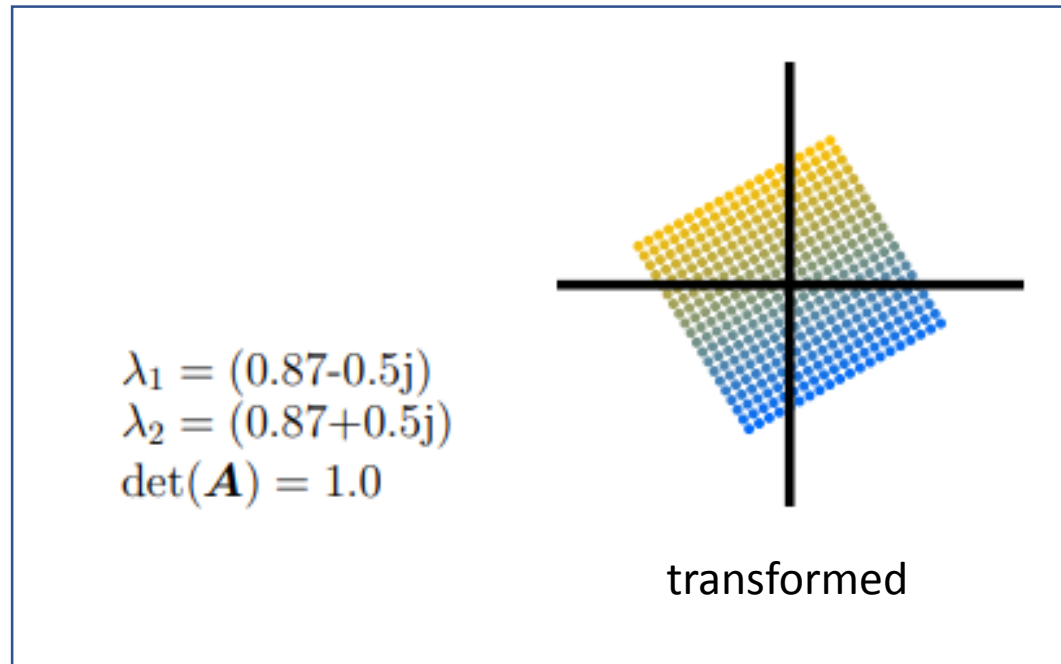
- Shears the points along the horizontal axis to the right if on positive side of the vertical axis and left if on negative side. Area preserving.

Graphical Intuition - Rotation

$$A = \begin{bmatrix} \cos(\pi/6) & -\sin(\pi/6) \\ \sin(\pi/6) & \cos(\pi/6) \end{bmatrix} = \begin{bmatrix} \sqrt{3} & -1 \\ 1 & \sqrt{3} \end{bmatrix}$$



original

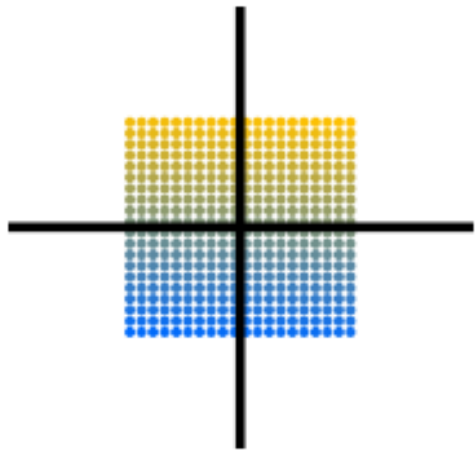


transformed

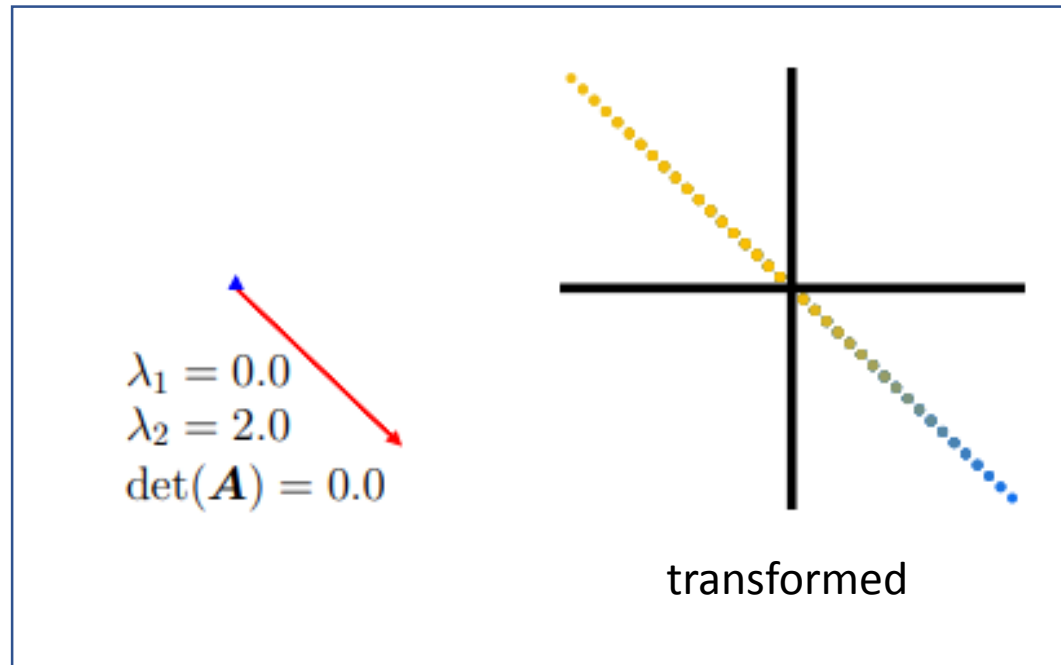
- The matrix rotates the points by $\pi/6$ rad (30 degrees) counterclockwise. The eigenvalues are complex, so cannot draw them on real axis. The rotation is area preserving.

Graphical Intuition – Collapse to Subspace

$$A = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$



original



transformed

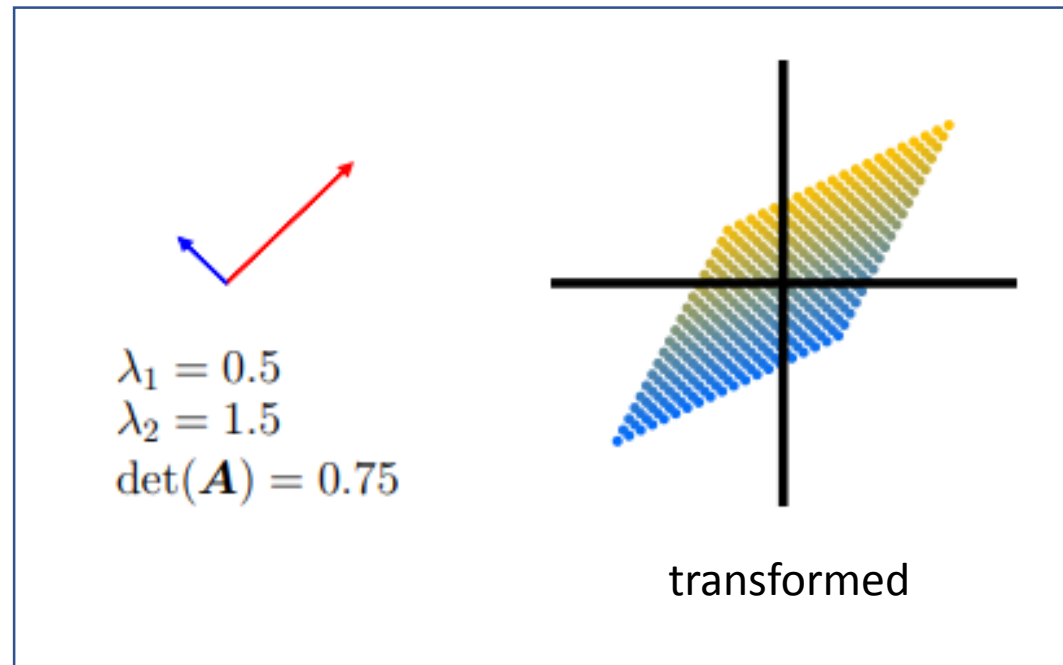
- Represents a mapping that collapses a two-dimensional domain onto one dimension. The eigenvalue $\lambda_1 = 0$ collapses while the other stretches the space by 2. The area in this case is 0.

Graphical Intuition – Shear and Stretch

$$A = \begin{bmatrix} 1 & 1/2 \\ 1/2 & 1 \end{bmatrix}$$



original



transformed

- Represents a shear and stretch mapping. It stretches along the red eigenvector by 1.5 and compresses along blue eigenvector by 0.5. That area is scaled by 75%.

Eigendecomposition

- A **square matrix** ($n \times n$) can be factored into:

$$A = PDP^{-1}$$

where P is $\mathbb{R}^{n \times n}$ and D is a diagonal matrix whose diagonal entries are eigenvalues of A (if and only if the eigenvectors form the basis of \mathbb{R}^n)

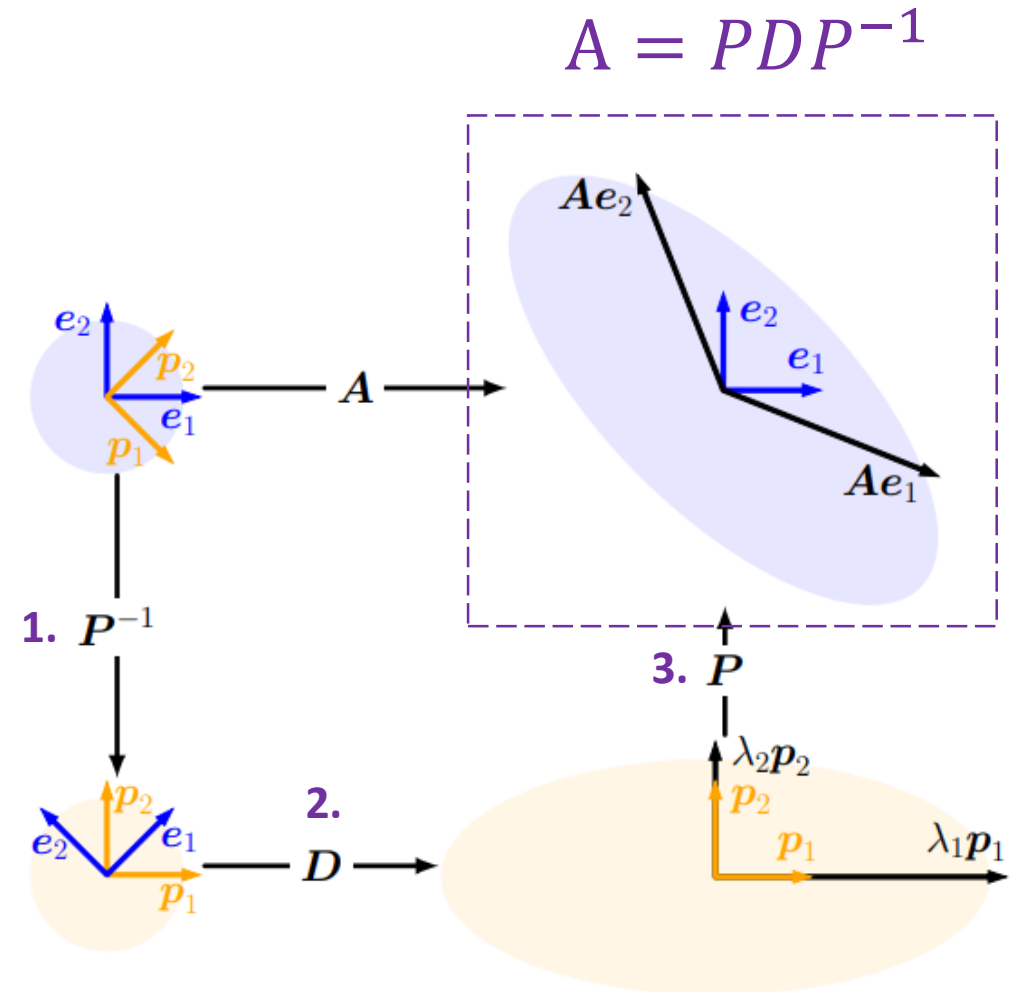
- For a **square symmetric matrix** the eigenvectors form an orthonormal basis (with real eigenvalues) and A can be factored into:

$$A = PDP^T \quad \text{Spectral Theorem}$$

Geometric Intuition

➤ **Eigendecomposition** can be thought of as a sequence of transformations:

1. P^{-1} performs a **basis change** depicted as a **rotation-like operation** from **standard basis** into the **eigenbasis**
2. D performs a **scaling** along the **remapped orthogonal vectors** depicted by a circle being stretched
3. P undoes the basis change depicted as a **reverse rotation** and restores the **original coordinate frame**.



Example 4.11 $A = \begin{bmatrix} 5/2 & -1 \\ -1 & 5/2 \end{bmatrix}$ Eigendecomposition of $A = PDP^{-1}$?

$$A = PDP^{-1} = 1/2 \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 7/2 & 0 \\ 0 & 3/2 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

Principal Component Analysis

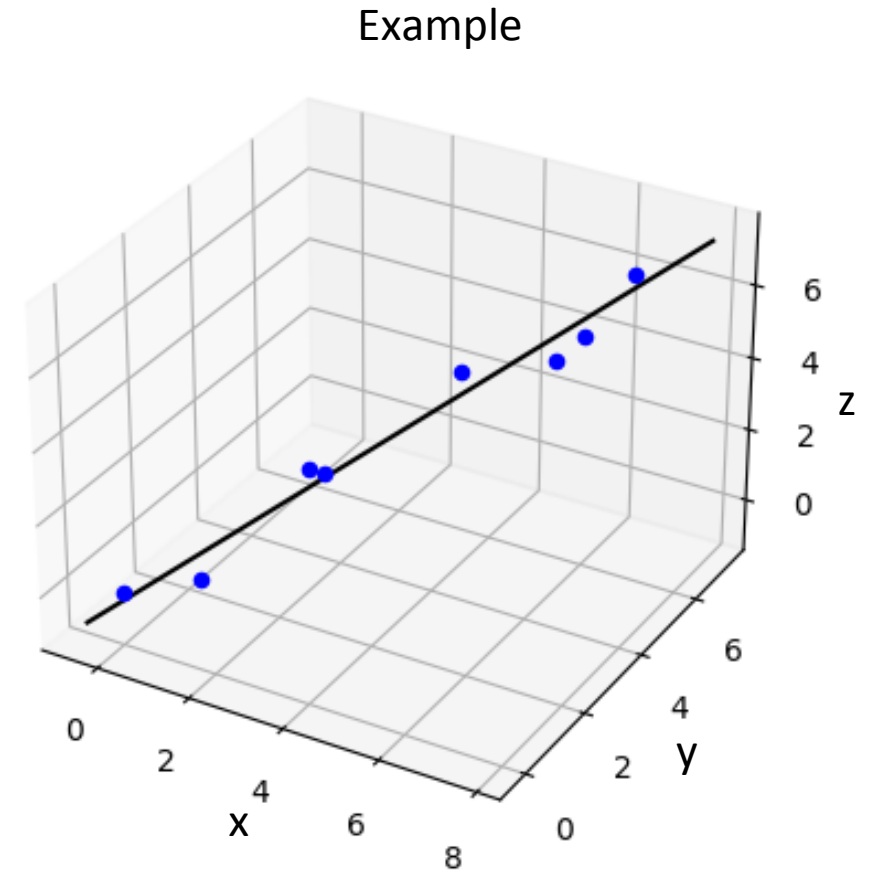
Readings:

- **Chapter 10 MML Textbook**

PCA

- Principle component analysis (PCA) is one example where eigendecomposition is used frequently
- Decomposing data in terms of factors has many benefits:
 - data interpretation
 - dimensionality reduction

You can think of it as projections that maximize the variance in the data (i.e. information content).



These slides are
adapted from

StatQuest

<https://www.youtube.com/watch?v=FgakZw6K1QQ>

Data

Examples!

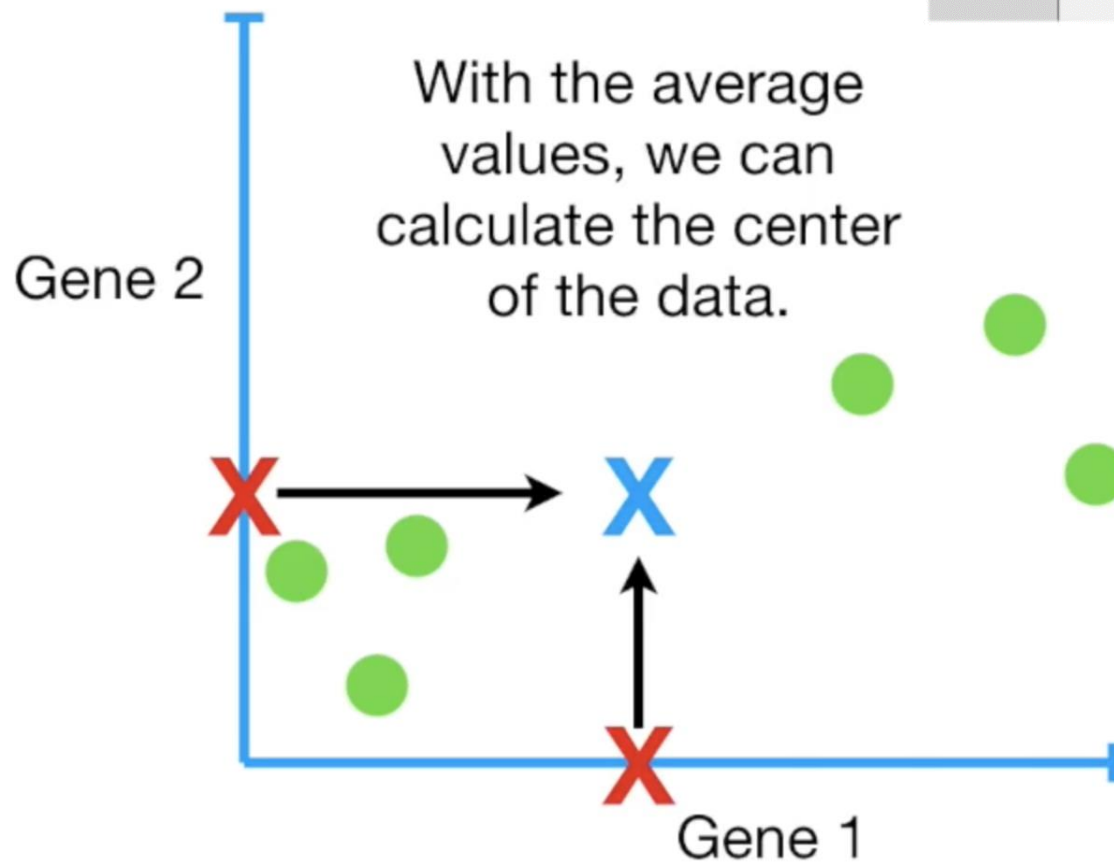
Variables!

	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

We can plot 2 variables in 2D.

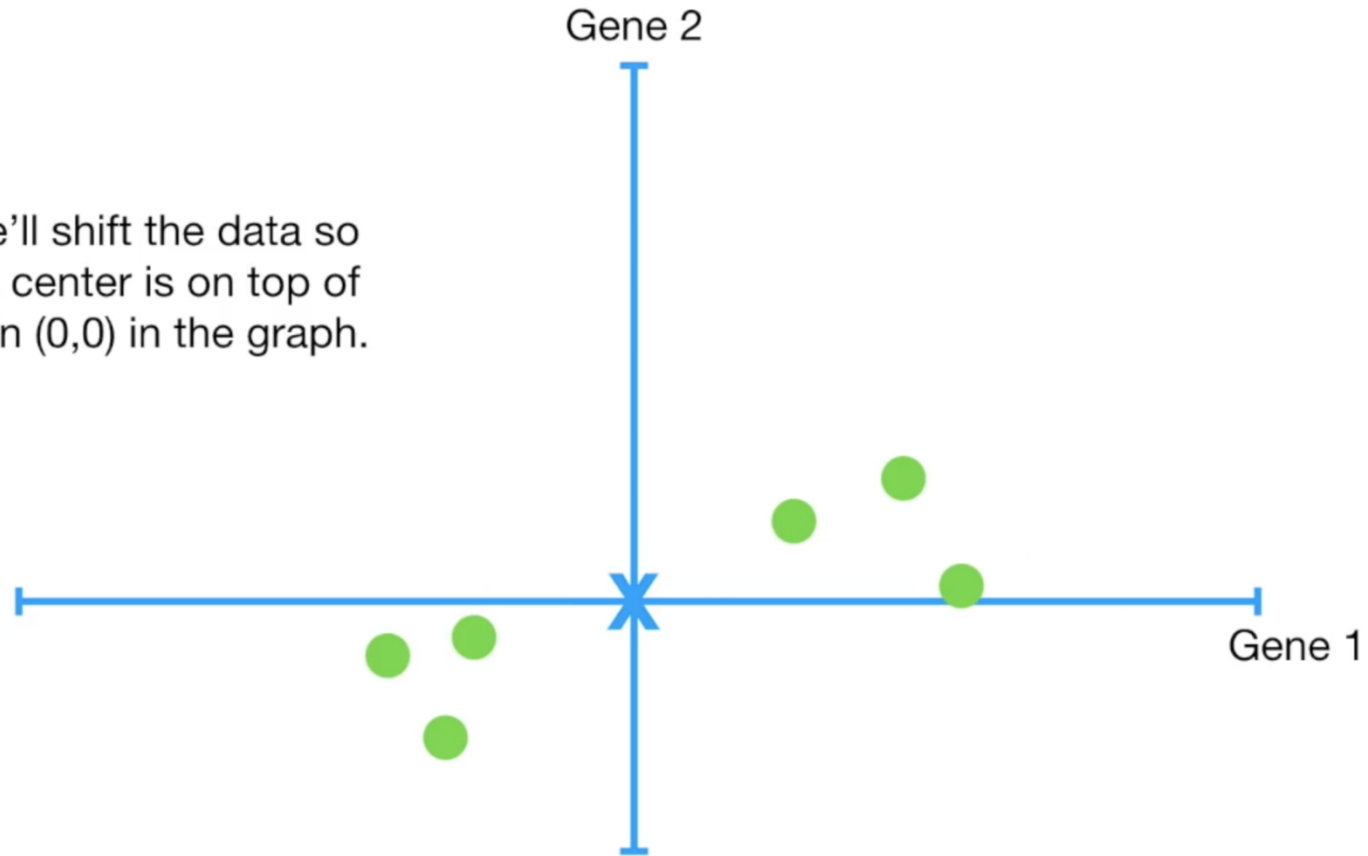
Average

	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

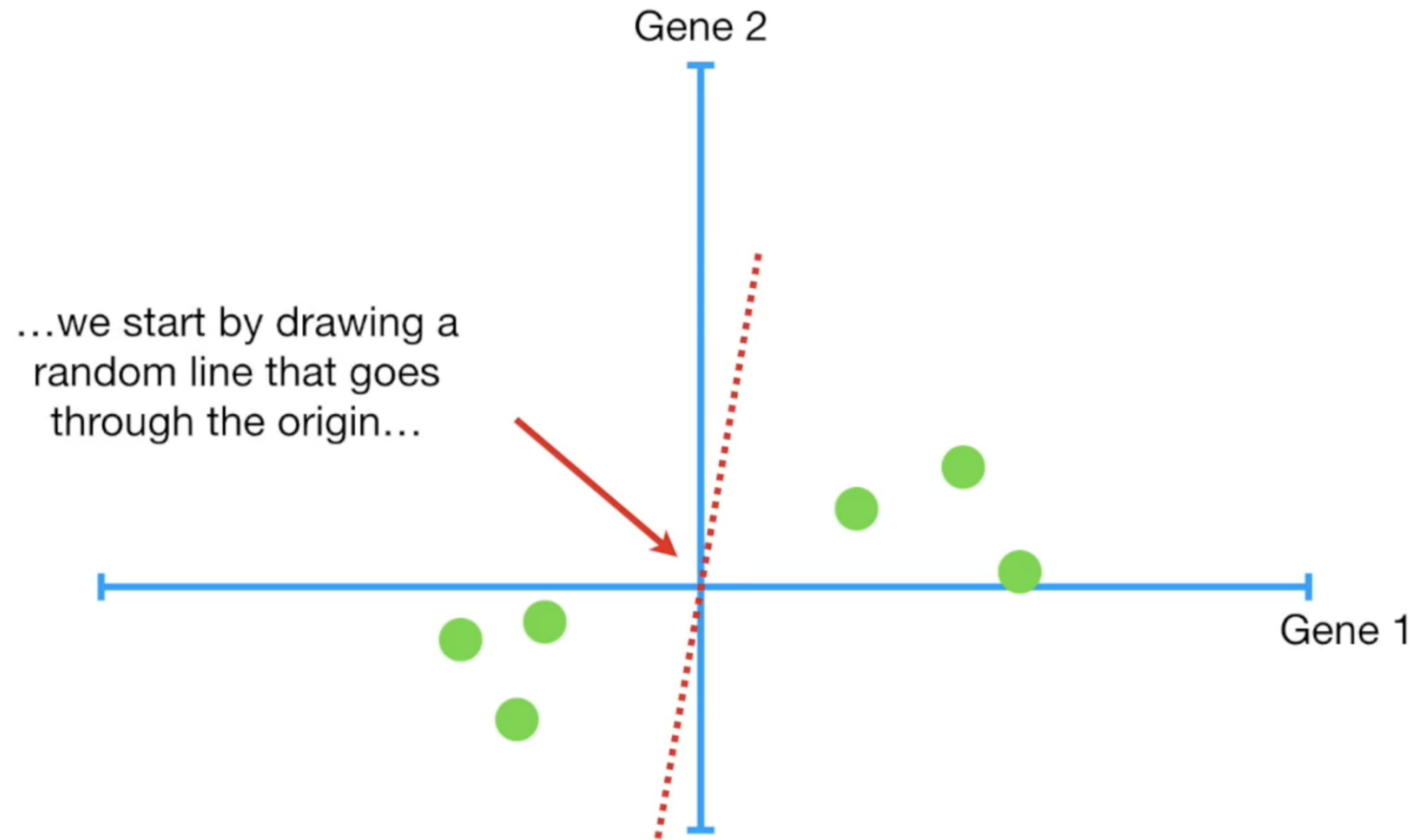


Shift the data

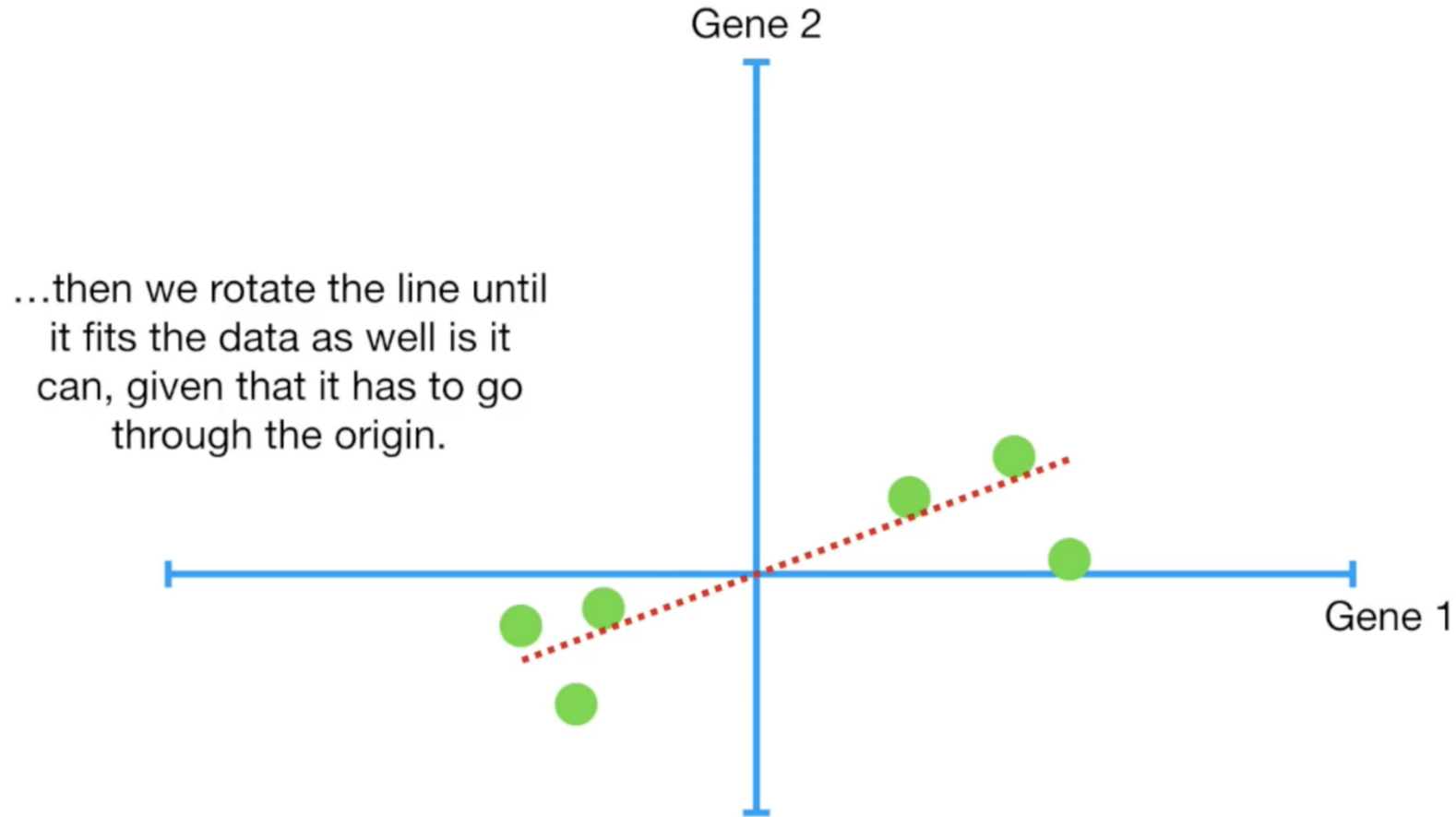
Now we'll shift the data so that the center is on top of the origin (0,0) in the graph.



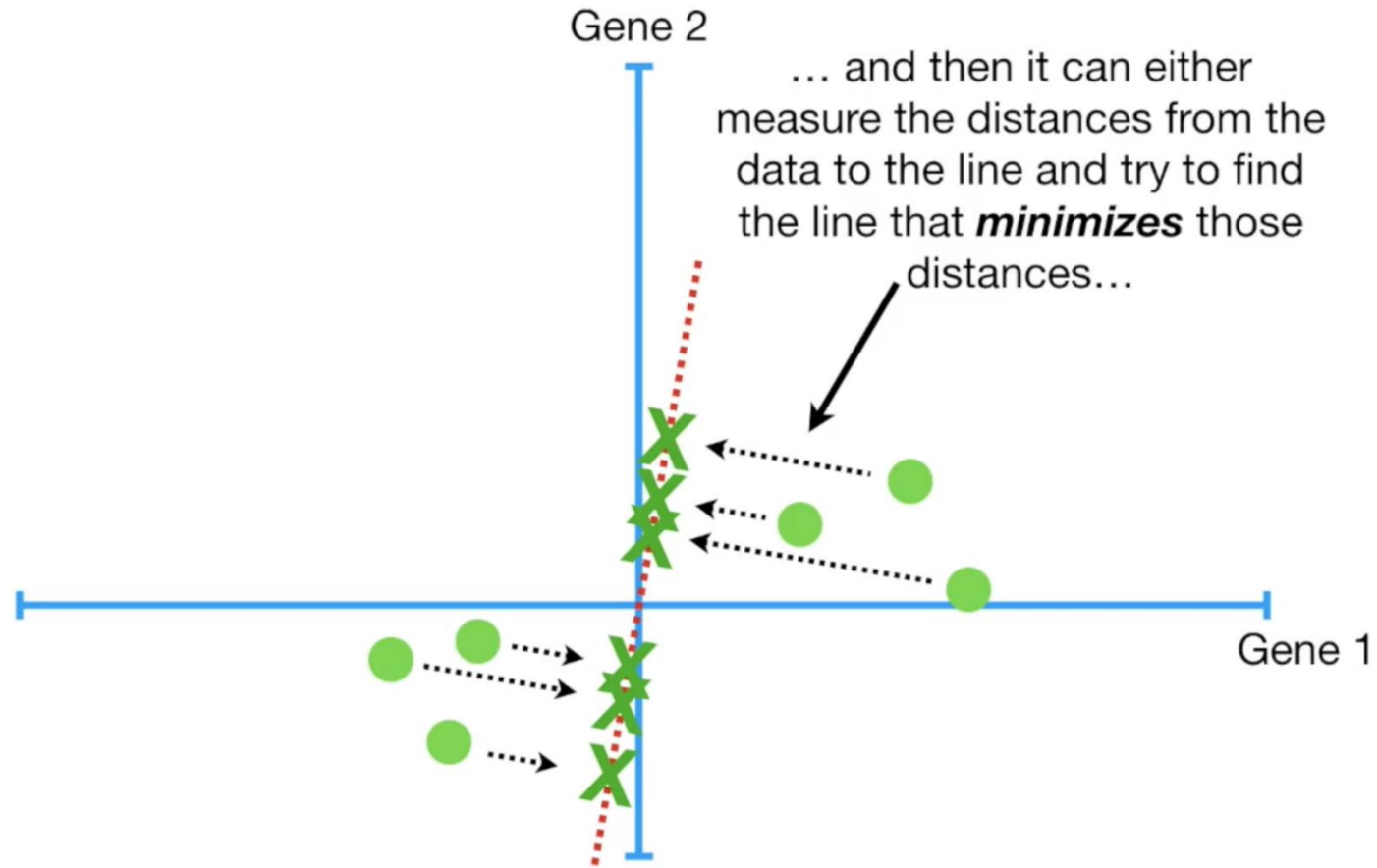
Draw a line crossing the origin



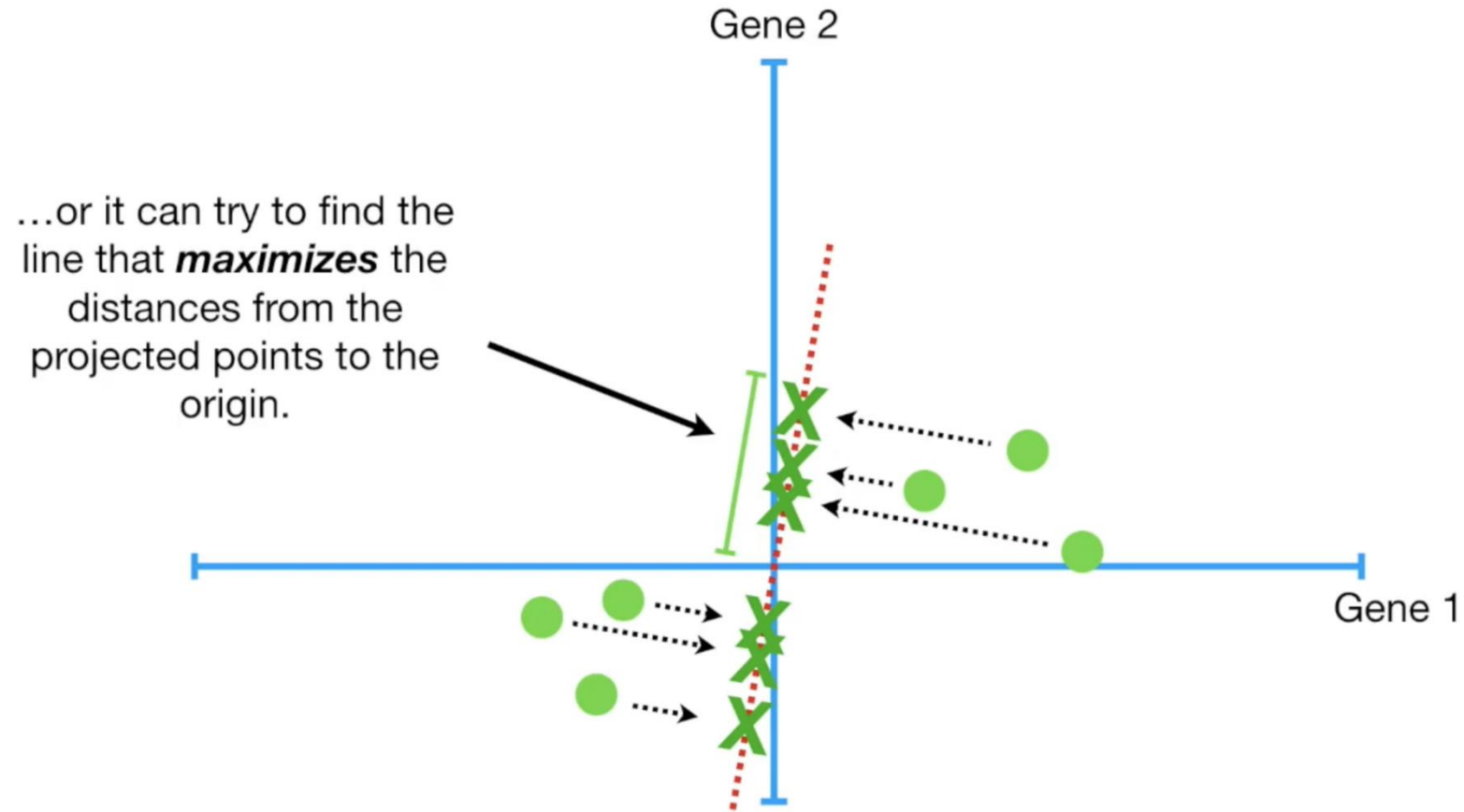
Goal: Fit the data as best as a line can



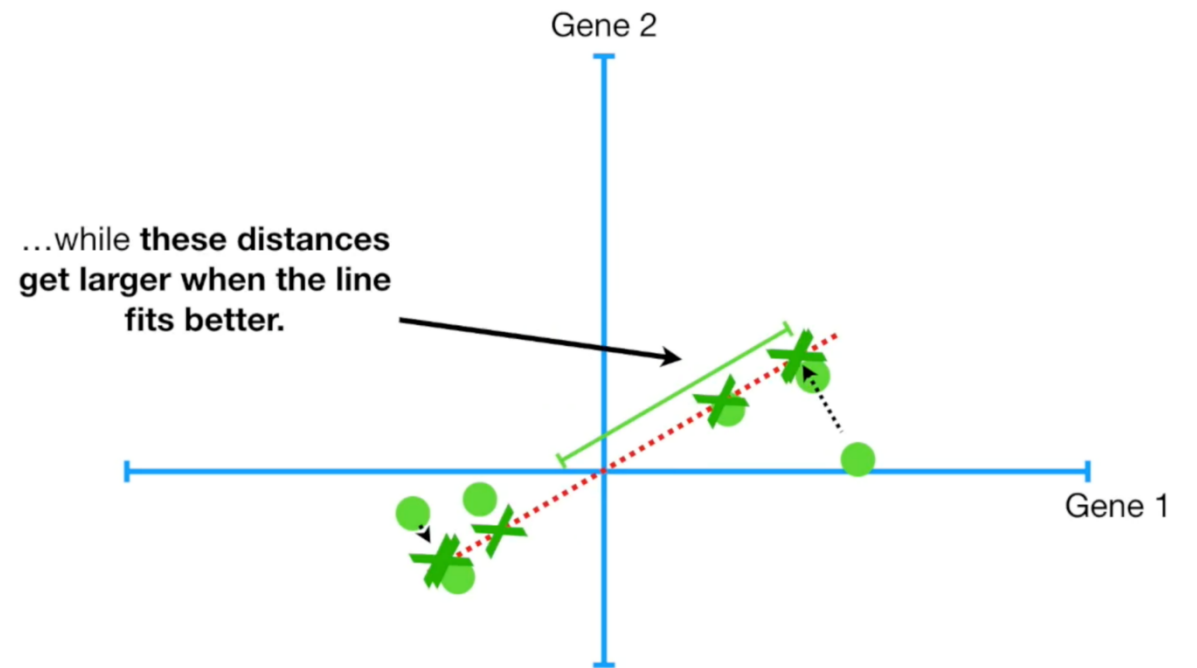
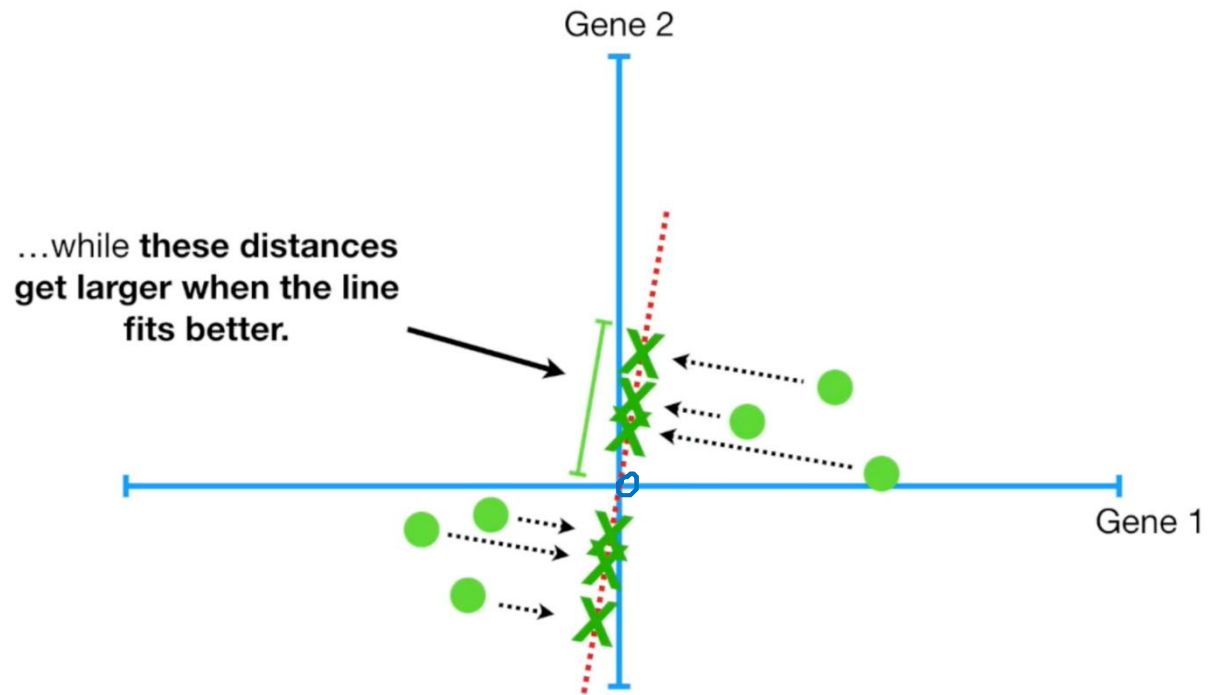
How do we know we have the best fit?



How we know the best fit?



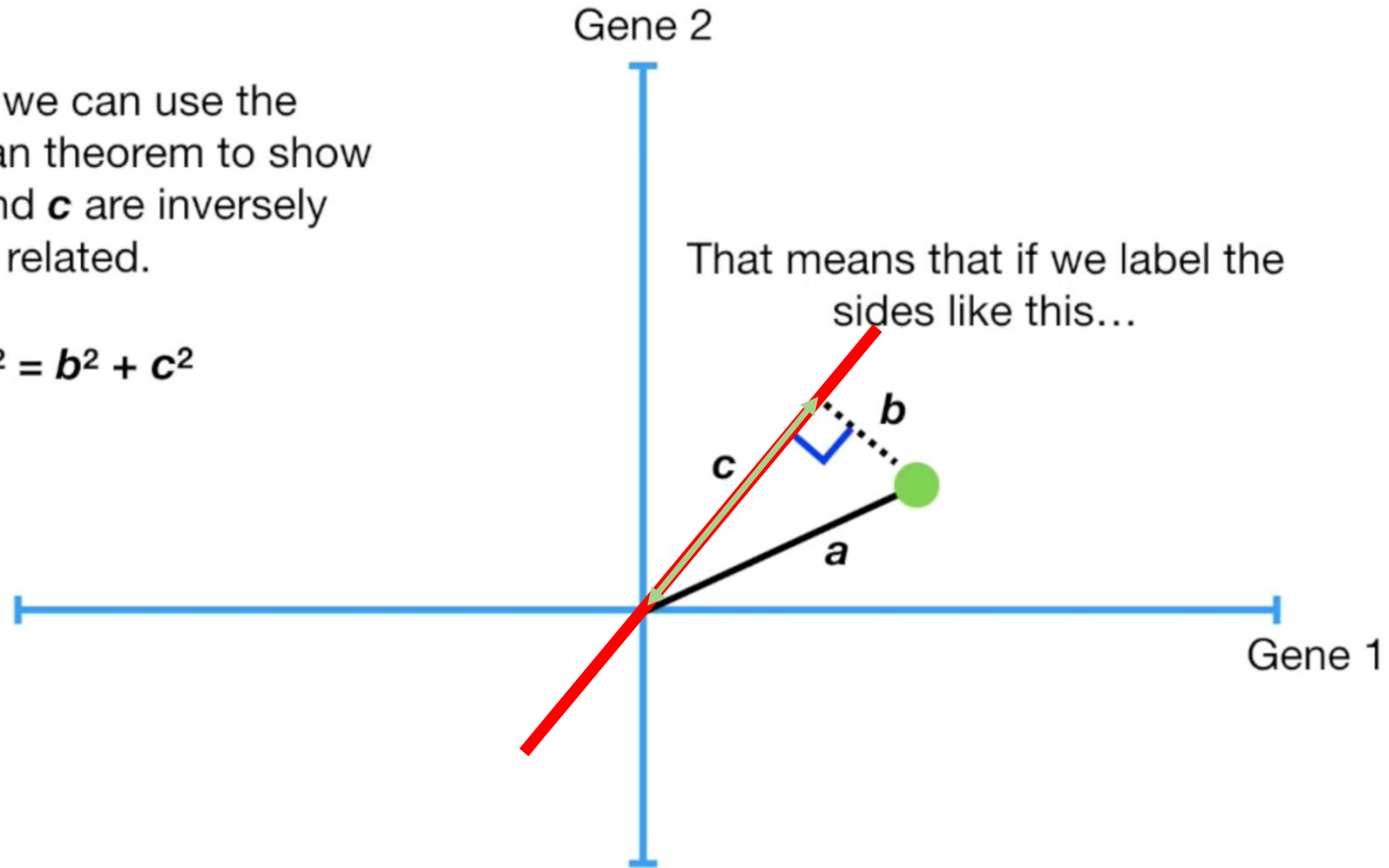
Intuition



Why?

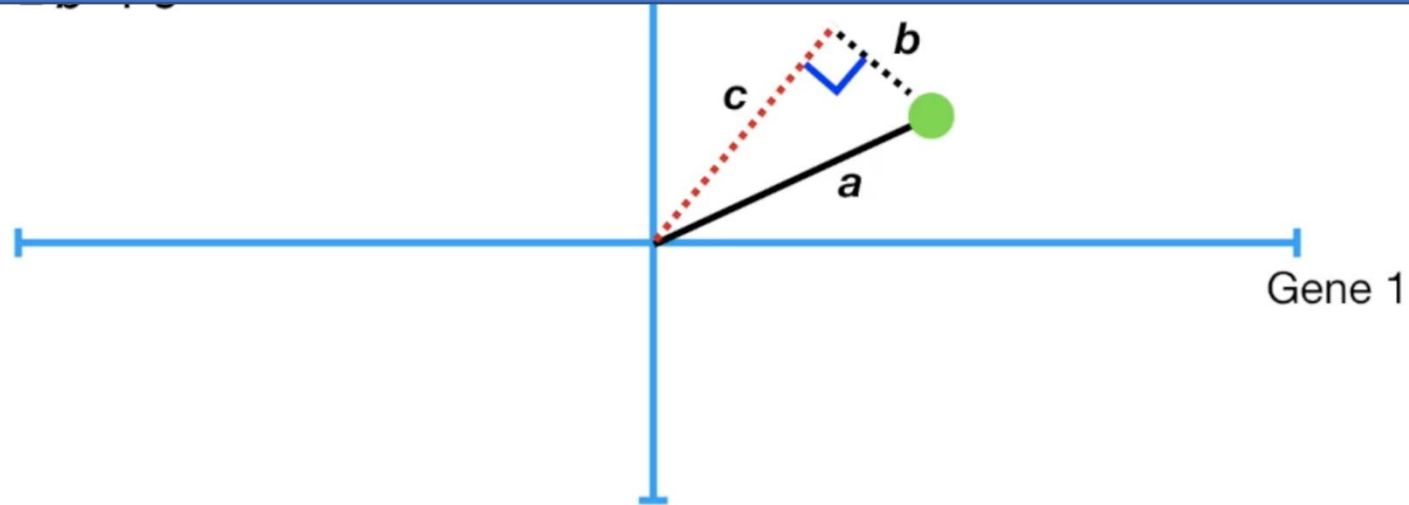
...then we can use the
Pythagorean theorem to show
how ***b*** and ***c*** are inversely
related.

$$a^2 = b^2 + c^2$$

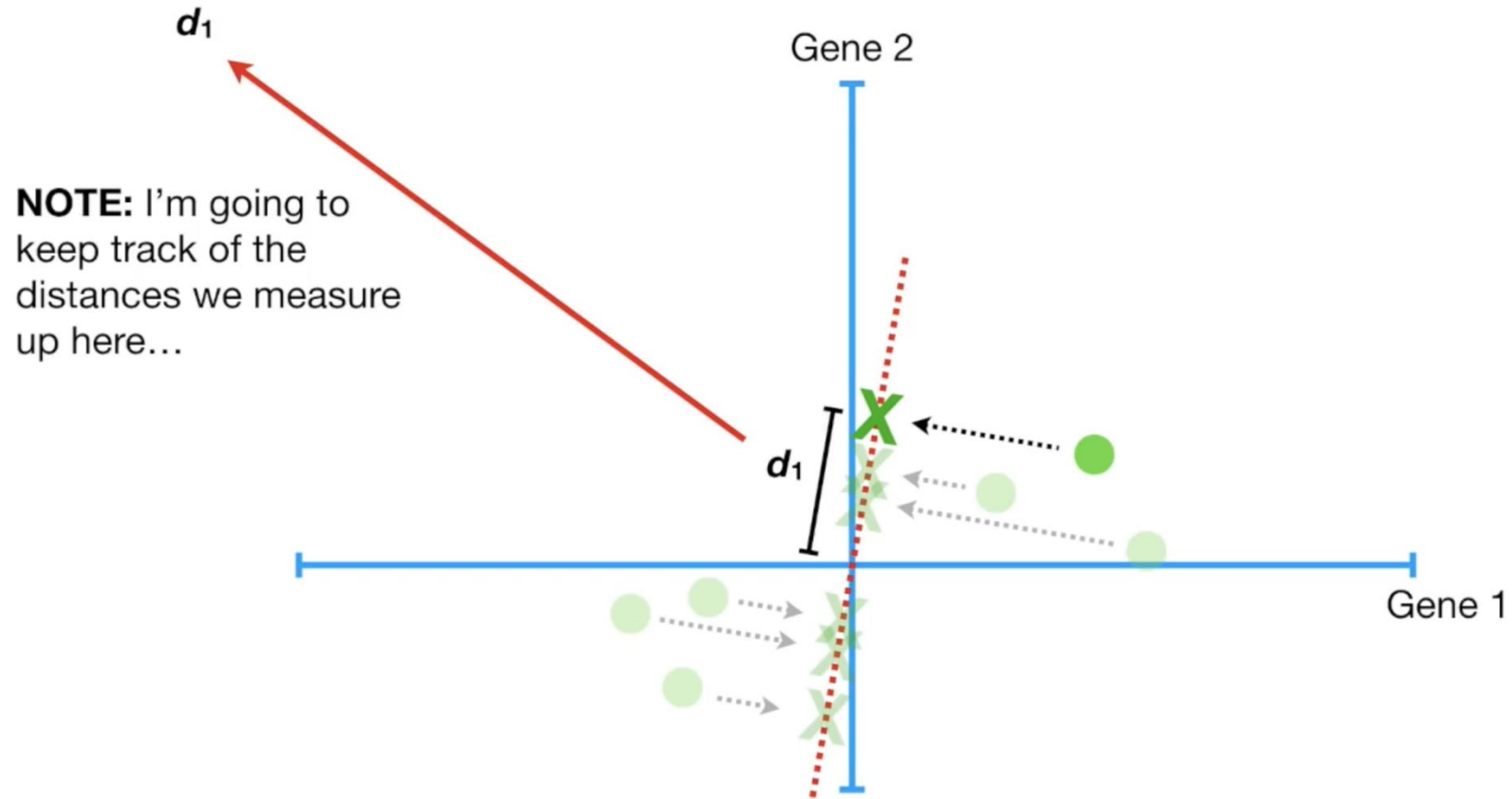


Why?

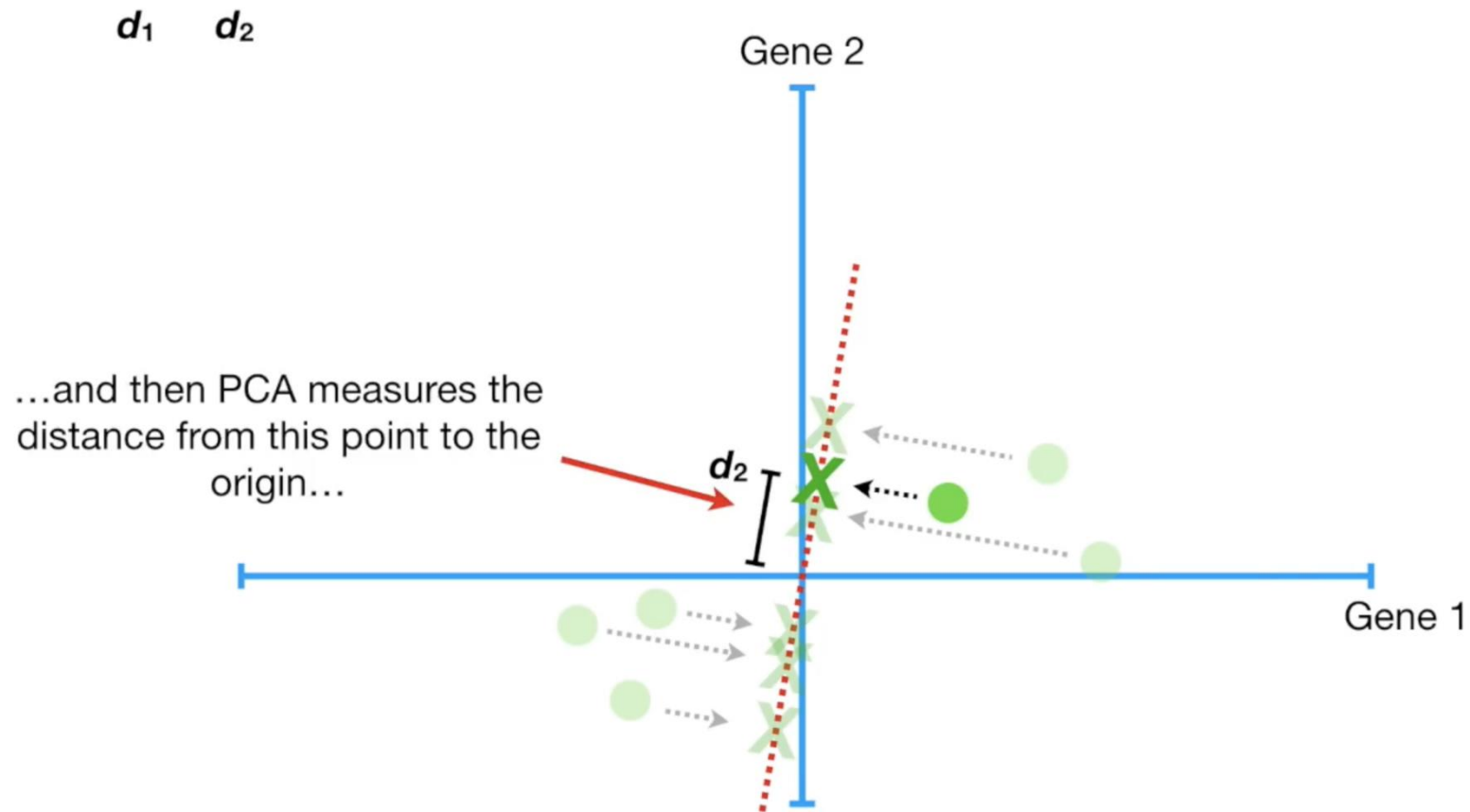
Objective: Minimize **b** or Maximize **c**



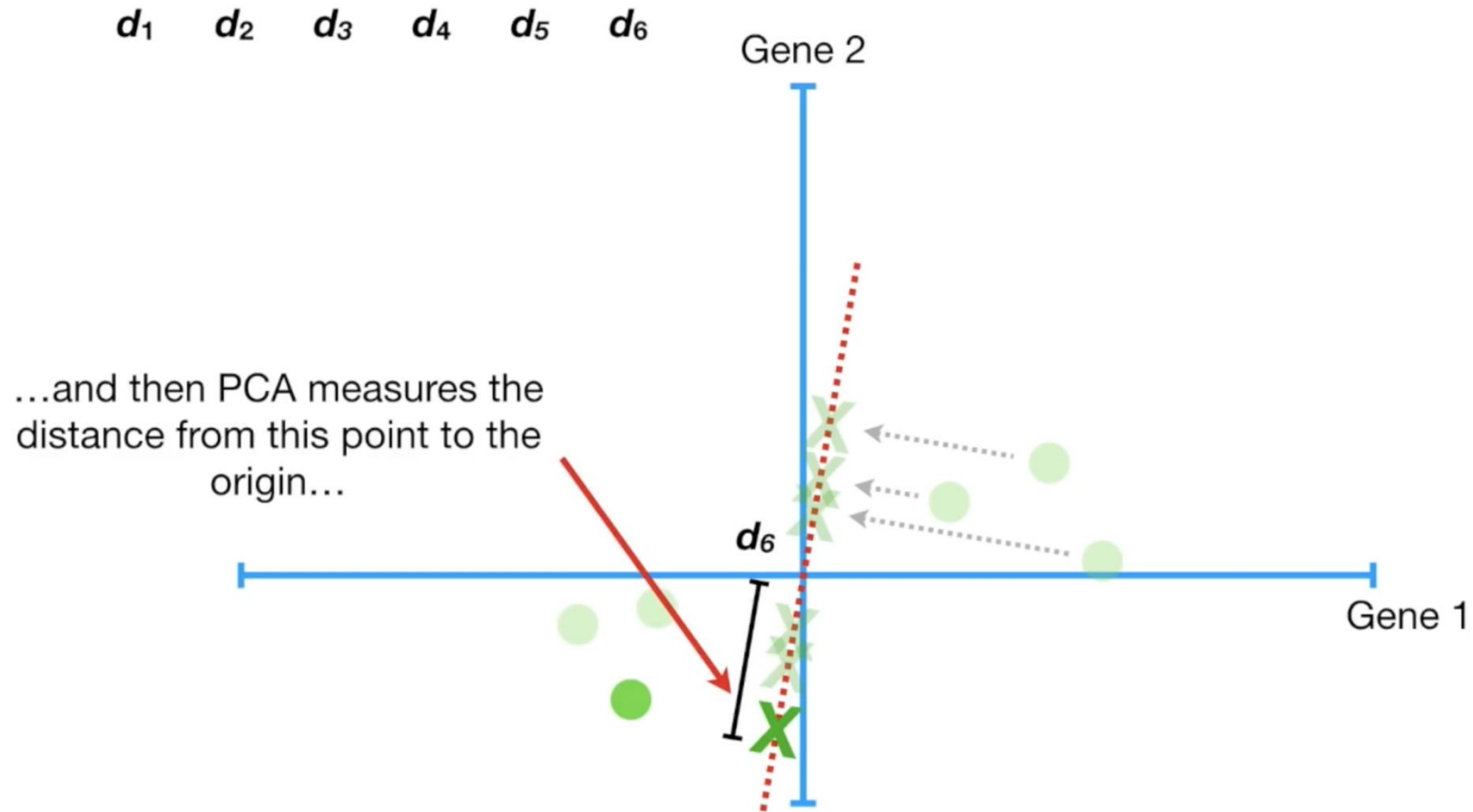
Distances



Distances



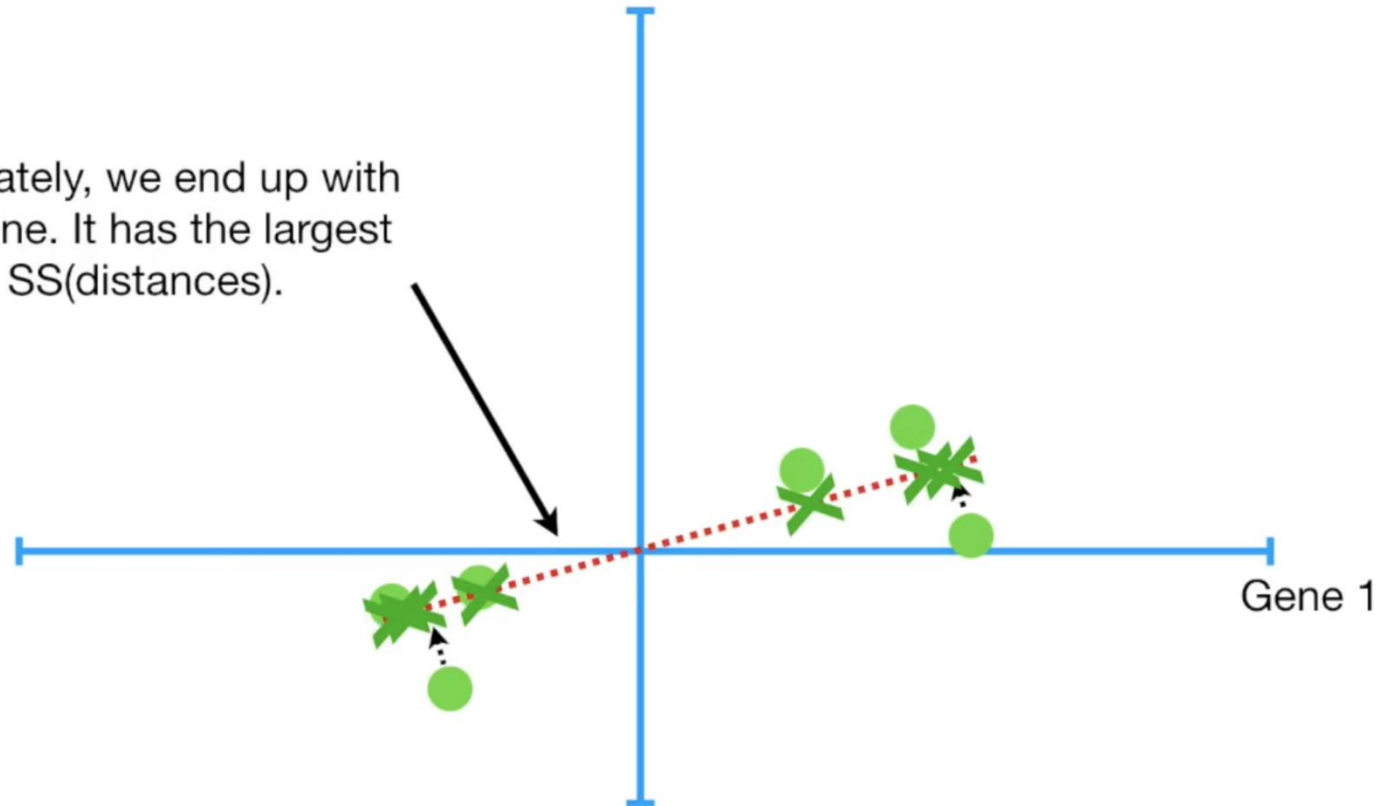
Distances



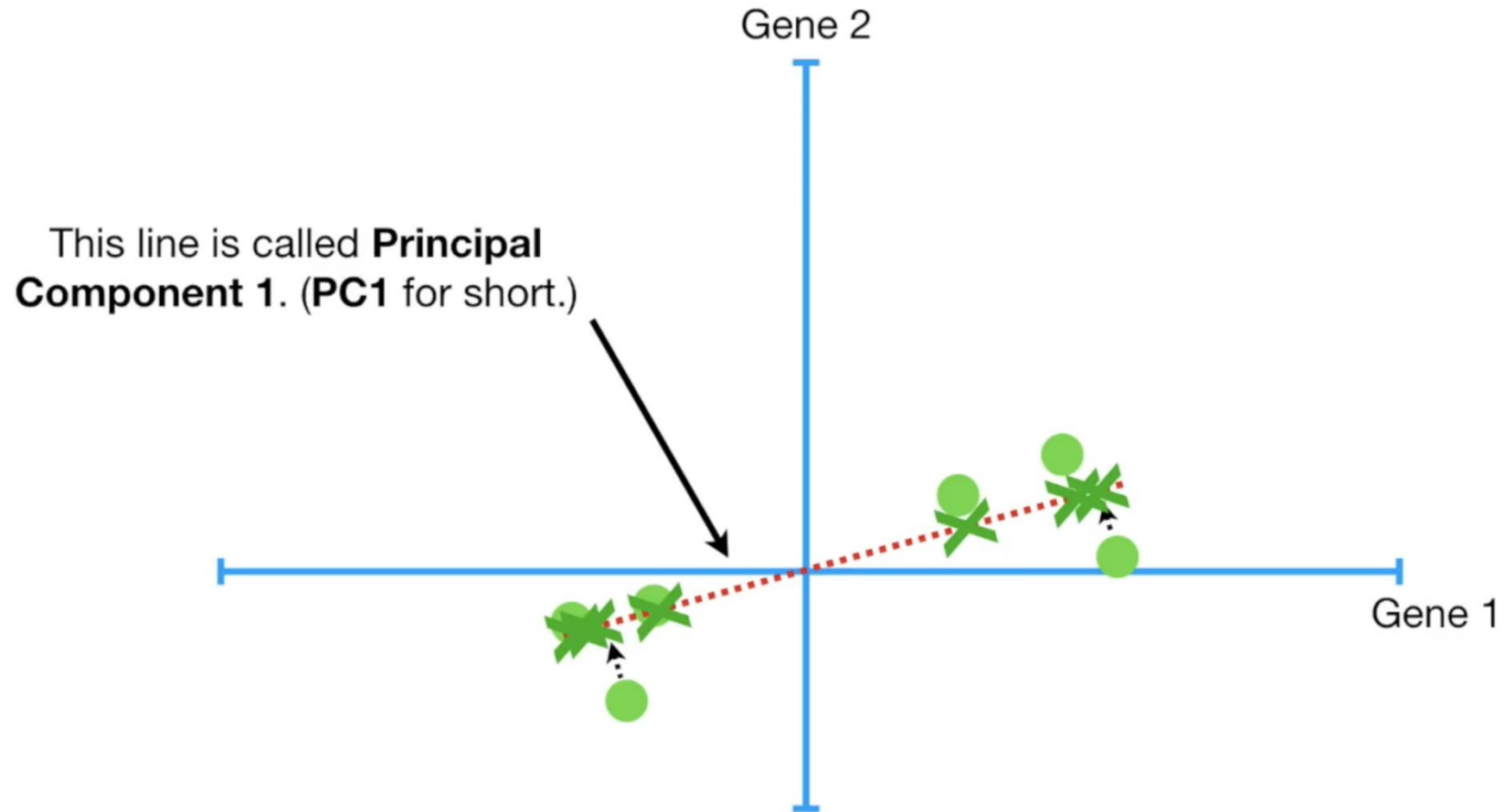
Maximize the Sum of squared distances

$$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2 = \text{sum of squared distances} = \text{SS}(\text{distances})$$

Ultimately, we end up with this line. It has the largest SS(distances).



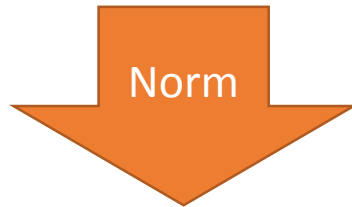
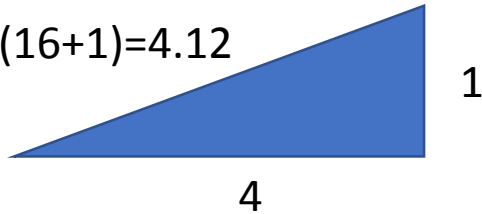
First Principal Component



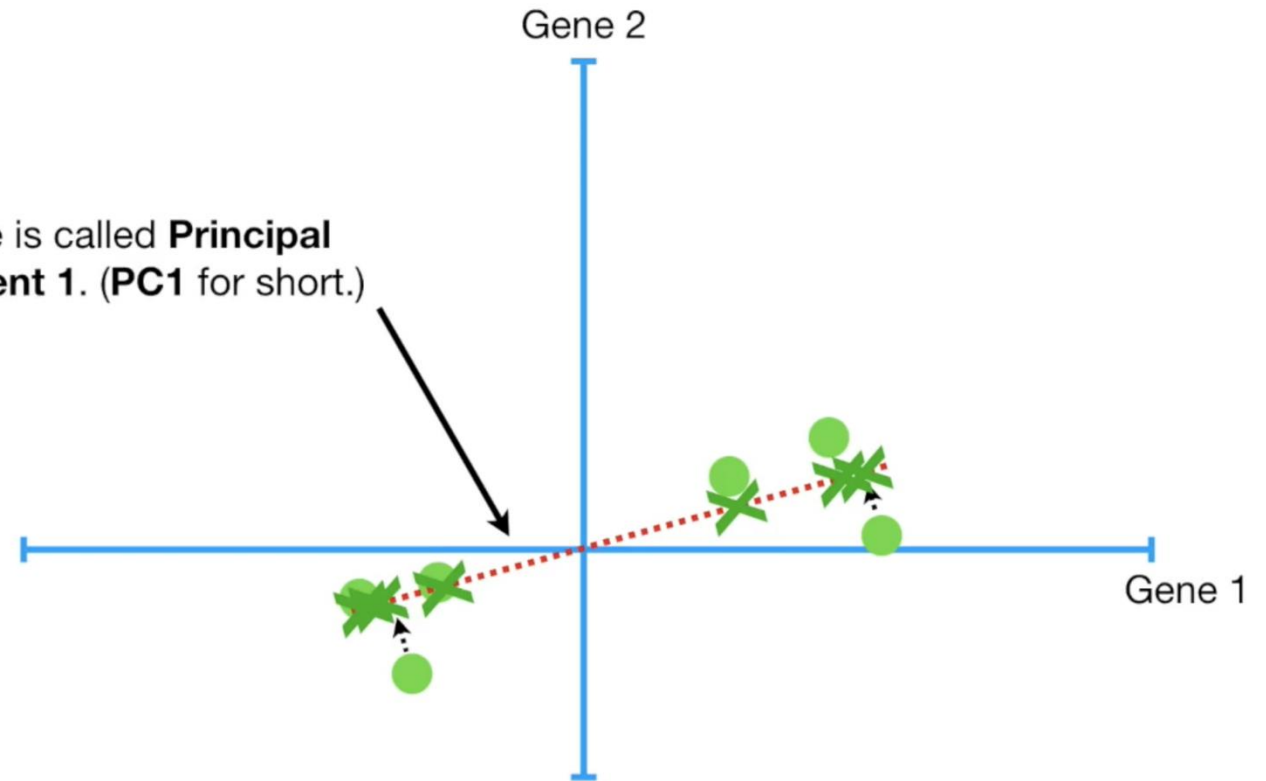
First Principal Component

The slope of the PC1 = 0.25

Sqrt (16+1)=4.12



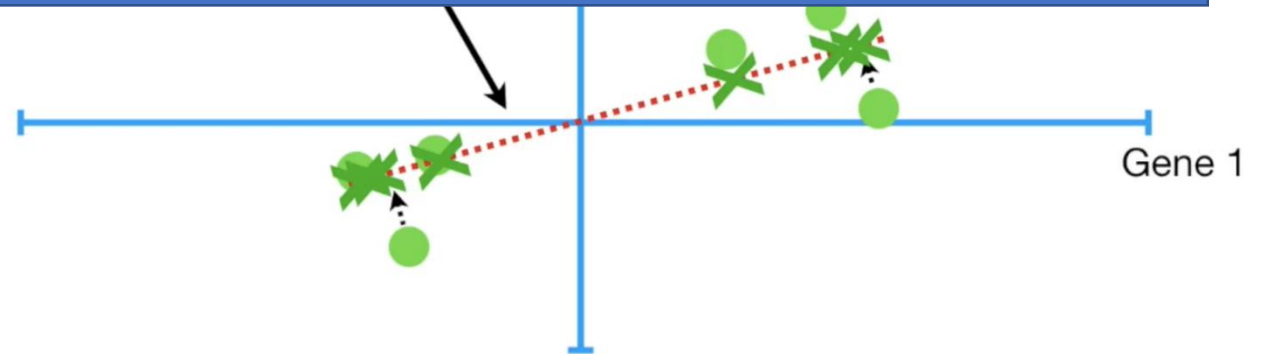
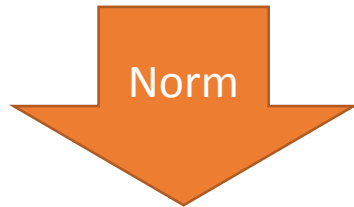
This line is called **Principal Component 1. (PC1 for short.)**



First Principal Component

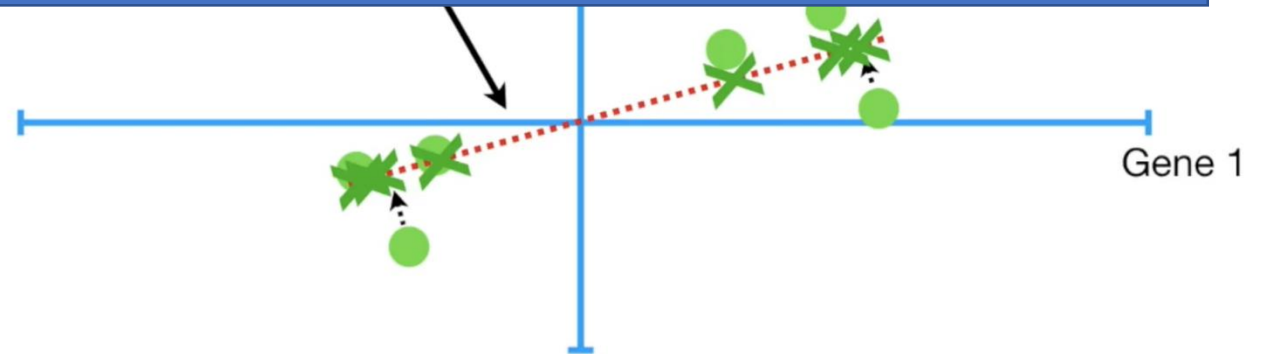
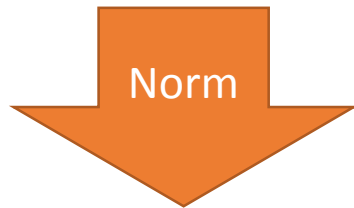
Eigenvector for PC1 = $[0.97, 0.242]$

Other name: Singular vector

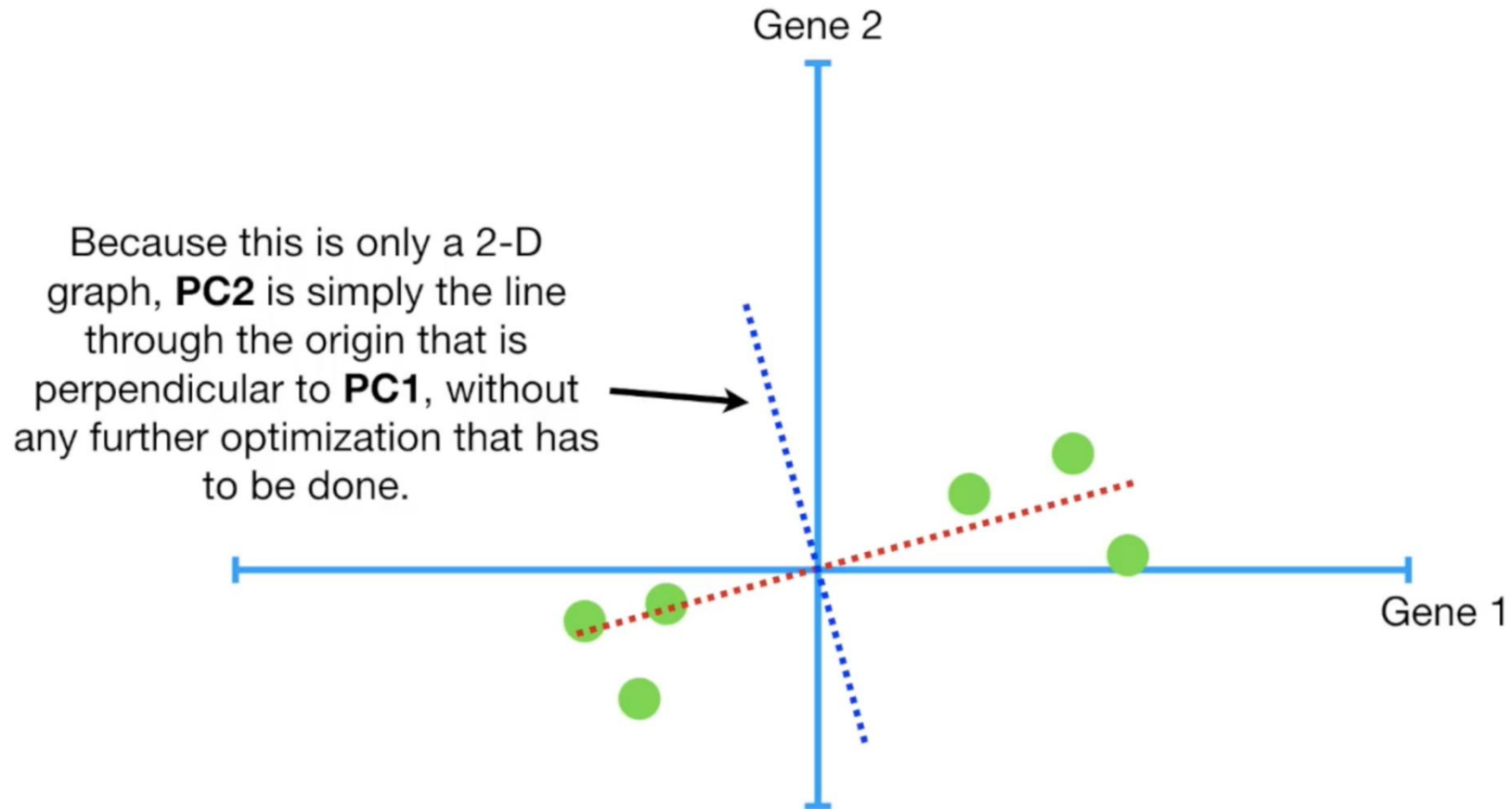


First Principal Component

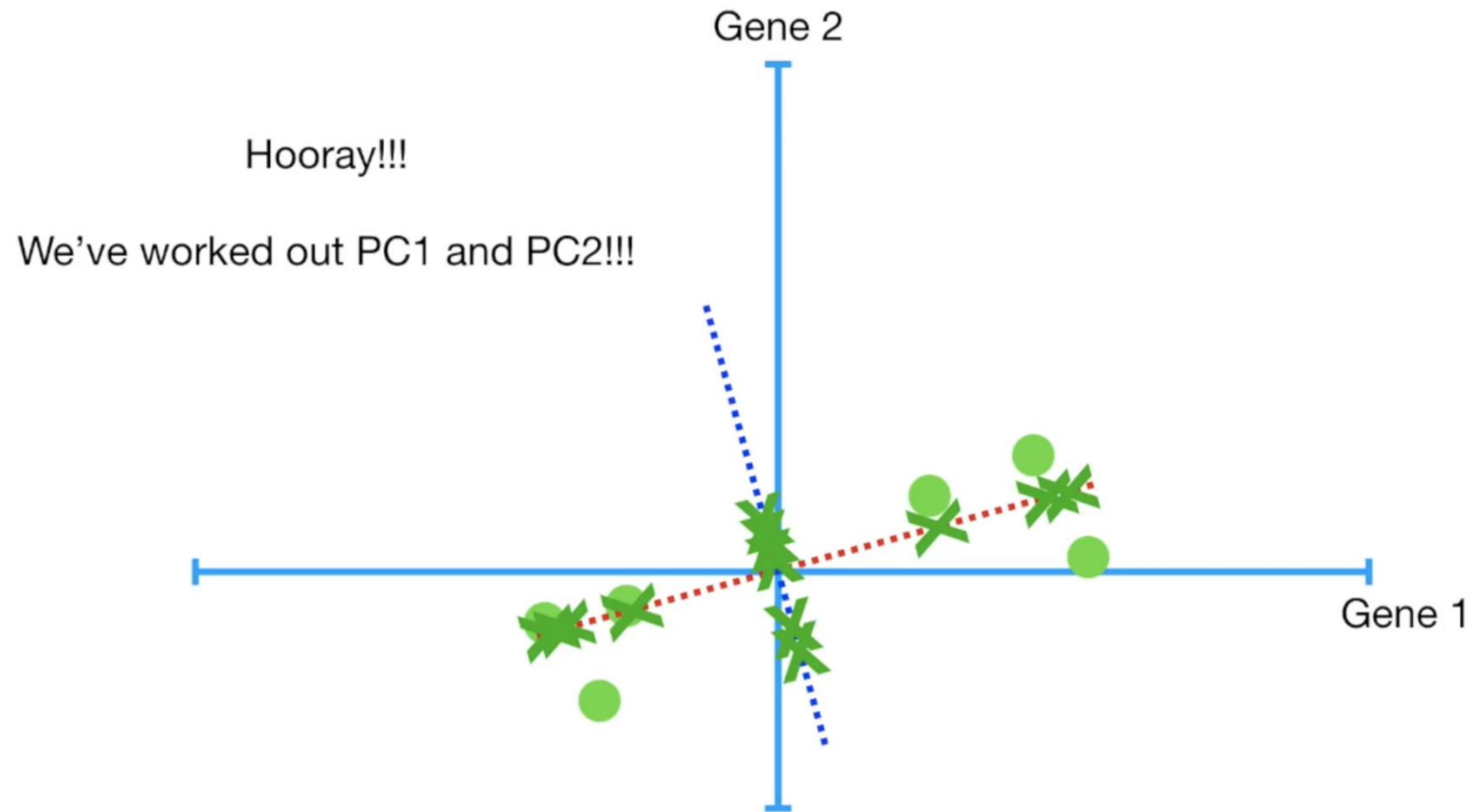
Sum of squared distances for PC1 = **Eigenvalue** for PC1
 $\text{Sqrt}(\text{Eigenvalue for PC1}) = \text{Singular Value for PC1}$



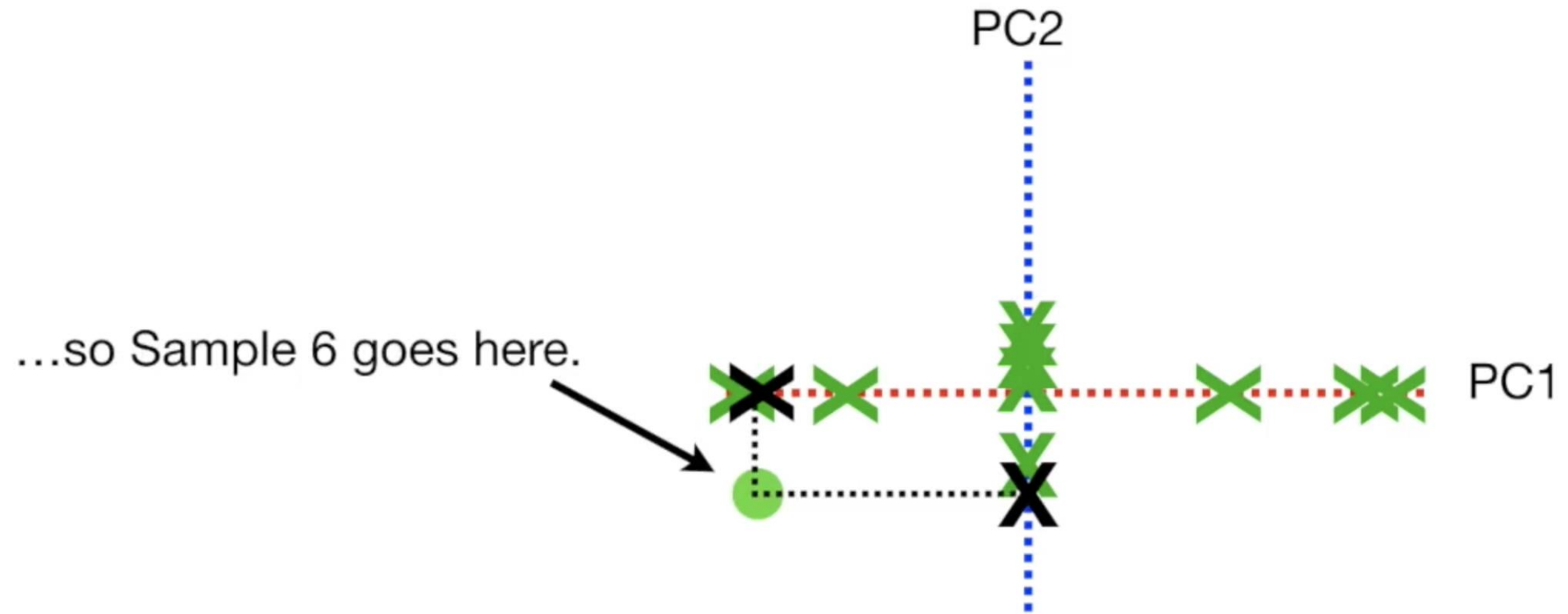
PC2



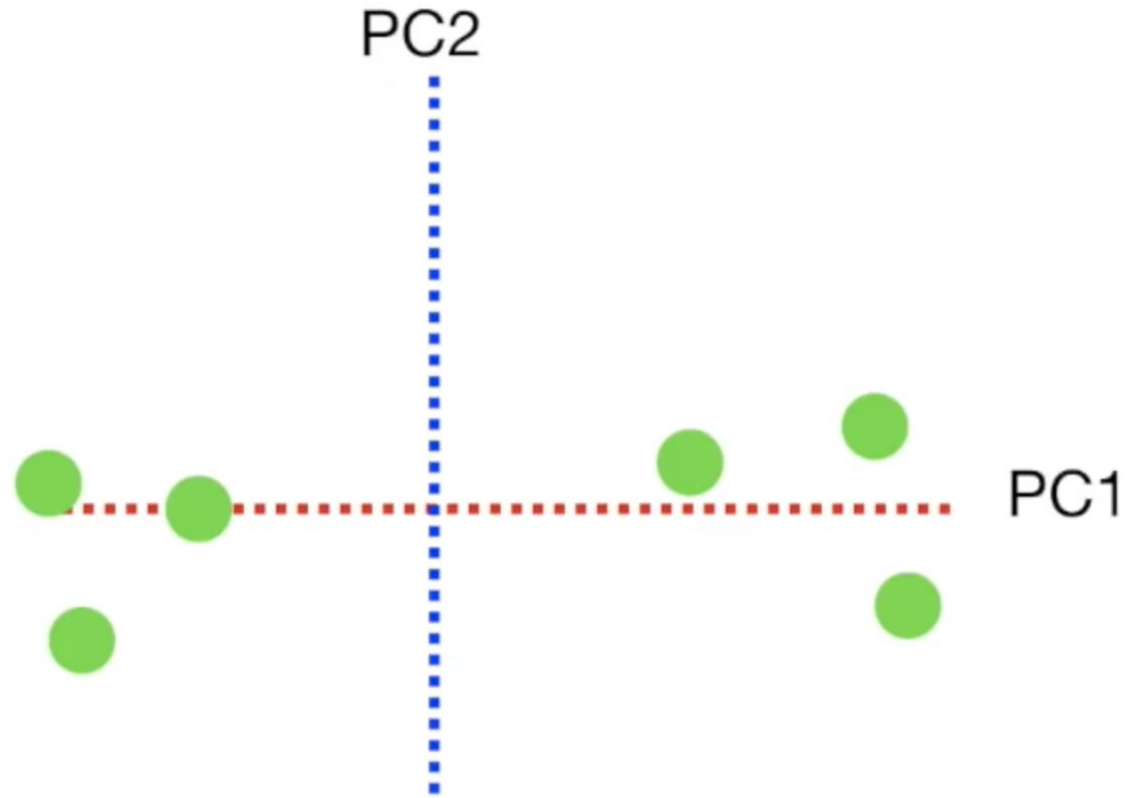
Projection - PC1 and PC2



Reconstruction

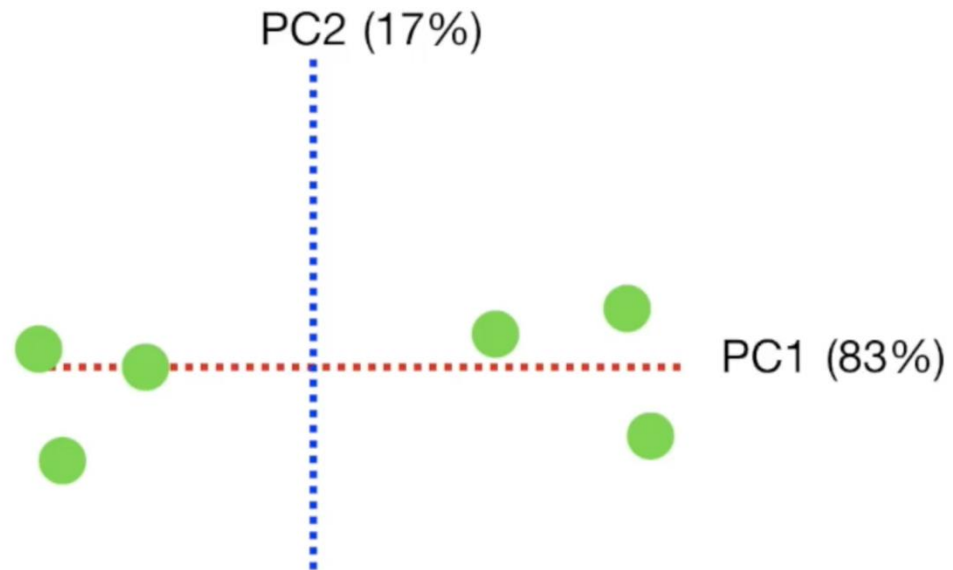
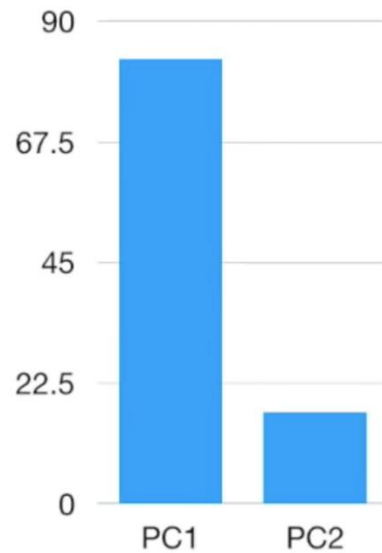


Reconstruction



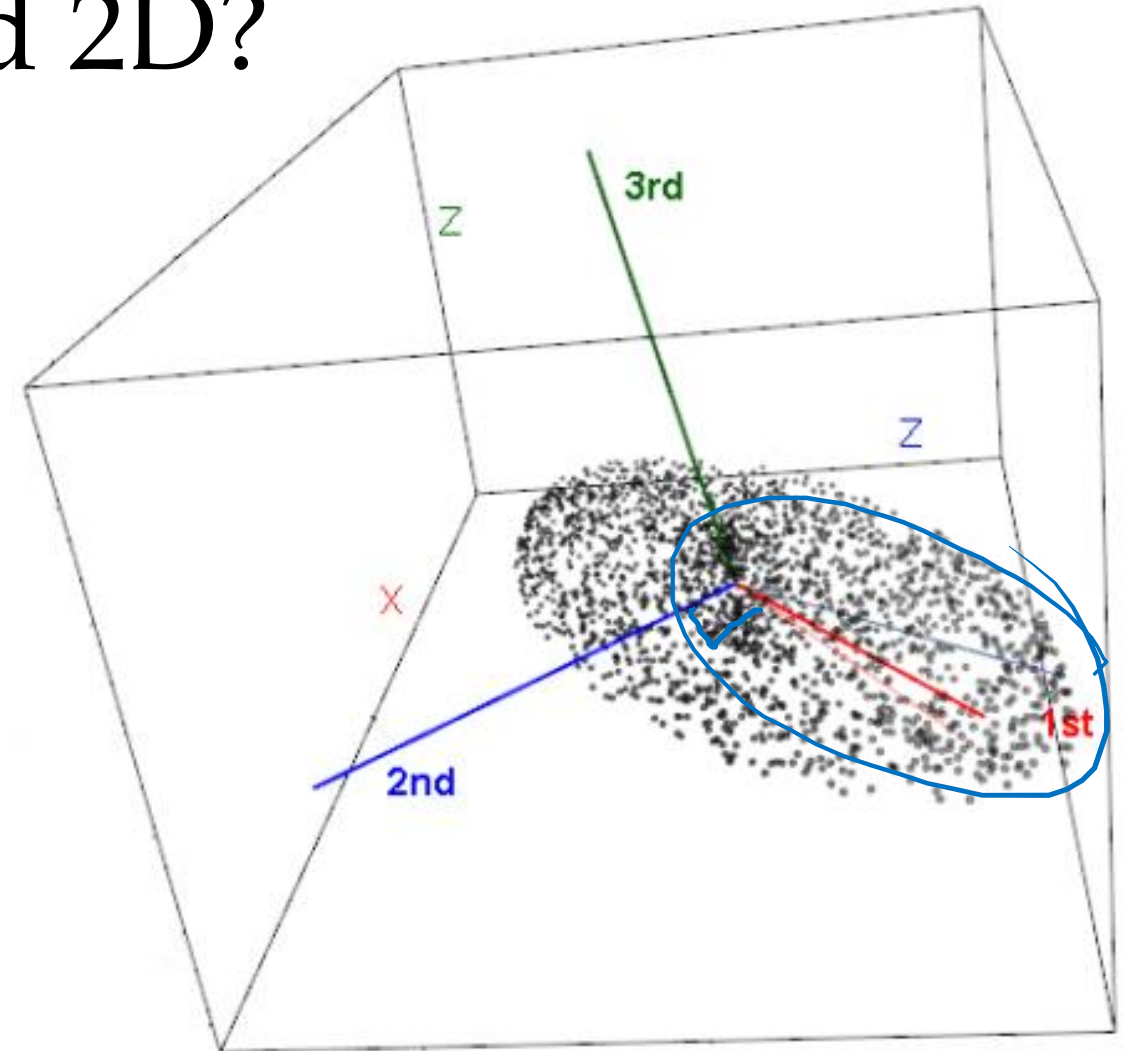
Scree Plot

TERMINOLOGY ALERT!!!! A **Scree Plot** is a graphical representation of the percentages of variation that each PC accounts for.

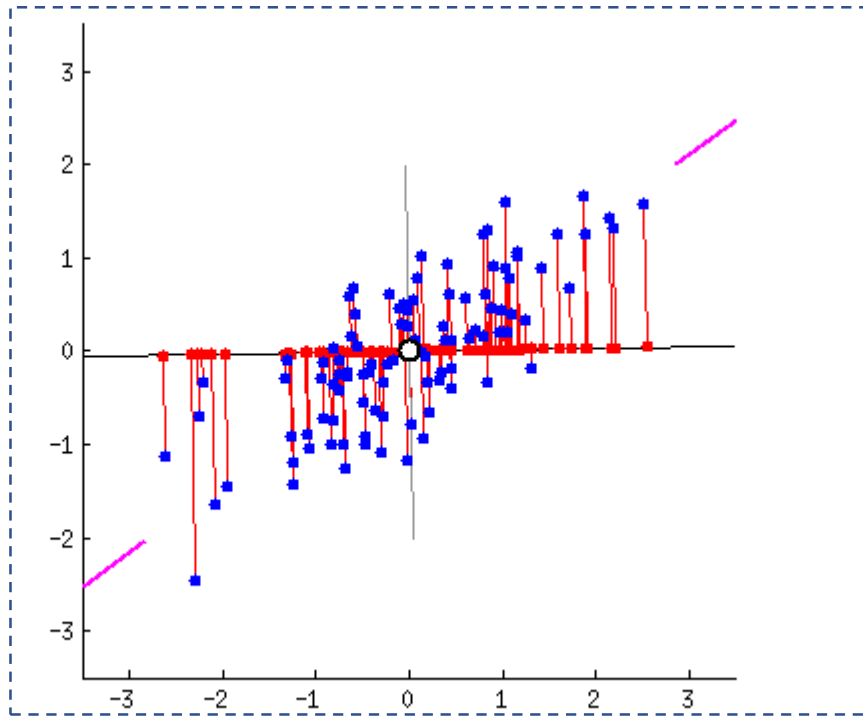


What happens beyond 2D?

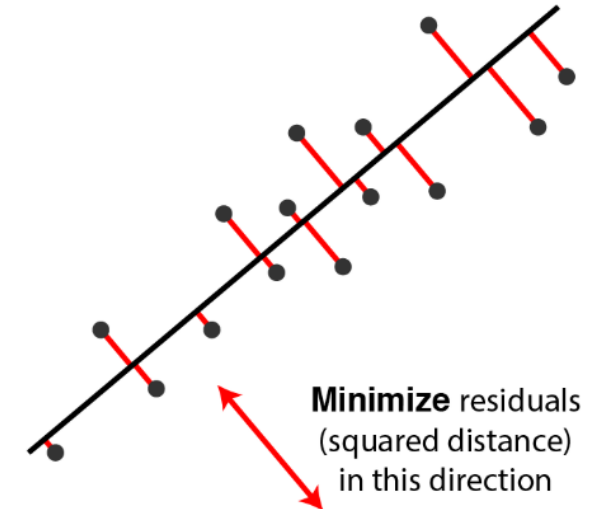
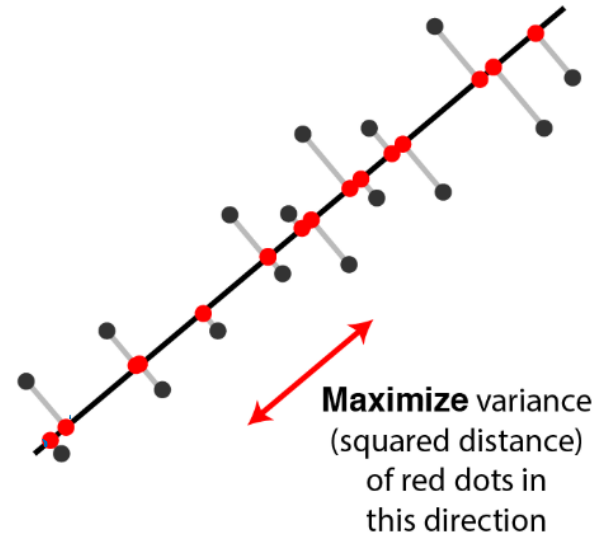
- Rigidly rotate the coordinate axes to new (principal axes) such that:
 - principal axis 1 corresponds to the highest variance,
 - axis 2 has the next highest variance,
 - . . . ,
 - and axis p has the lowest variance.
 - All principal axes are uncorrelated (orthogonal).



In Summary...



Source: [Stackoverflow](#)



Source: Alex Williams

- Project D-dimensional data onto an M-dimensional subspace that retains as much information as possible
- Informally: information = diversity = variance

PCA Algorithm Overview

1. Standardize the data (zero mean)
2. Obtain the Eigenvectors and Eigenvalues
3. Choose the k eigenvectors that correspond to the k largest eigenvalues
4. Construct the projection matrix \mathbf{W} from the selected k eigenvectors.
5. Transform the original dataset \mathbf{X} via \mathbf{W} to obtain a k -dimensional feature subspace \mathbf{Y} .



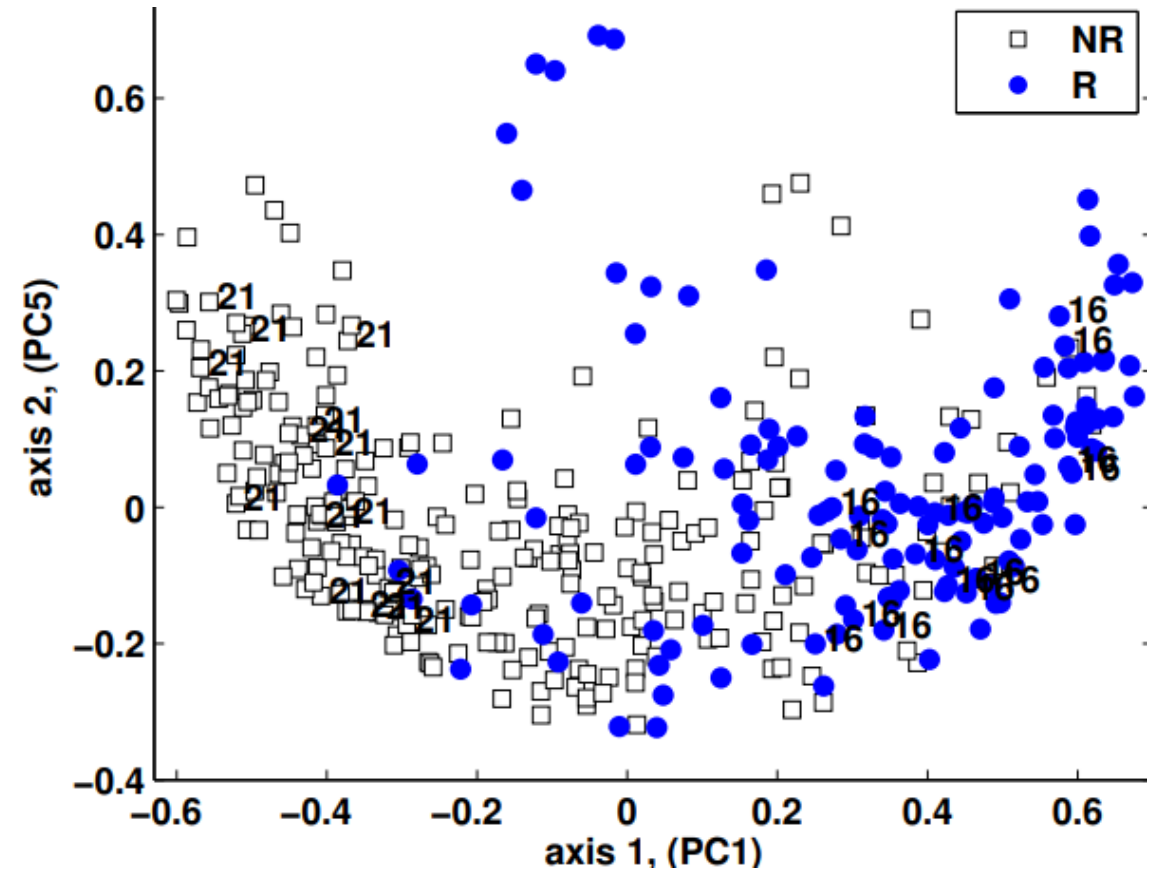
interpretation of data by examining principal components



discard components for **dimensionality reduction**

Visualization of Principle Components

- PCA is commonly applied to datasets to assess information content.
- Example:
 - PCA applied on patient dataset and samples projected on top principal components.
 - Patients with known treatment outcomes are coloured identified (responders vs non-responders).



Source: Khodayari-Rostamabad (2011)

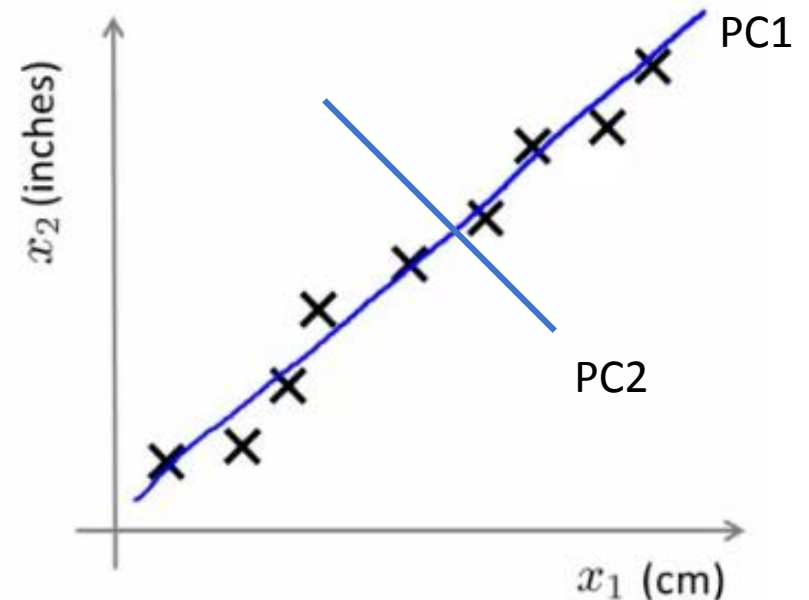
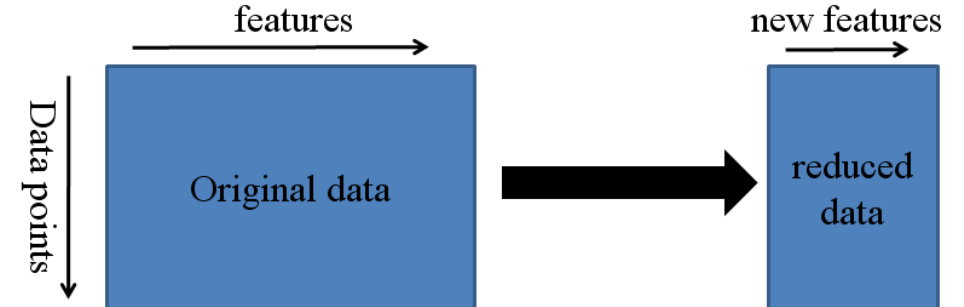
Dimensionality Reduction

- Projecting onto principal components can improve model/algorithm performance.

- Reduces the number of parameters and required complexity

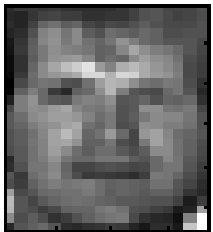
- Example:

- PC1 captures most of the variance (information).
 - PC2 limited variance and projections onto PC2 would look similar (or the same) for all samples.

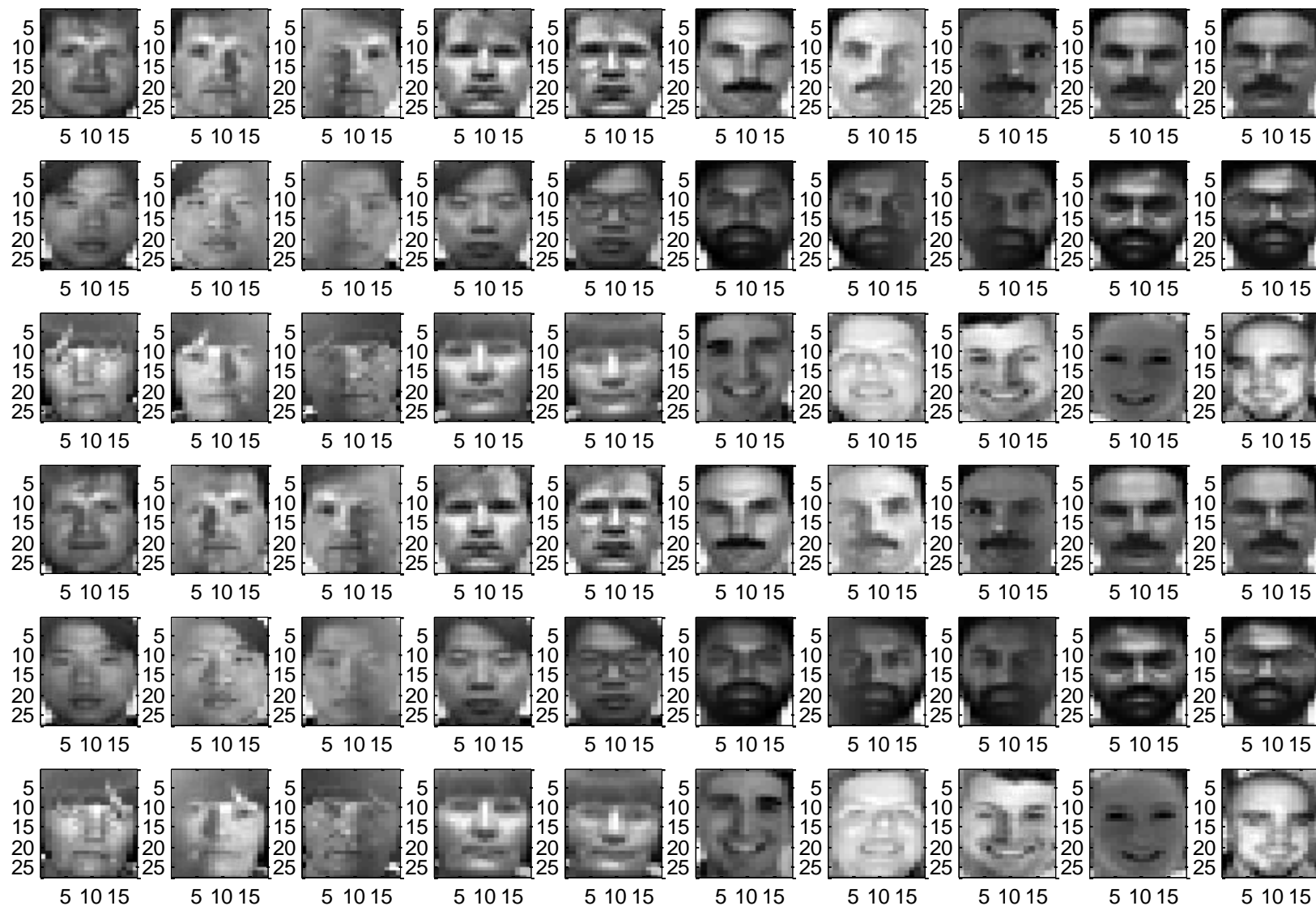


What about Images?

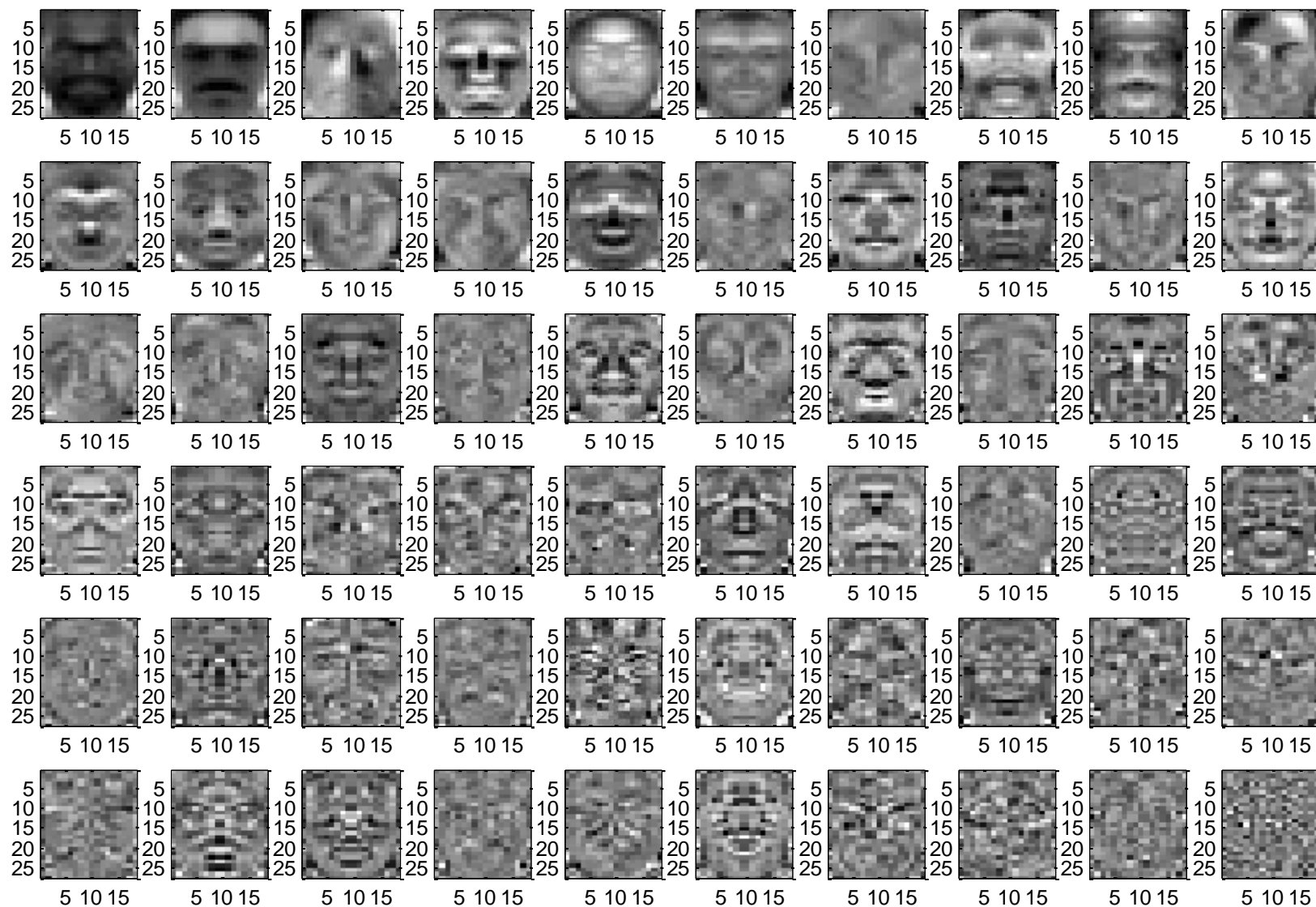
- Images?
 - Yes, we can use PCA! (Turk, Pentland, 1991)
- Each 150 pixel \times 150 pixel image can be represented by a 22,500-long vector



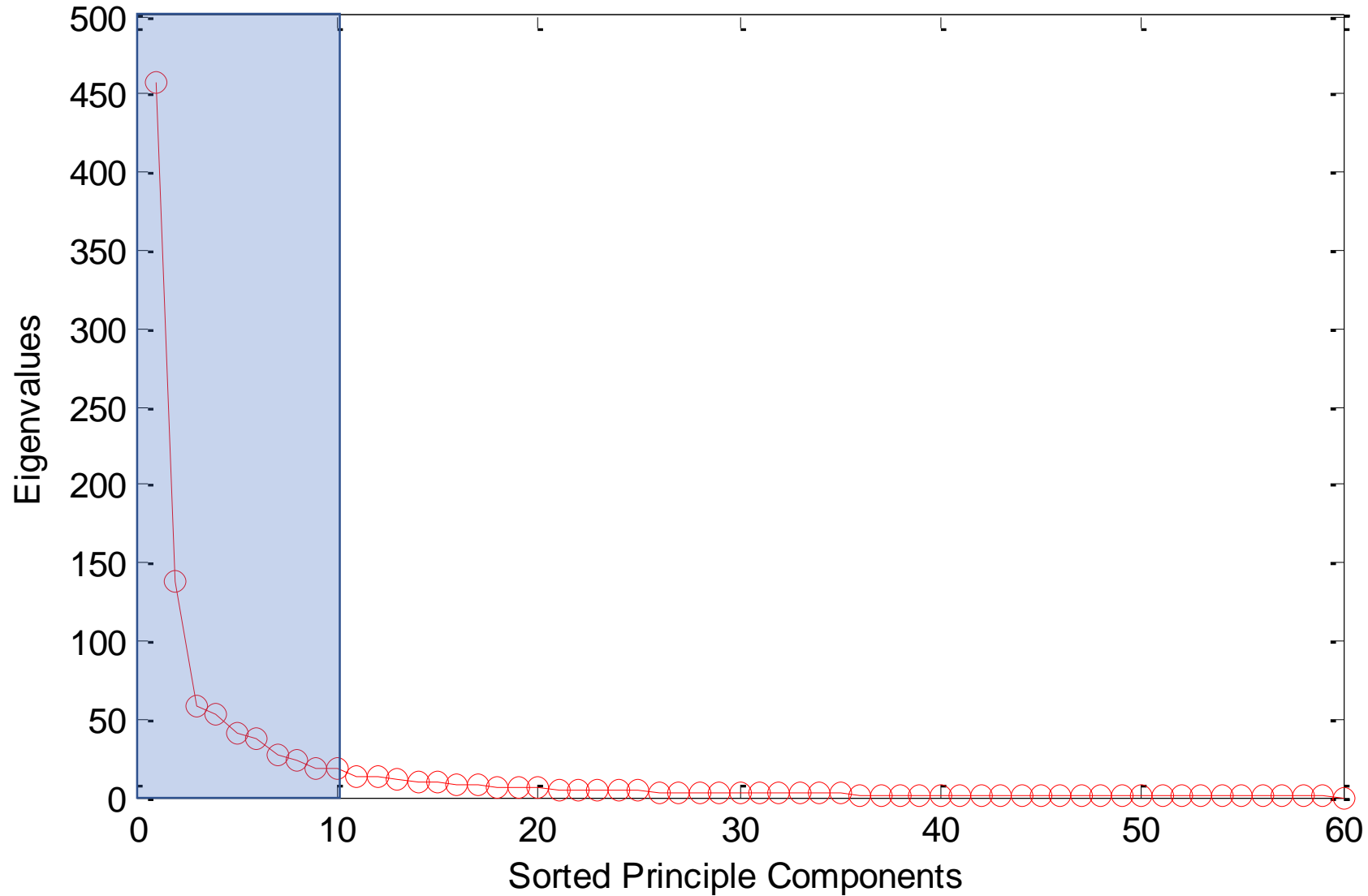
Normalized Face Data



The Eigenfaces



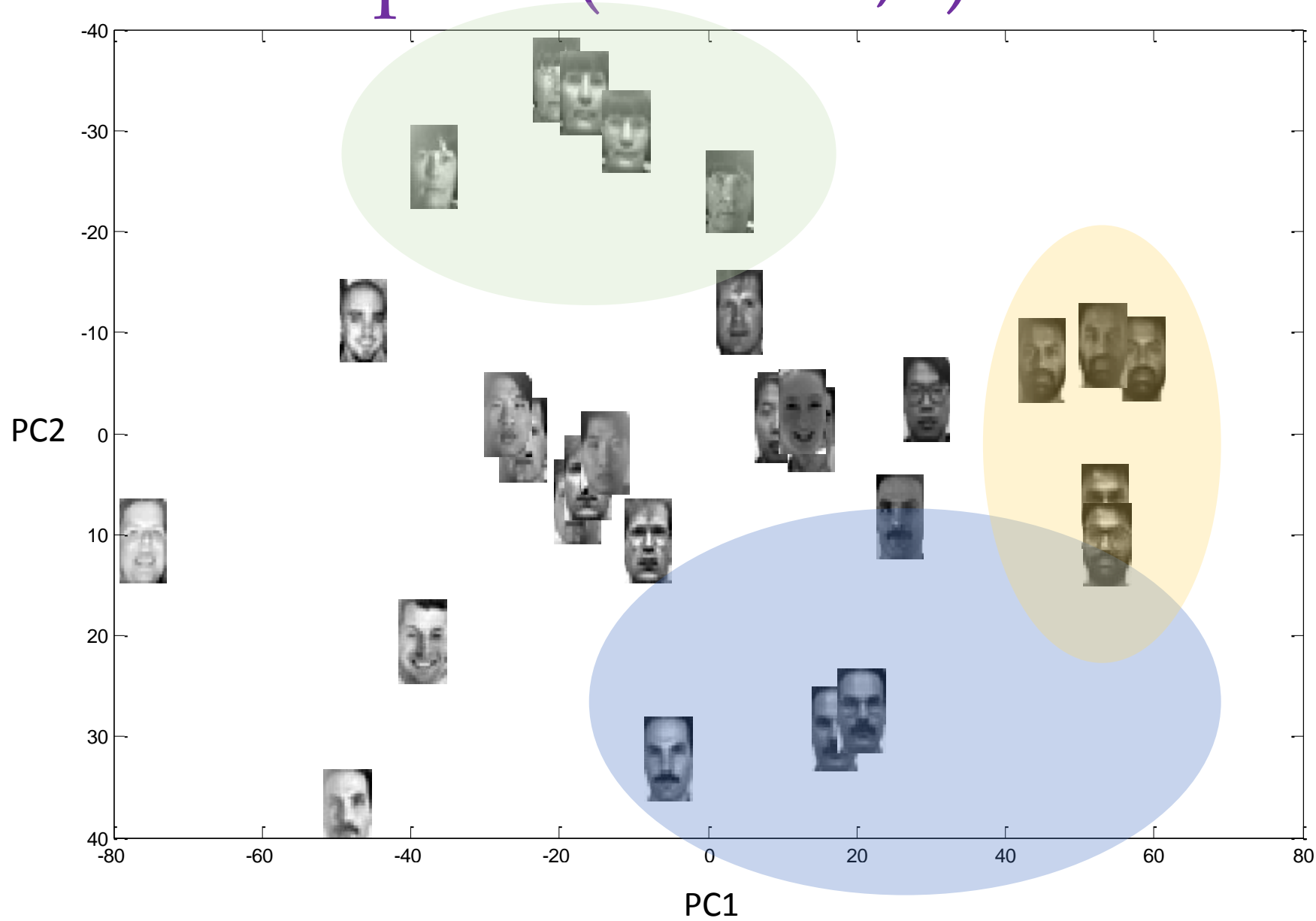
Eigenvalues for each Eigenface



Reconstructing a Face from PCs

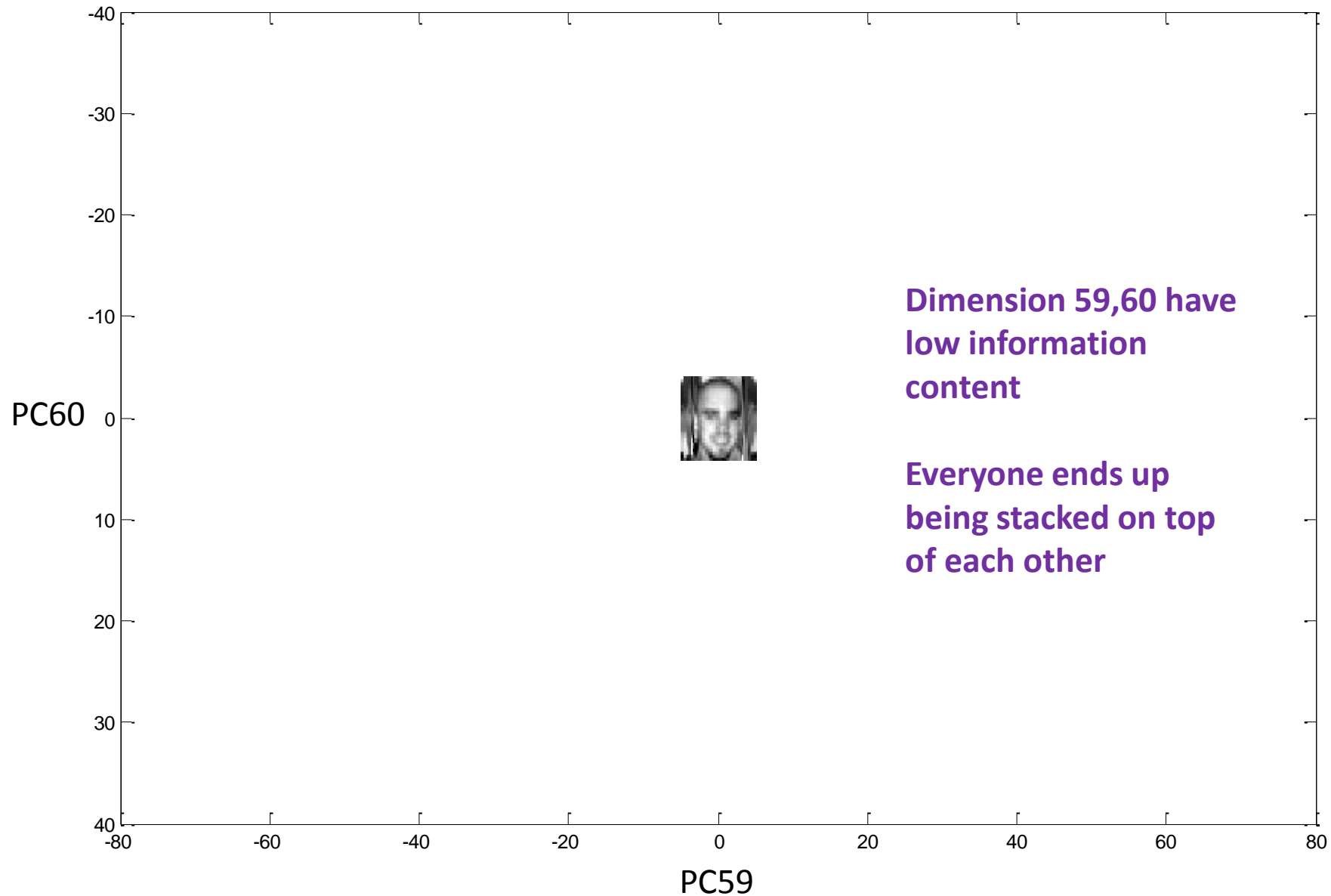


Faces in PC-space (Dims 1,2)



**Are there
any patterns
presented?**

Faces in PC-space (Dims 59,60)



Data Security

- To reconstruct data we require the mean and standard deviation used for normalization
- Can be used to **secure your datasets** as described for a sample dataset on Kaggle:

“It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to **confidentiality issues, we cannot provide the original features** and more background information about the data. **Features V1, V2, ... V28 are the principal components obtained with PCA**, the only features which have not been transformed with PCA are 'Time' and 'Amount’.”

Source: [Kaggle Credit Card Fraud](#)

Limitations of PCA

- High dimensional data can be problematic. For example, when you have short and wide data, PCA may not work as well.
- We can speed up eigenfaces decomposition by exploiting smaller dimension of $m \times n$, to do this we need a way to work with rectangular features...

Singular Value Decomposition (SVD) to the rescue!

Next Time

- Week 7 Tutorial
 - Project 3 is on PCA and is due Friday, Mar 11
- Reading Assignment
- Lecture – Dimensionality Reduction Part 2
 - Vector Calculus
 - SVD
 - Singular Value Decomposition (SVD)
 - Reconstruction
 - Applications
 - More on PCA