# Singular Value Decomposition

**Geometric Algorithms**
**Lecture 26**

CAS CS 132

# Objectives

1. Introduce the **singular value decomposition** (probably the most important matrix decomposition for computer science)

2. Talk very briefly about what to do after this course if you want (or have to) to see more linear algebra
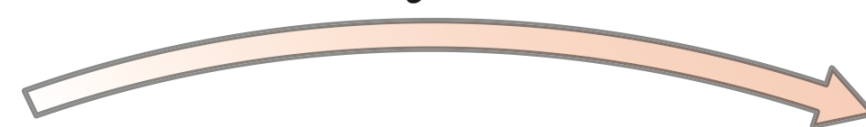
3. Fill out course evals(!)

# Motivation

# Question

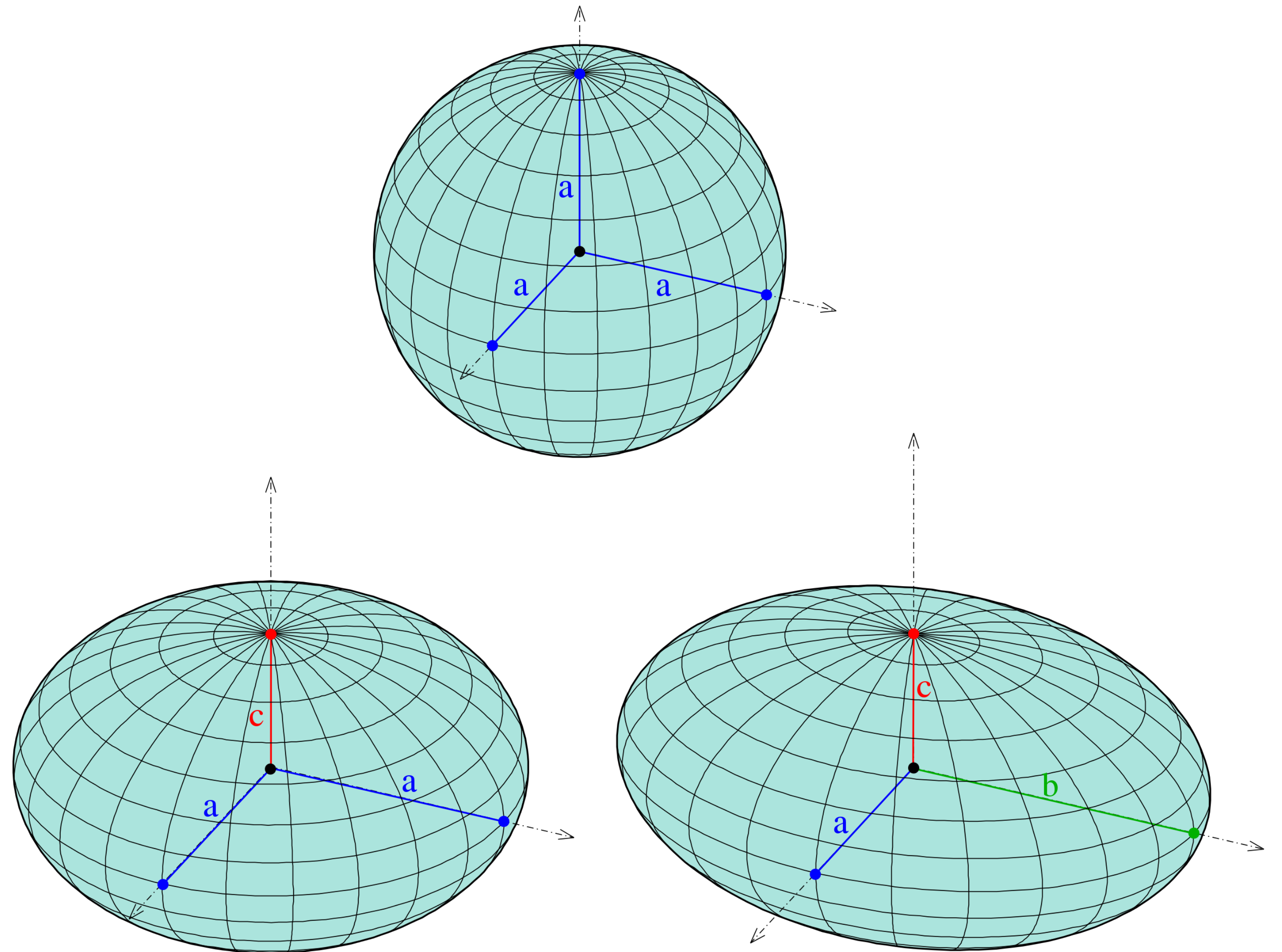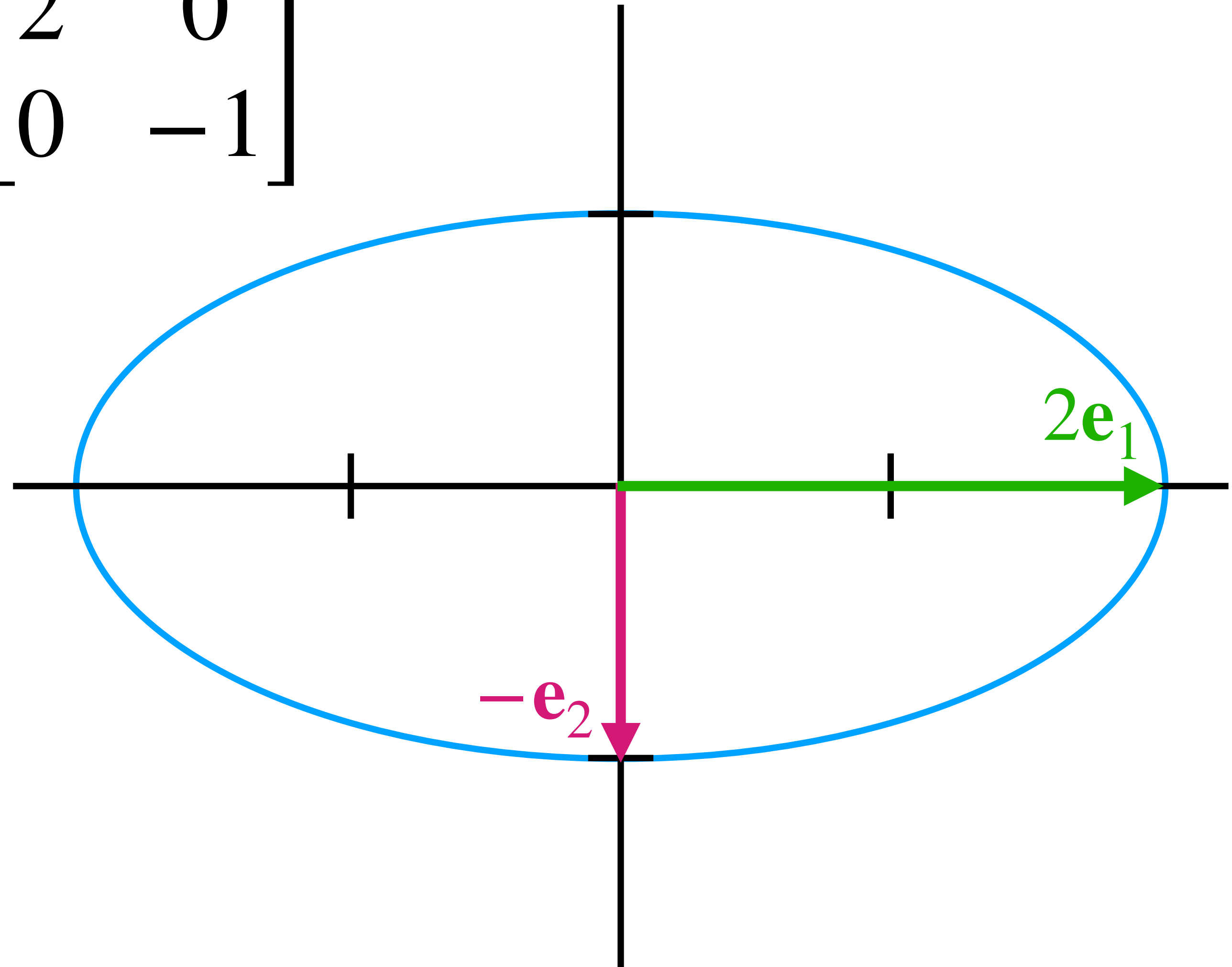*What shape is a the unit sphere after a linear transformation?*
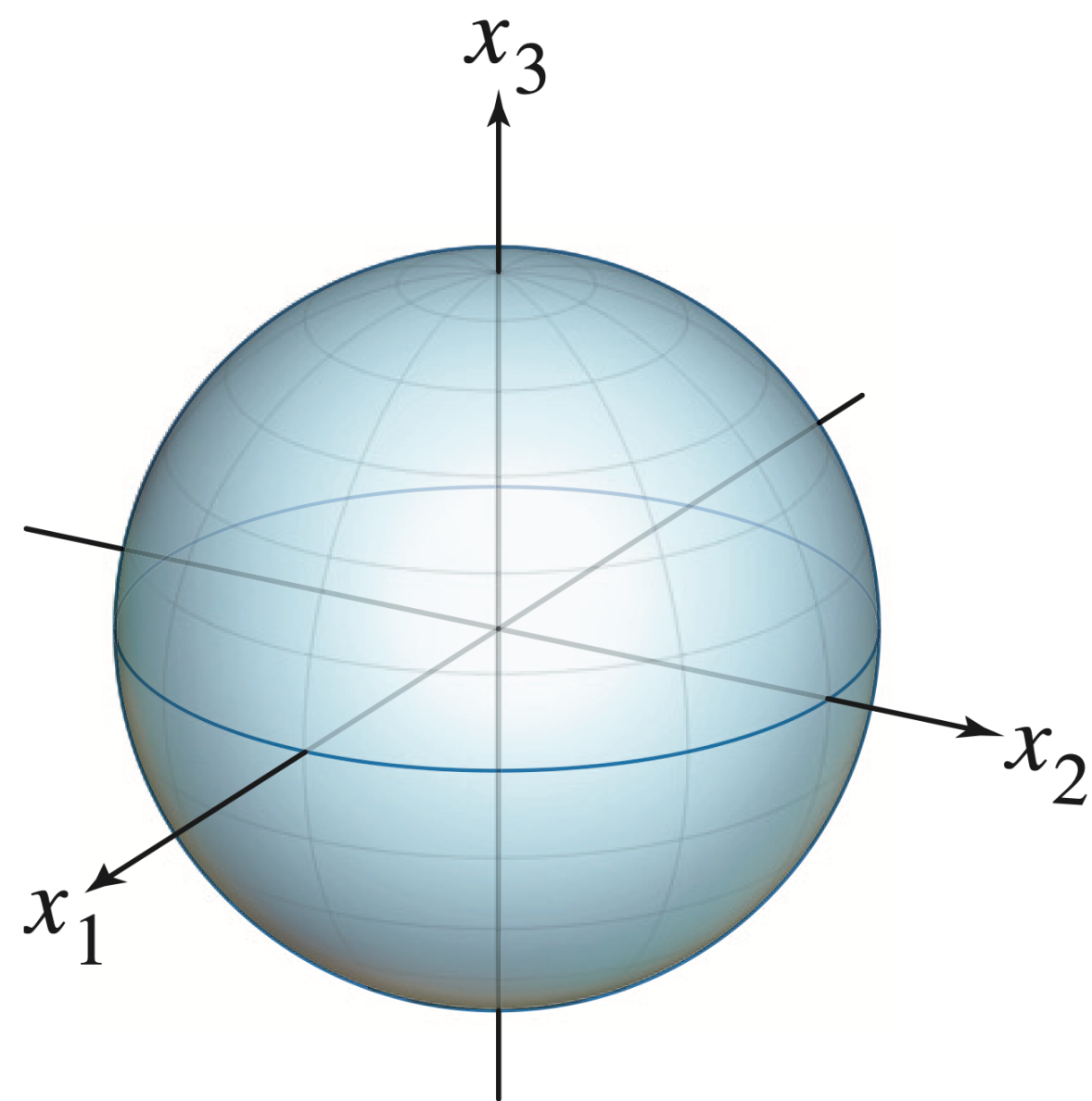
# Ellipsoids
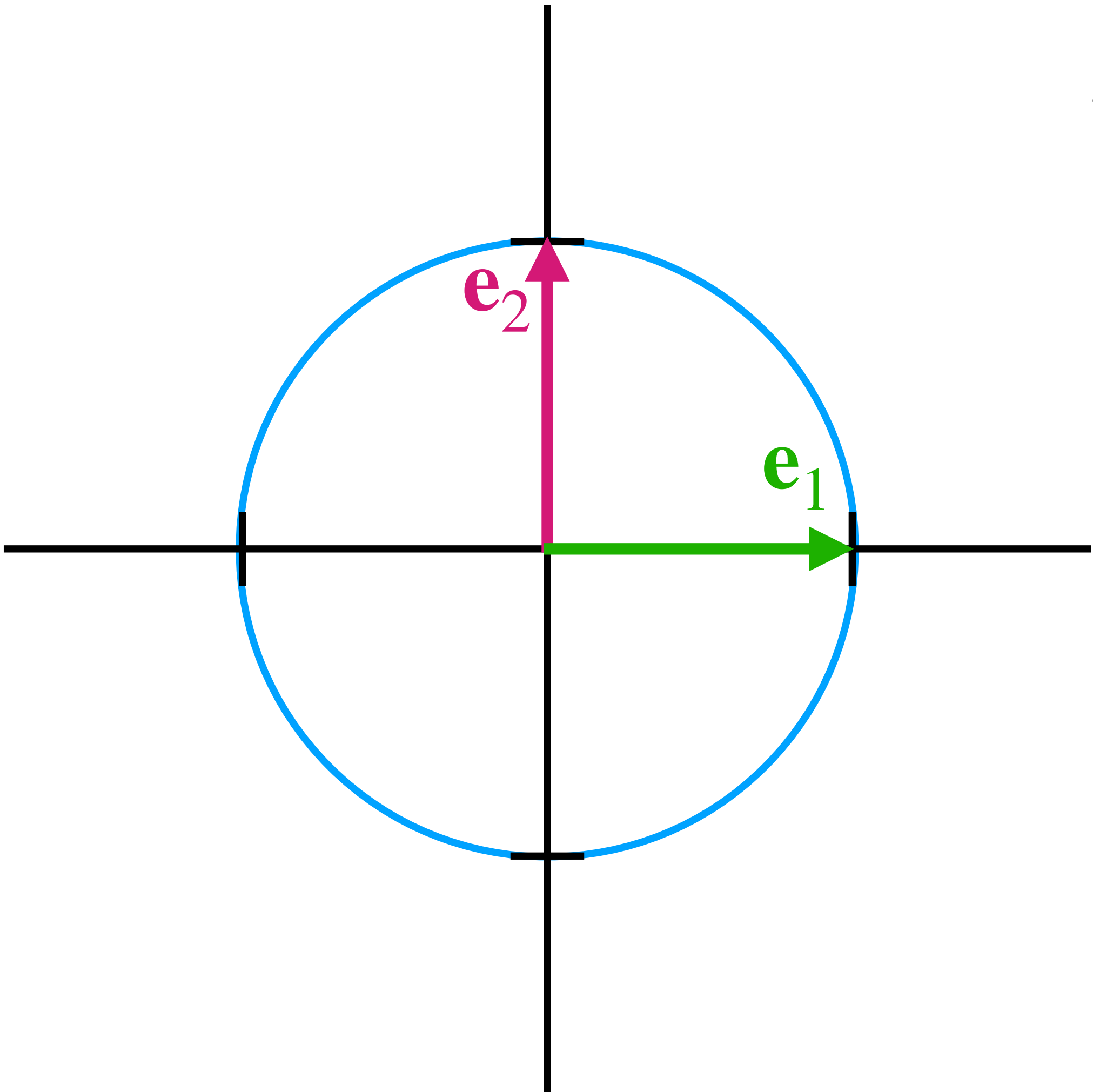
Ellipsoids are spheres "stretched" in orthogonal directions called the **axes of symmetry** or the **principle axes.**

**Linear transformations maps** <u>spheres</u> **to** <u>ellipsoids</u>.

# Simple Example : Scaling Matrices

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

# The Picture

# The Picture



Multiplication by $A$

who get mapped here?

$(18, 6)$

$(3, -9)$

# The Picture

$\|\mathbf{x}\| = 1$
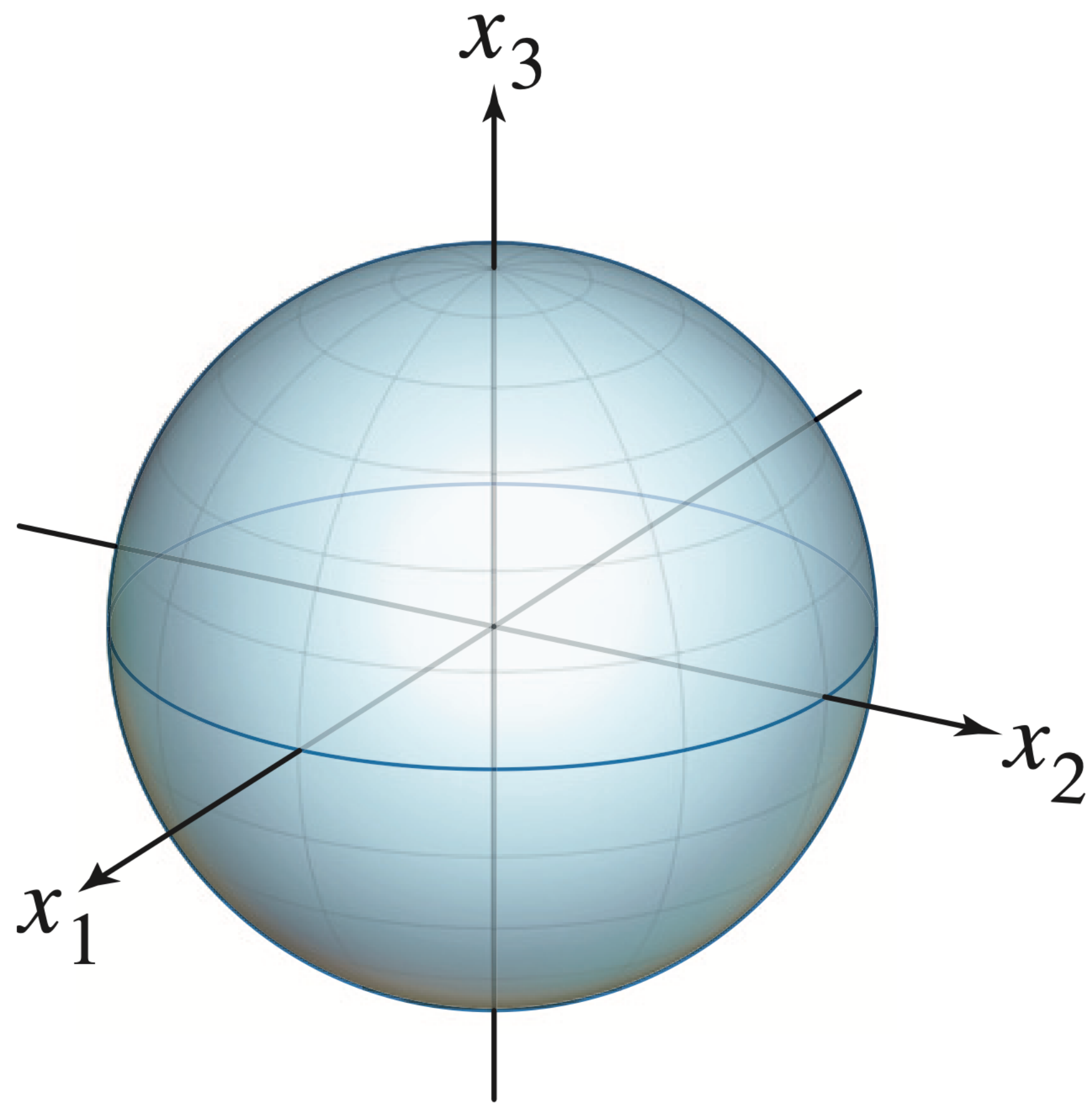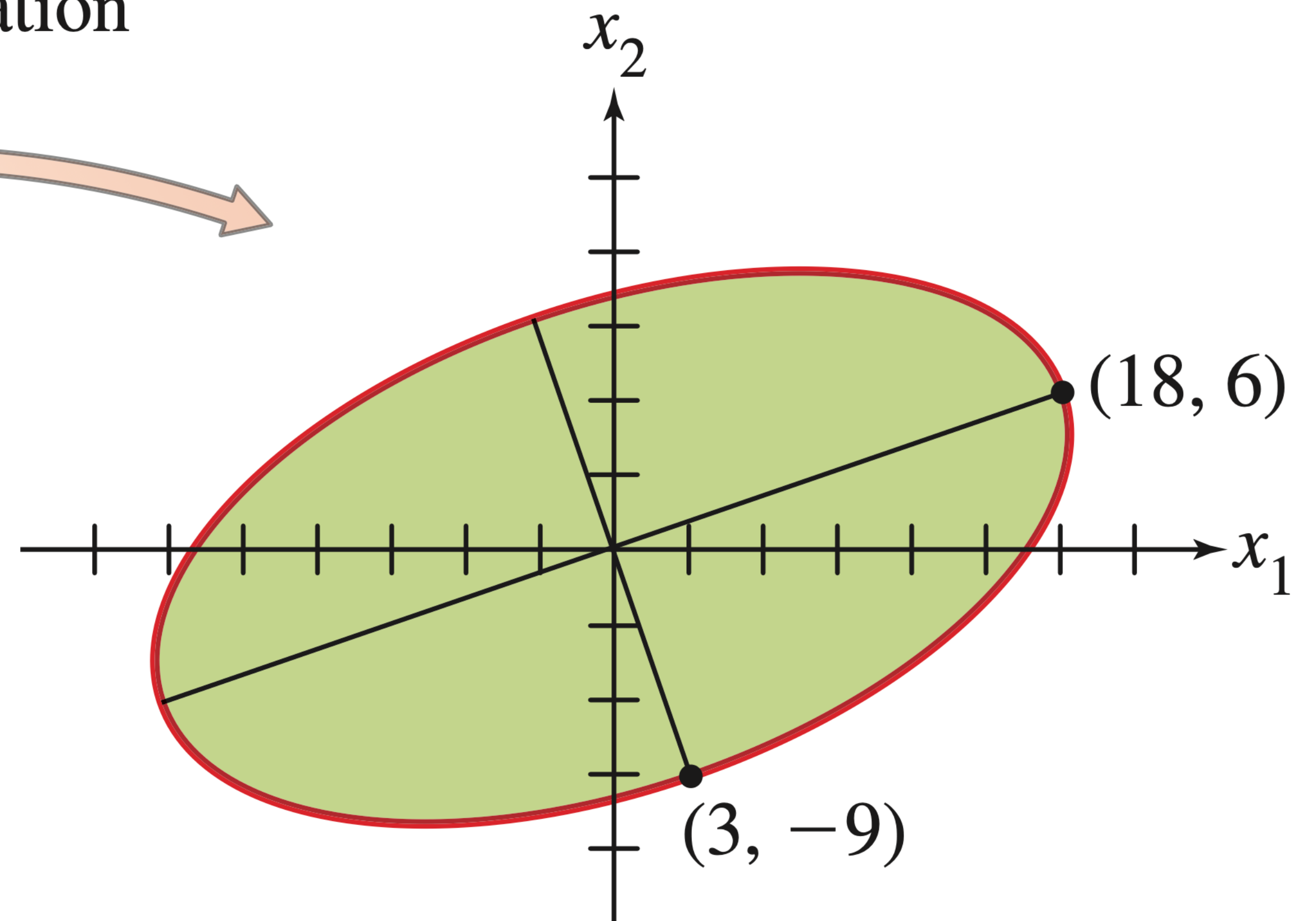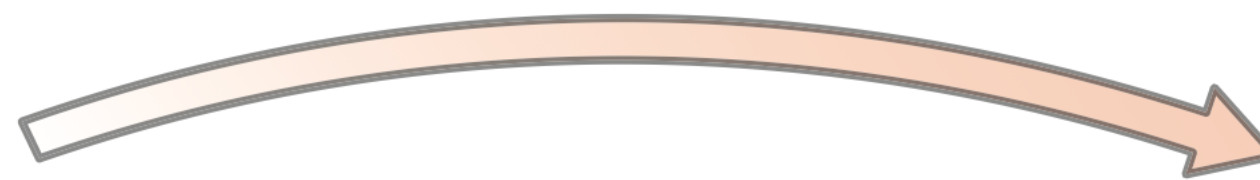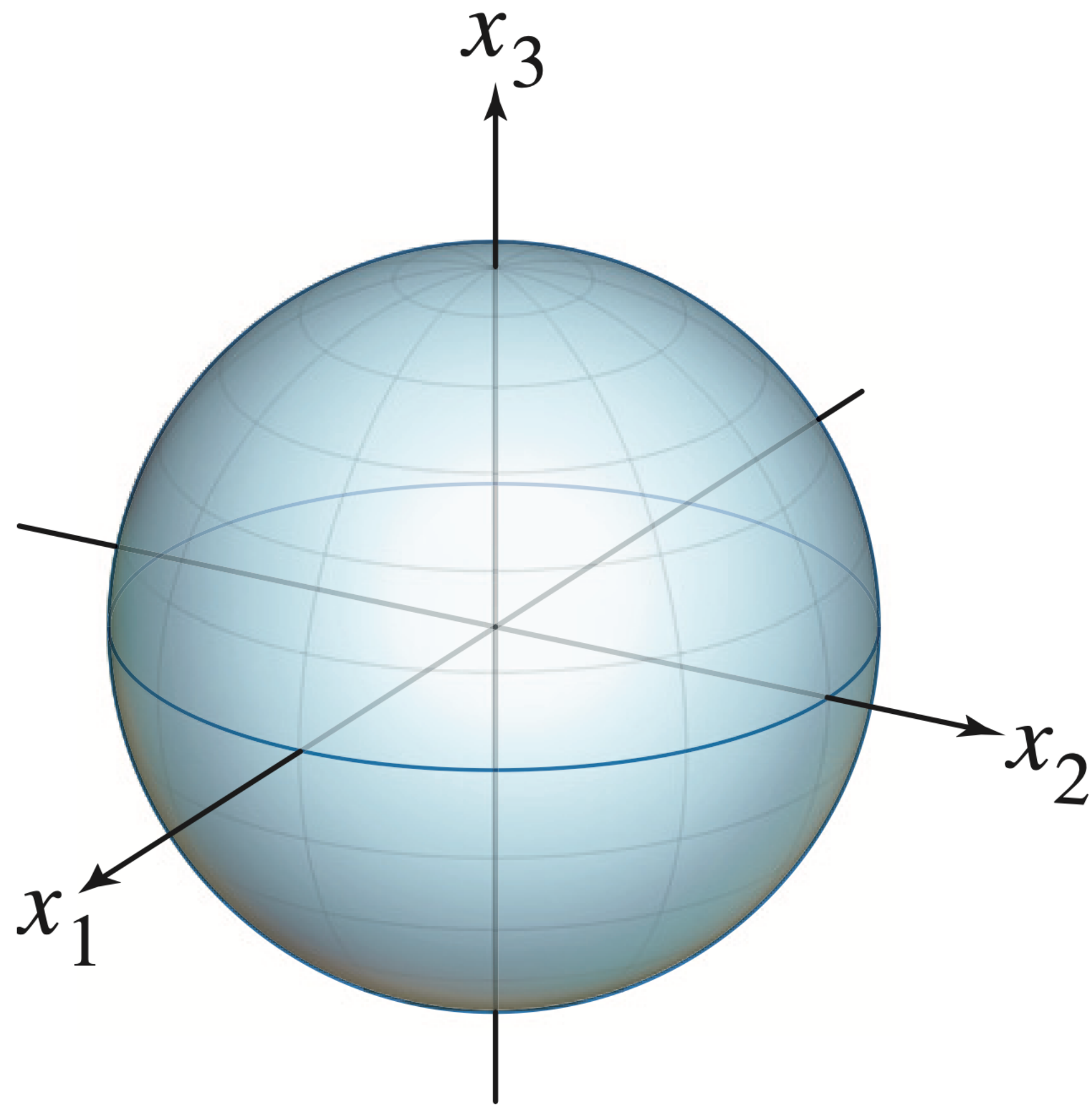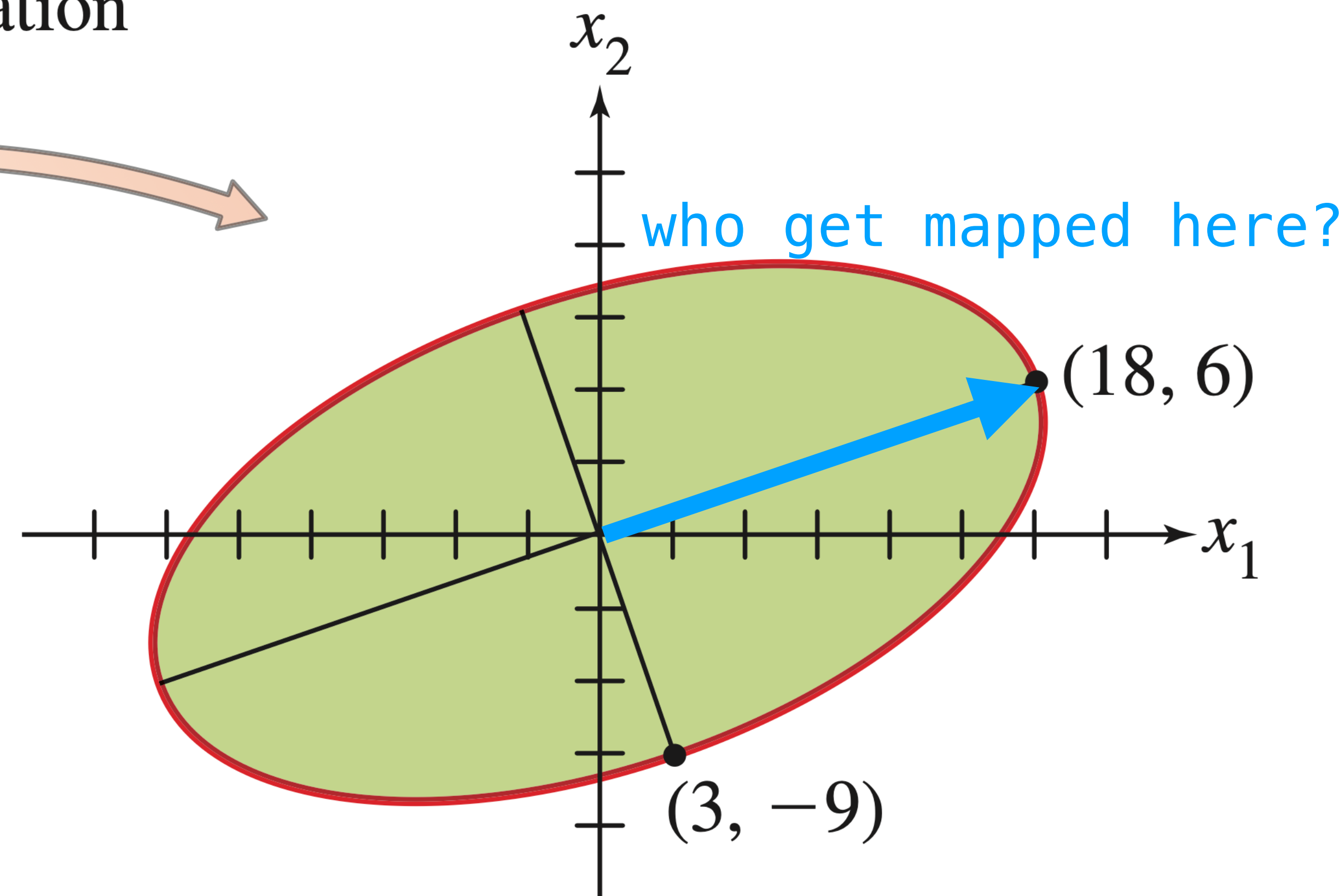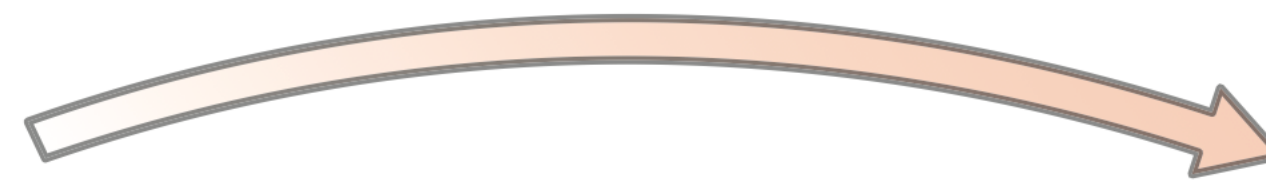
The longest end of the ellipse is the solution to a constrained optimization problem

Multiplication by $A$

$x_3$

$x_2$

$x_1$

$\|A\mathbf{x}\| = \max_{\|\mathbf{y}\|=1} \|A\mathbf{y}\|$

$x_2$

$x_1$

$(18, 6)$

$(3, -9)$

# The Picture



$$\|A\mathbf{x}\| = \max_{\|\mathbf{y}\|=1} \|A\mathbf{y}\|$$

$(18, 6)$

$(3, -9)$
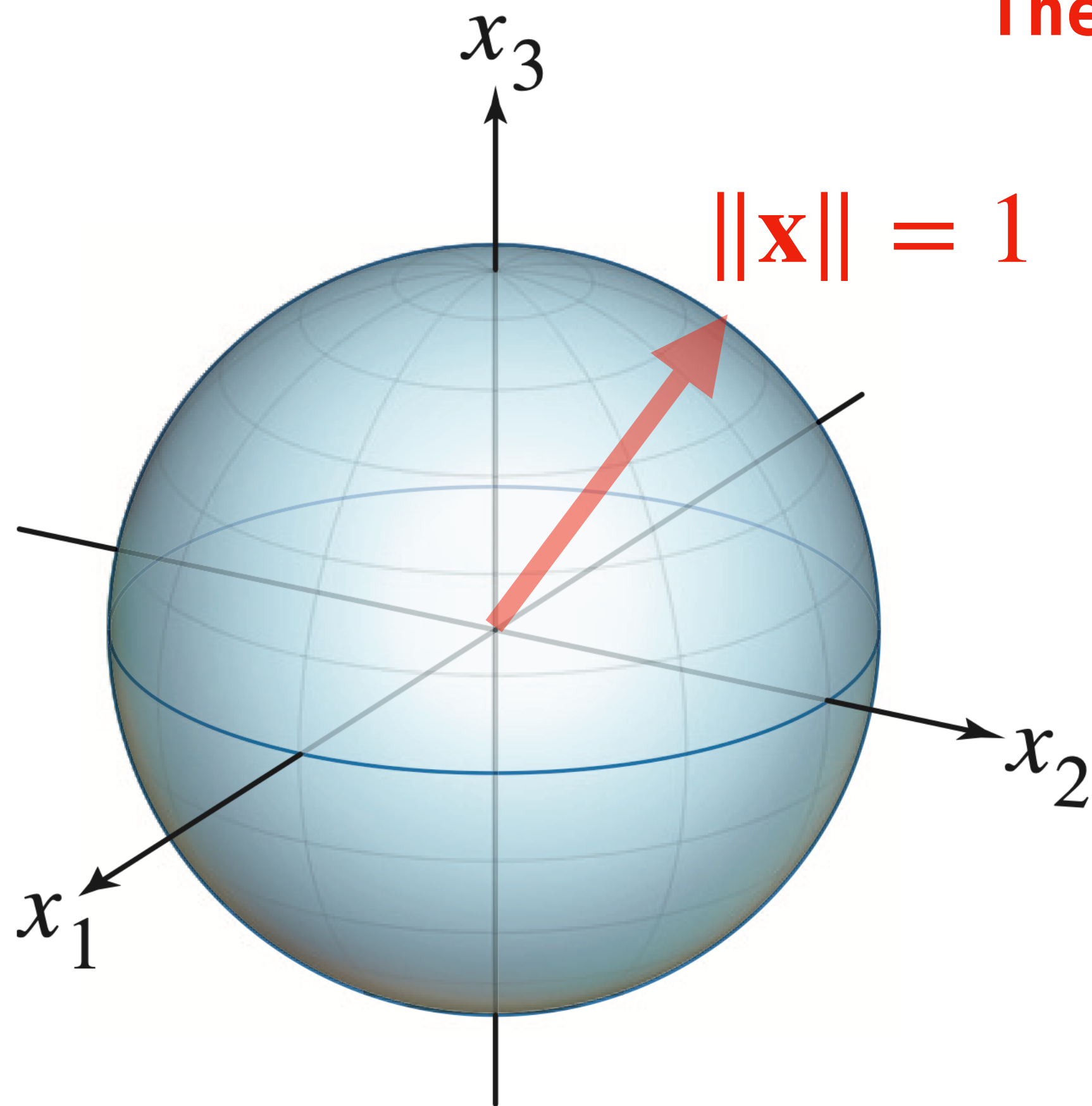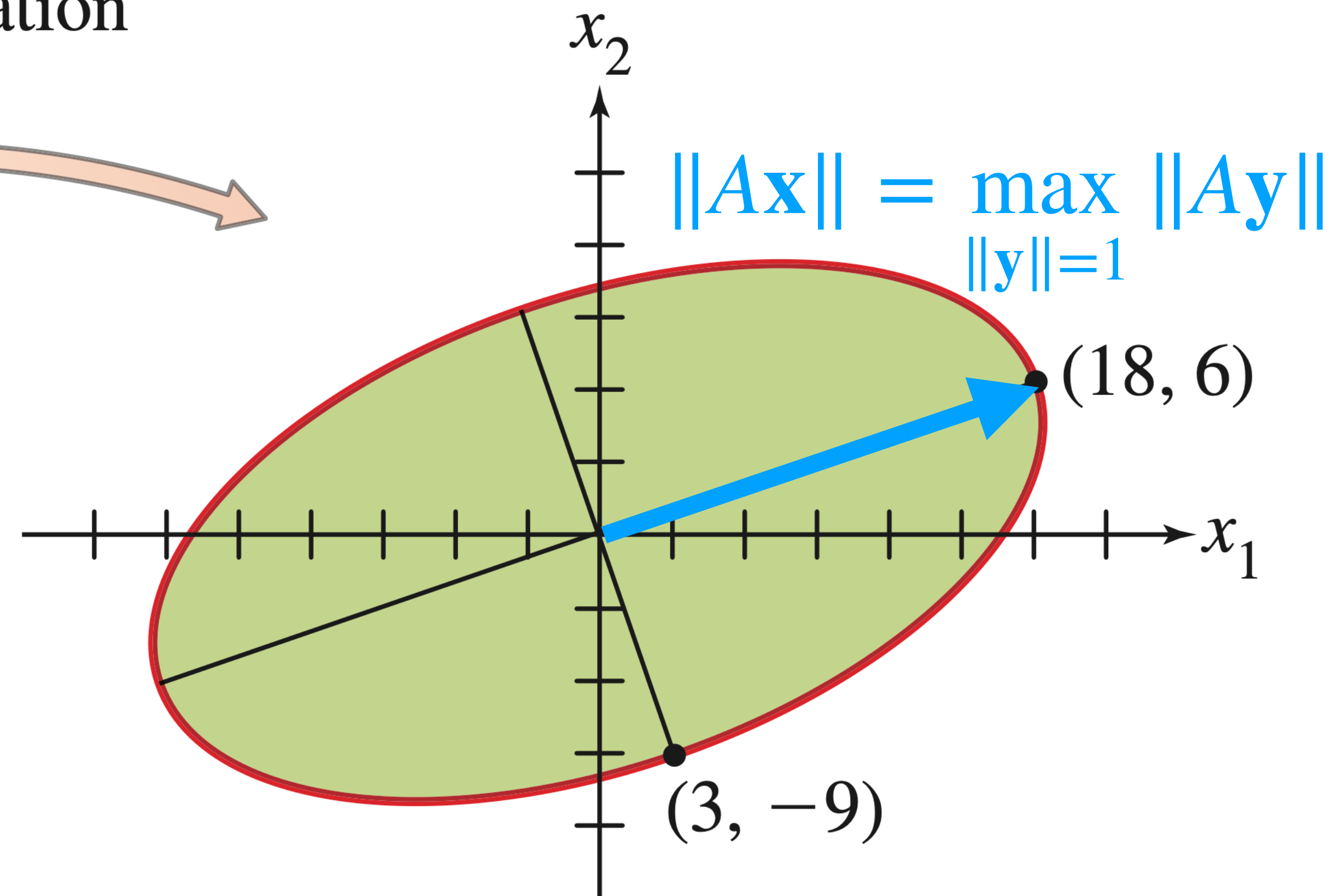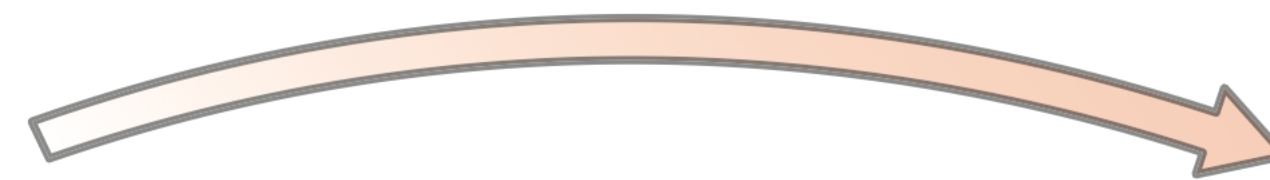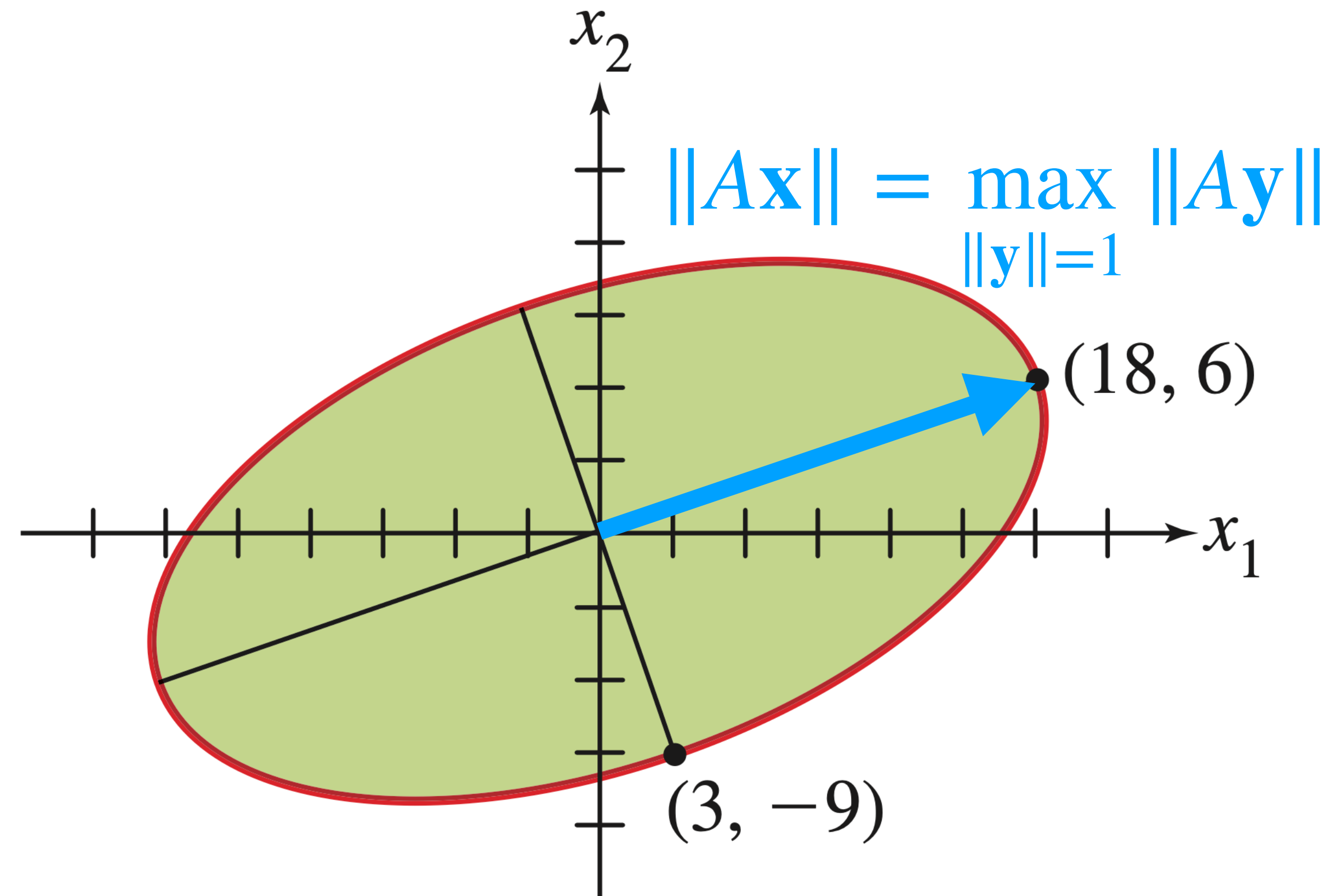
This is not a quadratic form...

# The Picture



$$\|A\mathbf{x}\|^2 = \max_{\|\mathbf{y}\|=1} \|A\mathbf{y}\|^2$$

**But this is.**

$(18, 6)$

$(3, -9)$
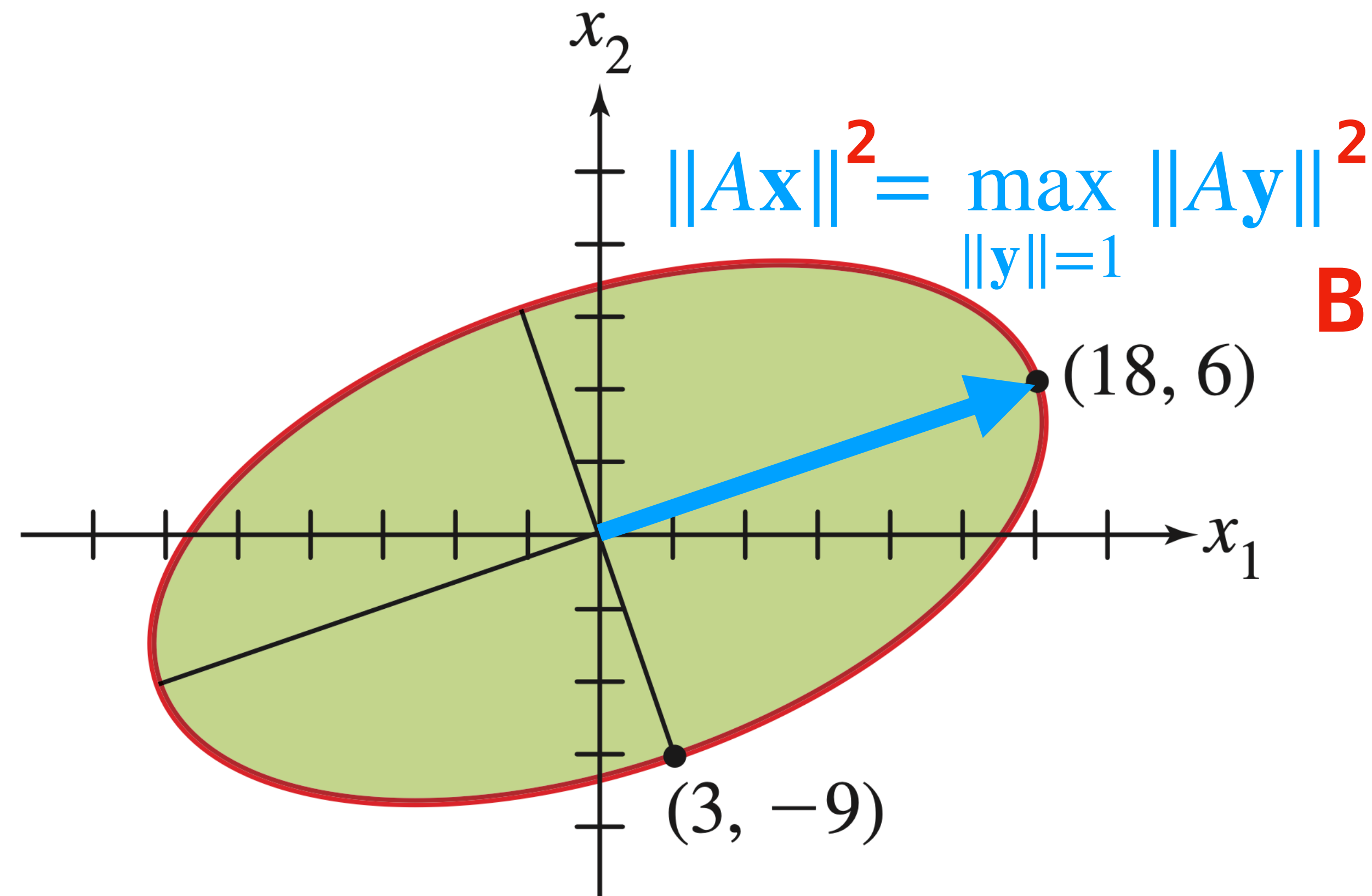
This is not a quadratic form...

# A Quadratic Form

What does $\|A\mathbf{x}\|^2$ look like?:

$$\|A\vec{x}\| = \left(\sqrt{\langle Ax, Ax \rangle}\right)^2 = \langle A\vec{x}, A\vec{x} \rangle$$

$$= (Ax)^T A\vec{x}$$

$$= \vec{x}^T A^T A \vec{x}$$

$A^T A$ — symmetric (exercise.)

$$= Q(x)$$

$$\max_{\|x\| = 1} Q(x)$$

# The Picture



The largest eigenvector of $A^T A$

$\mathbf{v}_1$

$A^T A \mathbf{v}_1 = \lambda_1 \mathbf{v}_1$

Multiplication by $A$

$\|A\mathbf{v}_1\| = \sqrt{\lambda_1}$

$(18, 6)$

$(3, -9)$
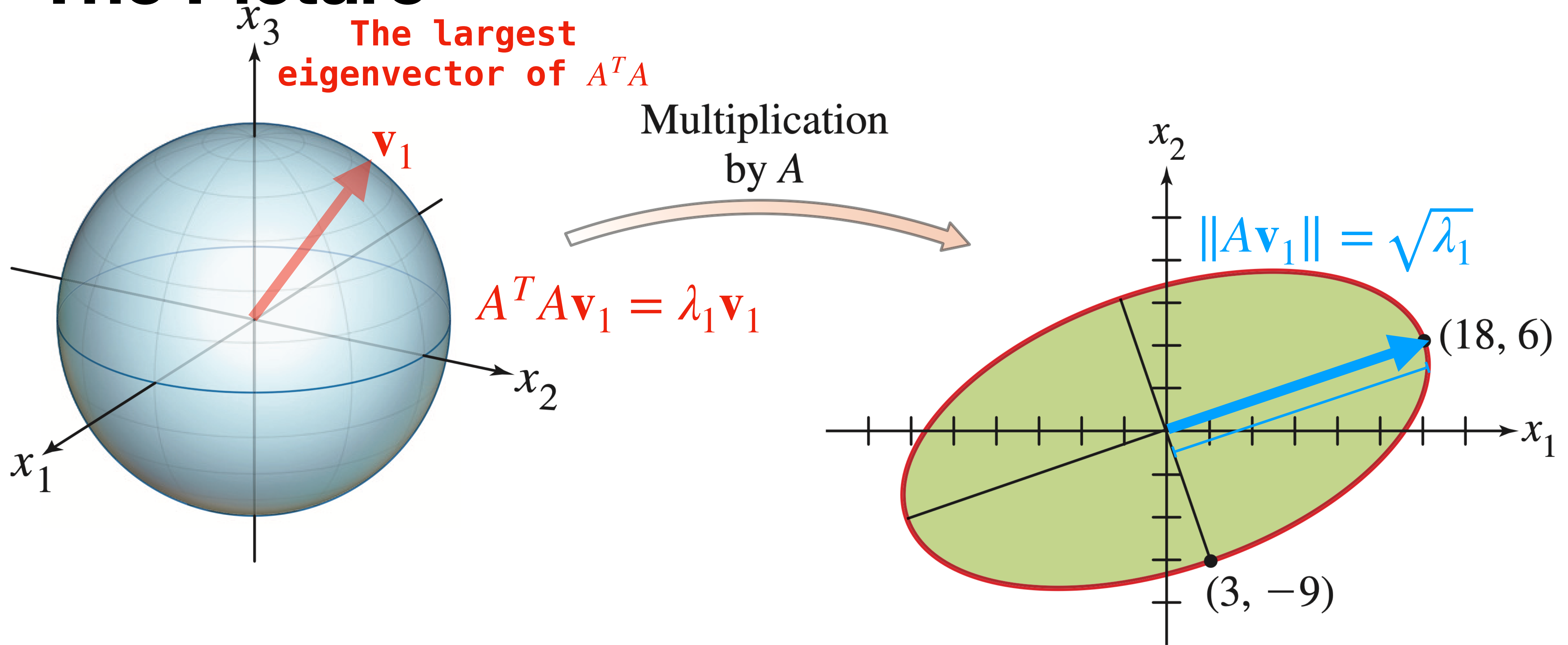
$\mathbf{v}_1$ solves the constrained optimization problem.

# The Picture
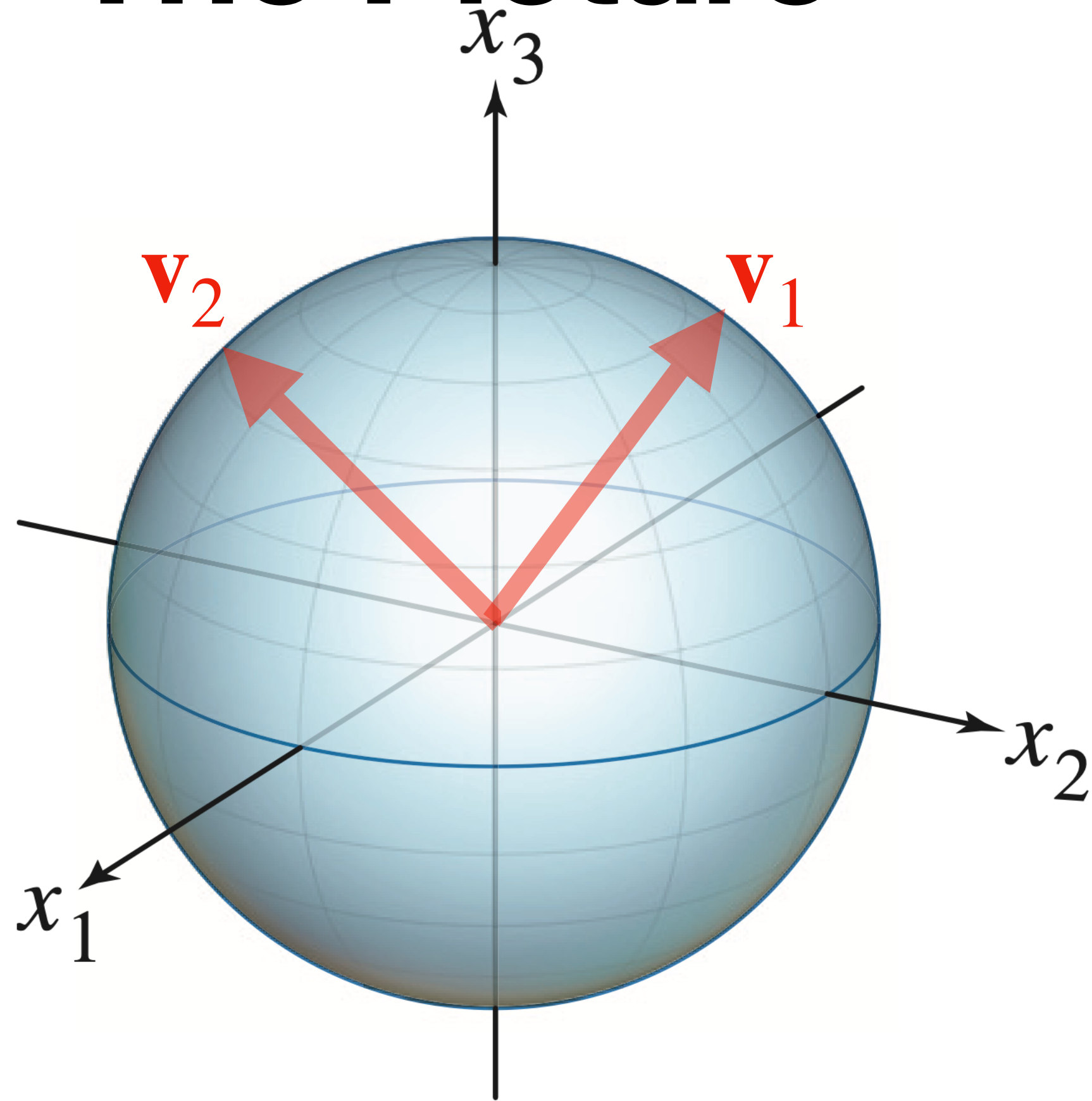


Multiplication by $A$
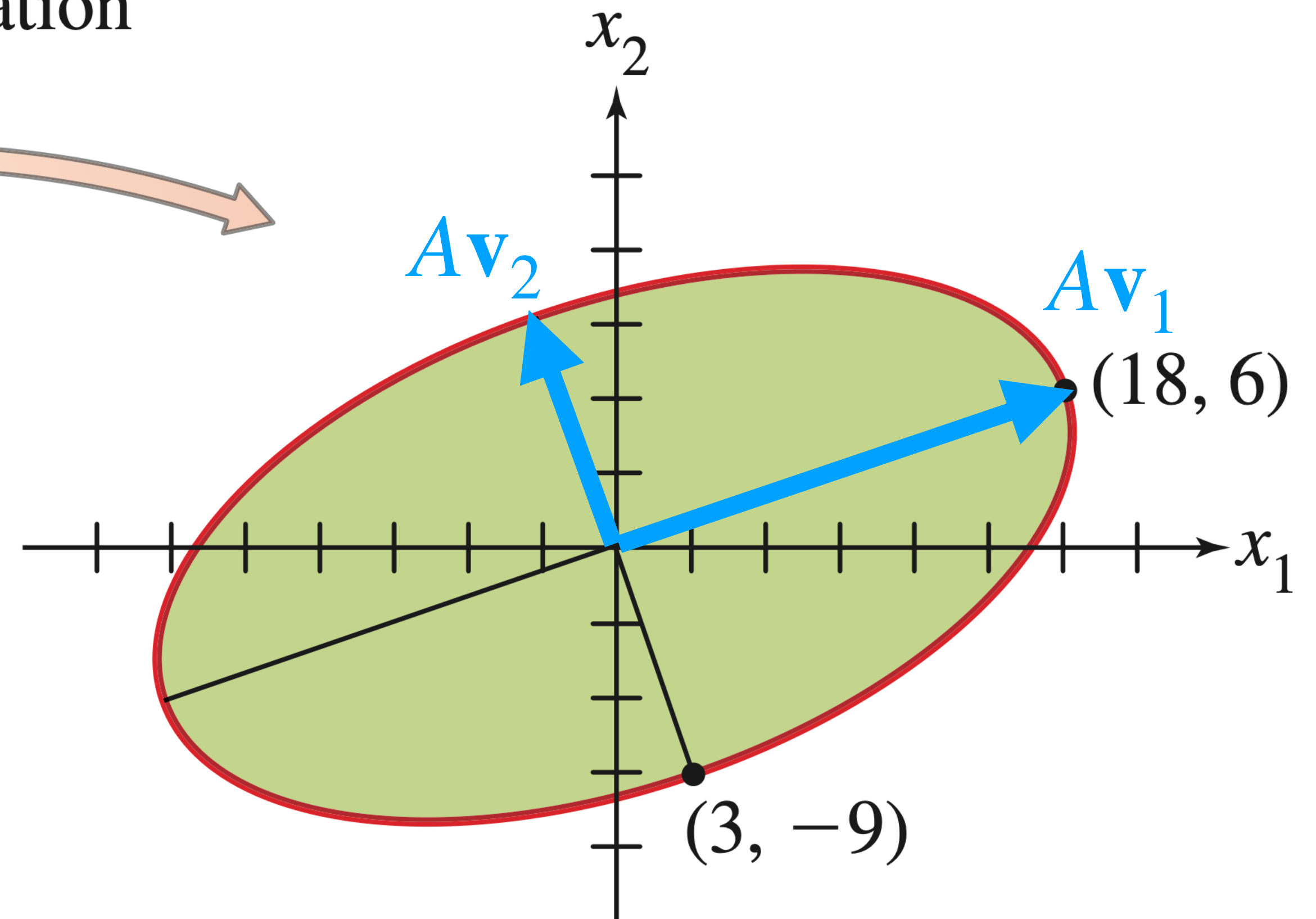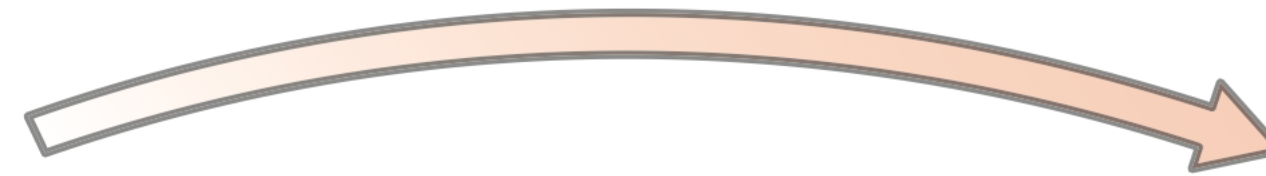
The second eigenvector is sent to the *minimum* principle axis

# Properties of $A^T A$

# Properties of $A^T A$

» It's symmetric $(A^T A)^T = A^T (A^T)^T = A^T A$

# Properties of $A^T A$

» It's symmetric

» So its <u>orthogonally diagonalizable</u> $A^T A = P D P^T$

*diagonal*

*orthonormal*

# Properties of $A^T A$

» It's symmetric

» So its <u>orthogonally diagonalizable</u>

» **There is an orthogonal basis of eigenvectors**

# Properties of $A^T A$

» It's symmetric

» So its <u>orthogonally diagonalizable</u>

» **There is an orthogonal basis of eigenvectors**

» It's eigenvalues are nonnegative

# Properties of $A^T A$

» It's symmetric

» So its <u>orthogonally diagonalizable</u>

» **There is an orthogonal basis of eigenvectors** $\|Av\|^2$

» It's eigenvalues are nonnegative $v, \lambda$ $v^T A^T A v$

$$v^T \lambda v$$

$$\lambda \|v\|^2 \quad \text{non-negative}$$

» **It's positive semidefinite**

# Singular Values

**Definition.** For an $m \times n$ matrix $A$, the **singular values** of $A$ are the $n$ values

$$\sigma_1 \geq \sigma_2 \ldots \geq \sigma_n \geq 0$$

where $\sigma_i = \sqrt{\lambda_i}$ and $\lambda_i$ is an eigenvalue of $A^T A$.

$\lambda_i \geq 0$

# Another picture

$$\|A\mathbf{v}_3\| = \sqrt{\lambda_3} = \sigma_3$$

$\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ are the eigenvectors of $A^T A$

c

b

a

$$\|A\mathbf{v}_1\| = \sqrt{\lambda_1} = \sigma_1$$

$$\|A\mathbf{v}_2\| = \sqrt{\lambda_2} = \sigma_2$$
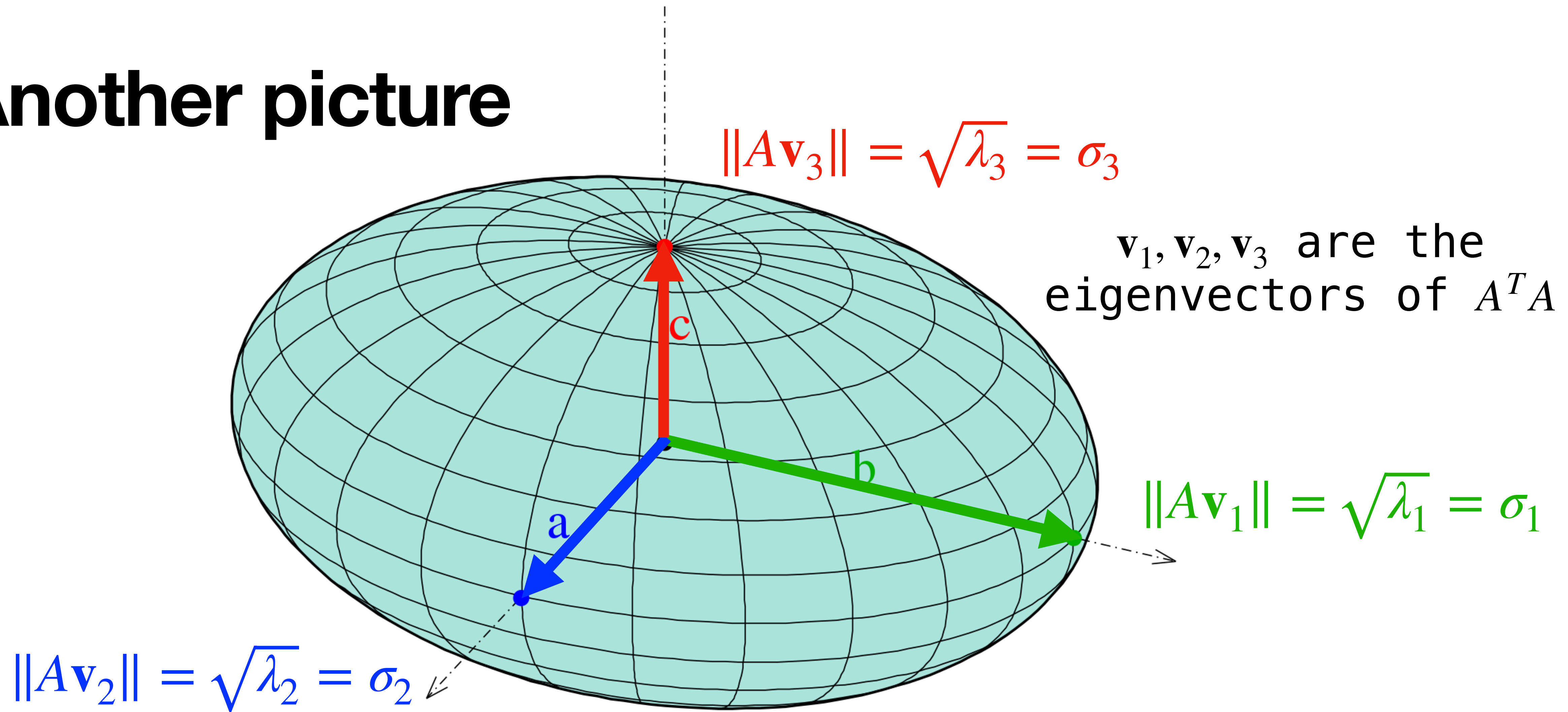
The **singular values** are the <u>lengths</u> of the *axes of symmetry* of the ellipsoid after transforming the unit sphere.

Every $m \times n$ matrix transforms the unit $m$–sphere into an $n$–ellipsoid

So <u>every</u> $m \times n$ matrix has $n$ singular values

# What else can we say?

Let $\mathbf{v}_1, \ldots, \mathbf{v}_n$ be an **orthogonal** eigenbasis of $\mathbb{R}^n$ for $A^T A$ and suppose $A$ has $r$ nonzero singular values

$$\text{rank}(A)$$

**Theorem.** $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ is an orthogonal basis of $\text{Col}(A)$

# What else can we say?

Let $\mathbf{v}_1, ..., \mathbf{v}_n$ be an **orthogonal** eigenbasis of $\mathbb{R}^n$ for $A^T A$ and suppose $A$ has $r$ <u>nonzero</u> singular values

**Theorem.** $A\mathbf{v}_1, ..., A\mathbf{v}_r$ is an orthogonal basis of $\text{Col}(A)$

**This is the most important theorem for SVD**

# Verifying it

Let's show $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ are orthogonal (and linearly independent):

$$\langle A v_i, A v_j \rangle = \vec{v}_i^T A^T A \vec{v}_j \qquad i \neq j$$

$$= \vec{v}_i^T \lambda_j \vec{v}_j$$

$$= \lambda_j \vec{v}_i^T \vec{v}_j$$

$$= \lambda_j \langle \vec{v}_i, \vec{v}_j \rangle = 0$$

# Verifying it

Let's show $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ span Col($A$):

$$y \in \text{Col}(A) \implies \text{there is } \vec{r} \in \mathbb{R}^n \text{ s.t. } \vec{y} = A\vec{r}$$

$$\text{and} \quad \vec{r} = \sum_{i}^{r} \alpha_i \vec{r}_i$$

$$\vec{y} = A\vec{r} = A\left(\sum_{i=1}^{r} \alpha_i \vec{v}_i\right) = \sum_{i=1}^{r} \alpha_i A\vec{r}_i \implies$$

$$y \in \text{span}\left\{A\vec{v}_1, \ldots, A\vec{v}_r\right\}$$

# Putting it all together



Multiplication by $A$

$\mathbf{v}_2$  $\mathbf{v}_1$

$A\mathbf{v}_2$  $A\mathbf{v}_1$

$(18, 6)$

$(3, -9)$

# Putting it all together



Multiplication by $A$
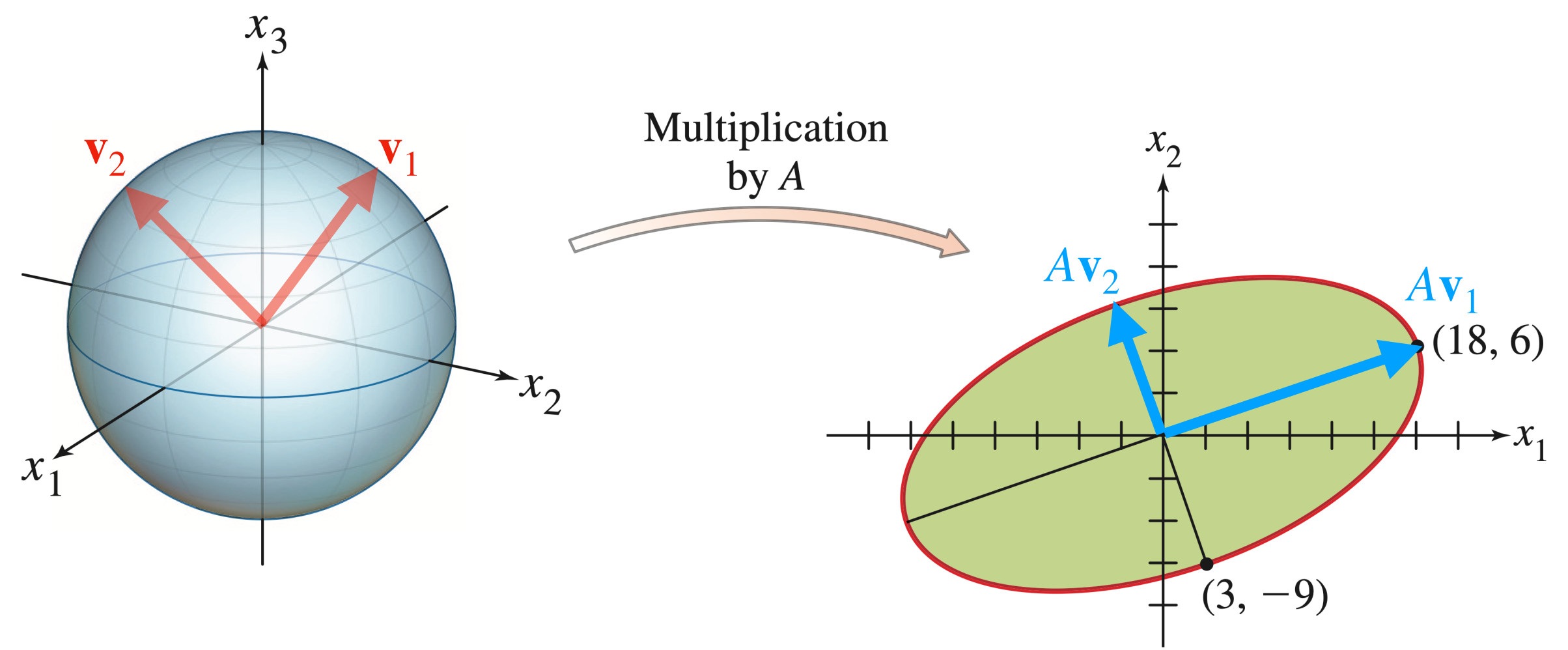
Let $A$ be an $m \times n$ matrix of rank $r$
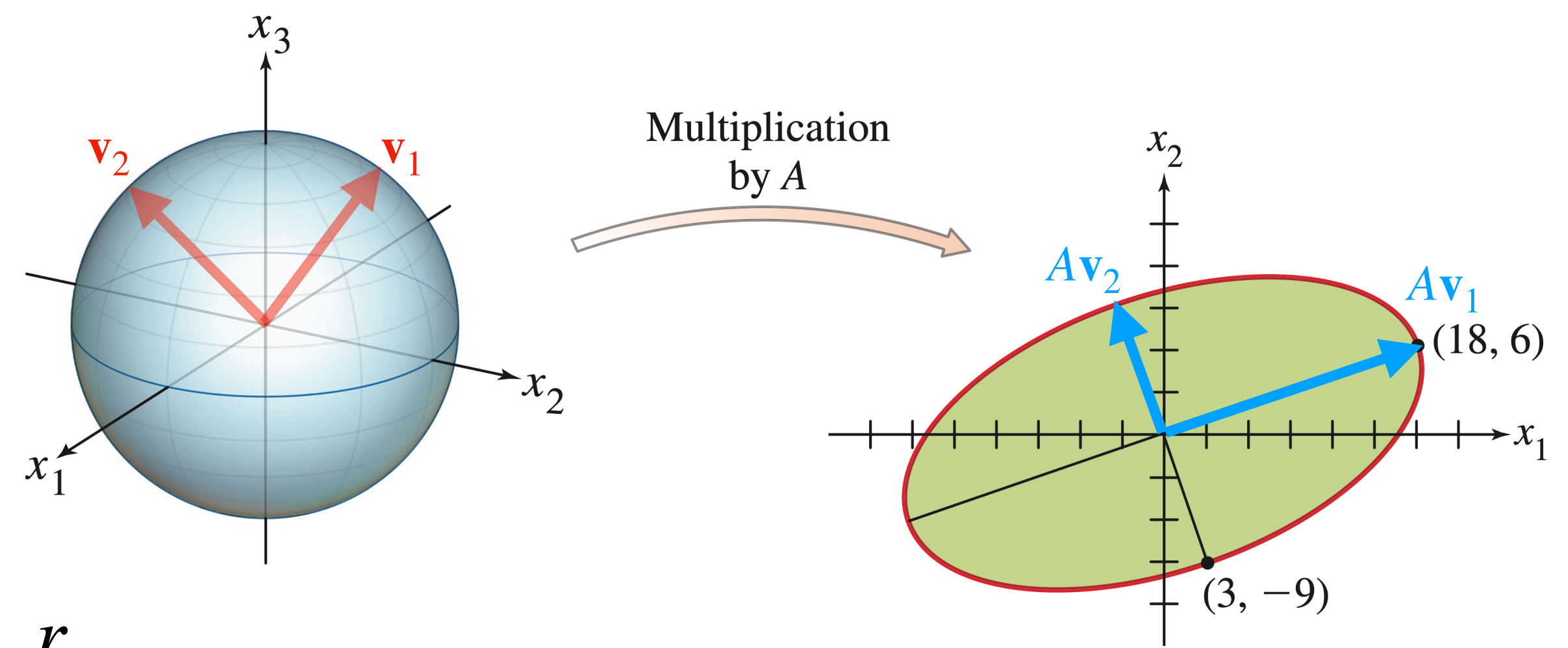
# Putting it all together
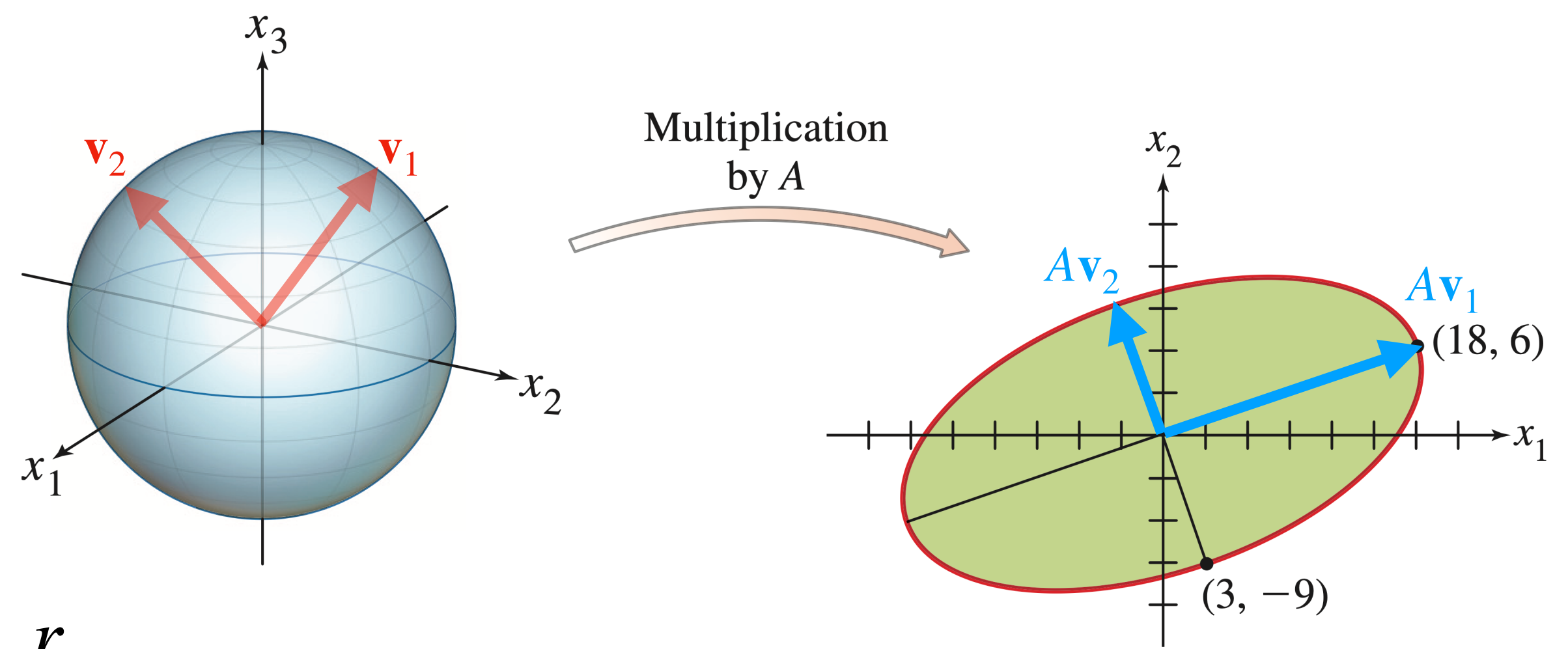


Let $A$ be an $m \times n$ matrix of rank $r$

What we know:

# Putting it all together
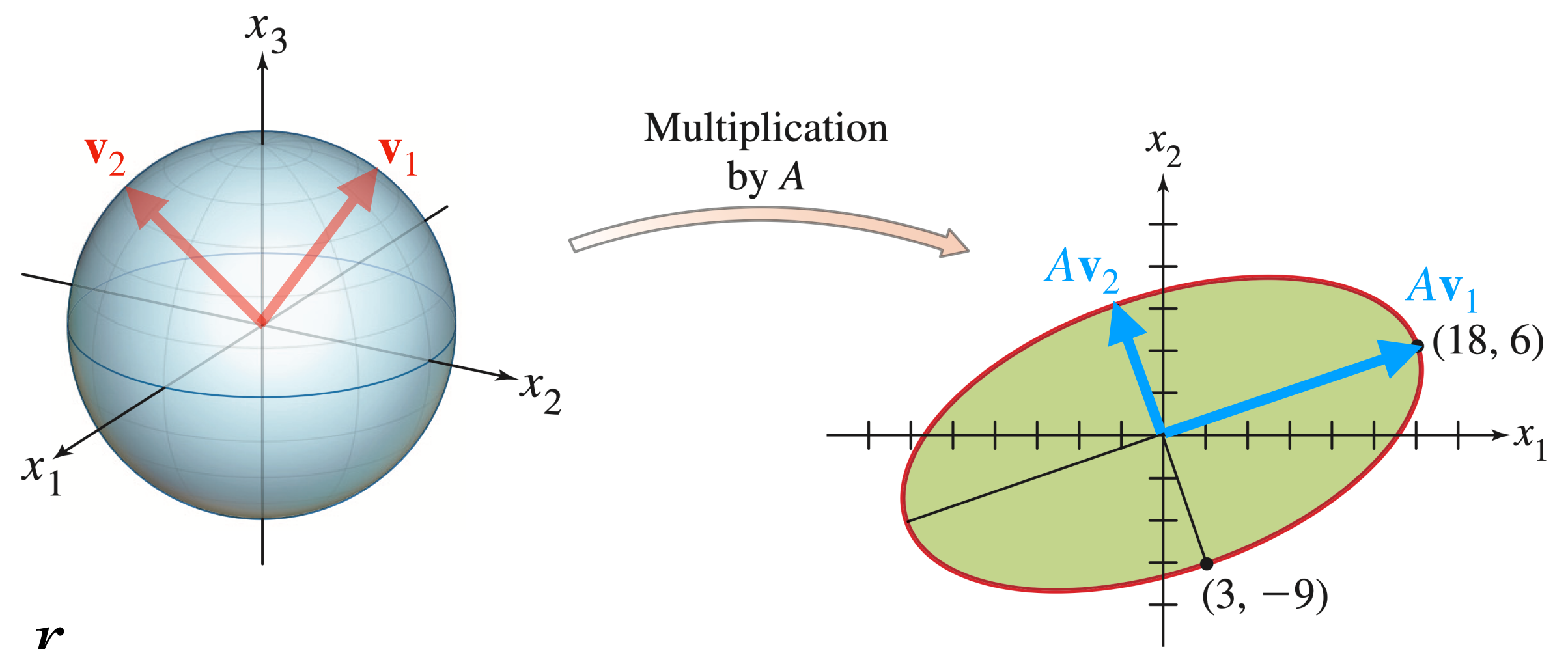


Multiplication by $A$

Let $A$ be an $m \times n$ matrix of rank $r$

## What we know:

» We can find orthonormal vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ in $\mathbb{R}^n$ such that $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ in $\mathbb{R}^m$ form an orthogonal basis for $\text{Col}(A)$

# Putting it all together
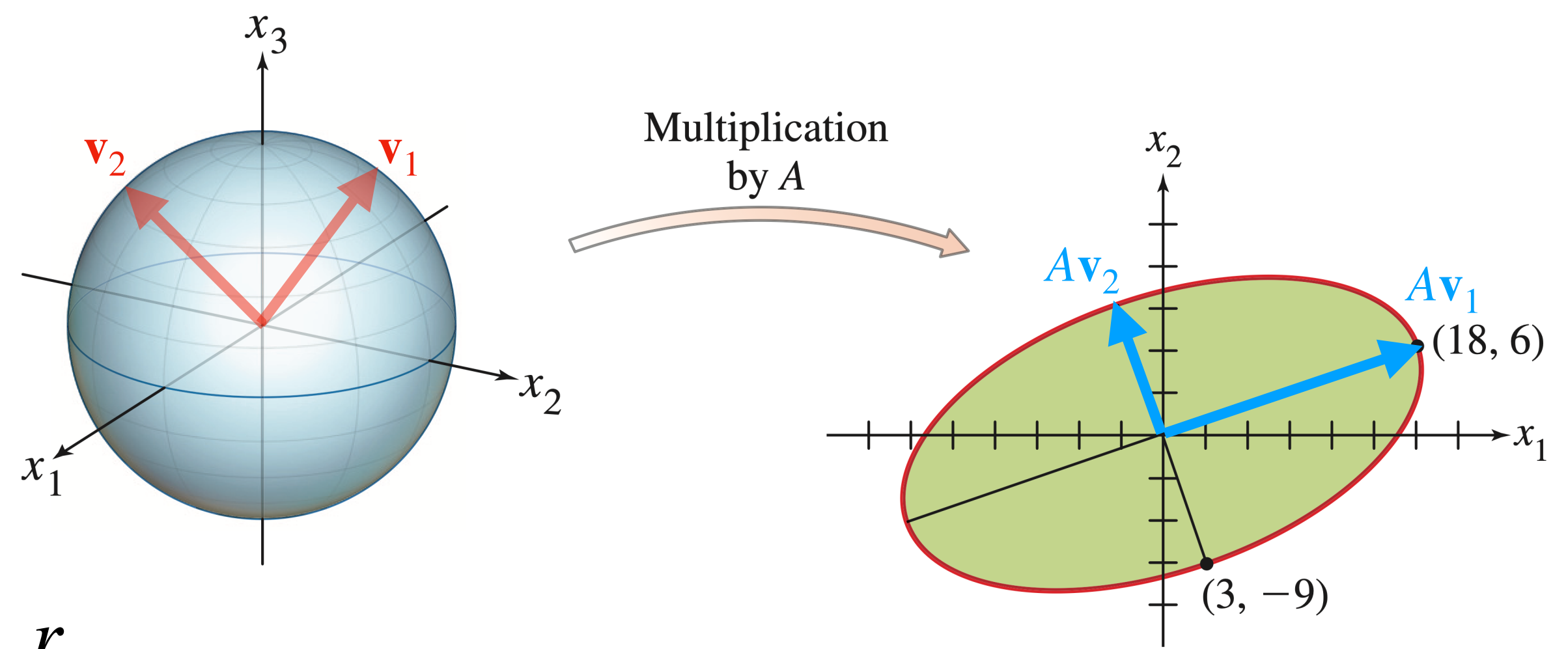
Let $A$ be an $m \times n$ matrix of rank $r$

What we know:

» We can find orthonormal vectors $\mathbf{v}_1, \dots, \mathbf{v}_n$ in $\mathbb{R}^n$ such that $A\mathbf{v}_1, \dots, A\mathbf{v}_r$ in $\mathbb{R}^m$ form an orthogonal basis for $\mathrm{Col}(A)$

» So if we take $\mathbf{u}_i = \dfrac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|}$, we get an **orthonormal** basis of $\mathrm{Col}(A)$

# Putting it all together
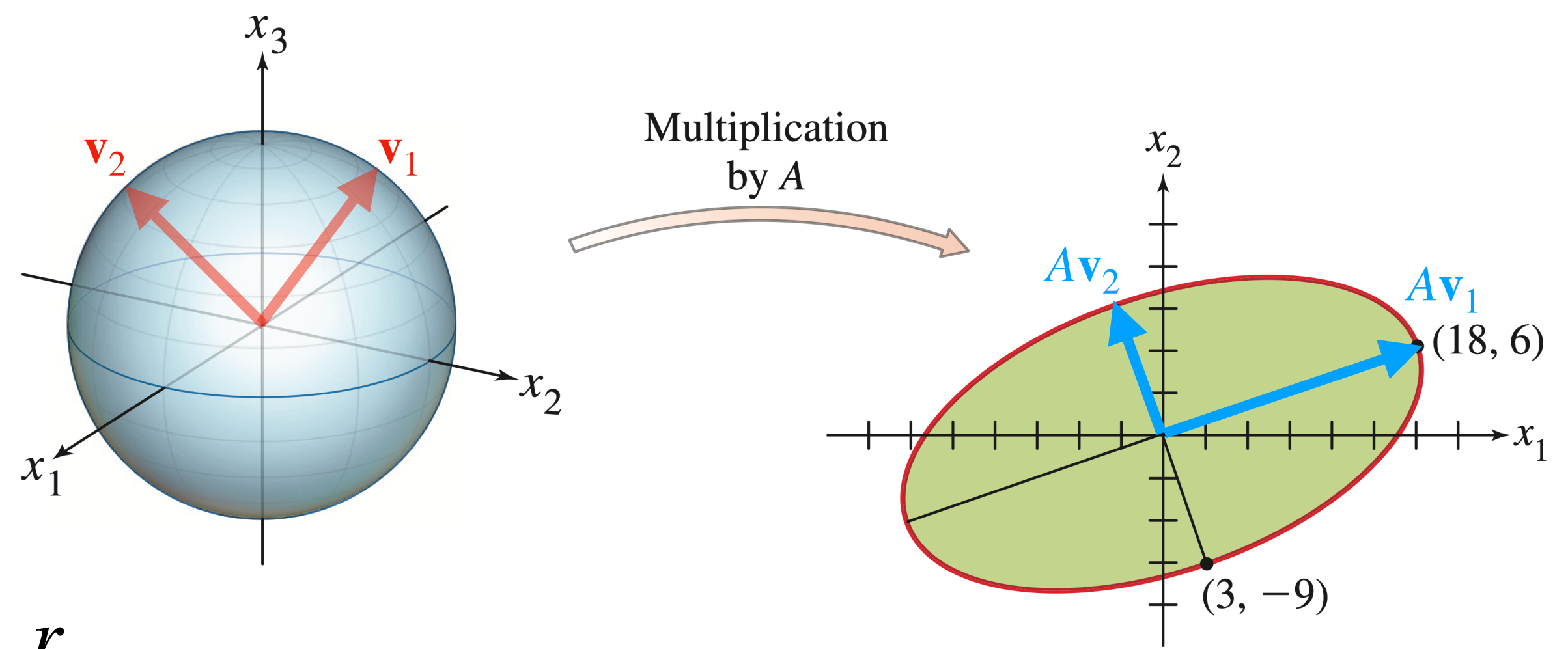


Multiplication by $A$

Let $A$ be an $m \times n$ matrix of rank $r$

What we know:

» We can find orthonormal vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ in $\mathbb{R}^n$ such that $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ in $\mathbb{R}^m$ form an orthogonal basis for $\text{Col}(A)$

» So if we take $\mathbf{u}_i = \dfrac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|}$, we get an **orthonormal** basis of $\text{Col}(A)$

» And we can extend this to $\mathbf{u}_1, \ldots, \mathbf{u}_m$ an orthonormal basis of $\mathbb{R}^m$ (via Gram–Schmidt).

# Singular Value Decomposition

# High Level View of the Decomposition



recall: Orthogonal
matrices preserves
lengths and angles

$M$

$V^T$

$U$

$\Sigma$

$\sigma_1$

$\sigma_2$
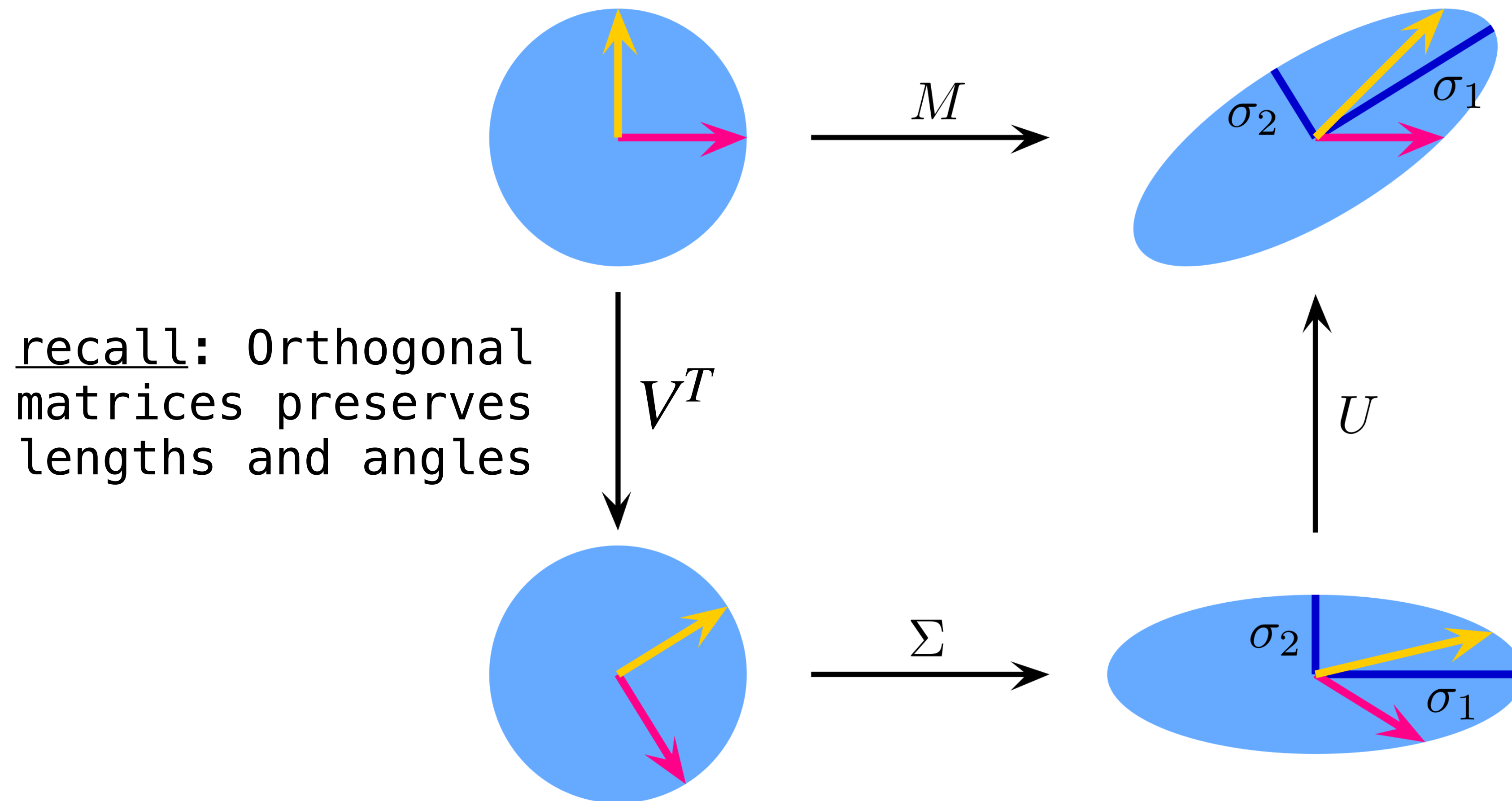
$$M = U \cdot \Sigma \cdot V^T$$

# The Important Equality

$$A\mathbf{v}_i = \|A\mathbf{v}_i\|\mathbf{u}_i = \sigma_i\mathbf{u}_i$$

$$\mathbf{u}_i = \frac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|}$$

# The Important Equality

$$\mathbf{u}_i = \frac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|}$$

$$A\mathbf{v}_i = \|A\mathbf{v}_i\|\mathbf{u}_i = \sigma_i\mathbf{u}_i$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$

# The Important Equality

$$\mathbf{u}_i = \frac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|}$$

$$A\mathbf{v}_i = \|A\mathbf{v}_i\|\mathbf{u}_i = \sigma_i\mathbf{u}_i$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$

What happens when we write this in matrix form?

# The Important Equality

$$A[\mathbf{v}_1 \ \ldots \ \mathbf{v}_n] = [\sigma_1 \mathbf{u}_1 \ \ldots \sigma_n \mathbf{u}_n]$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

# The Important Equality

$$A[\mathbf{v}_1 \ \ldots \ \mathbf{v}_n] = [\sigma_1\mathbf{u}_1 \ \ldots \sigma_n\mathbf{u}_n]$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \ldots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_m]$ and

# The Important Equality

$$A[\mathbf{v}_1 \ \ldots \ \mathbf{v}_n] = [\sigma_1 \mathbf{u}_1 \ \ldots \sigma_n \mathbf{u}_n]$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \ldots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_m]$ and

$$m > n$$
$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$$
or
$$m < n$$
$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_m & 0 & \ldots & 0 \end{bmatrix}$$
or
$$m = n$$
$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \end{bmatrix}$$

# The Important Equality

$$A[\mathbf{v}_1 \ \ldots \ \mathbf{v}_n] = [\sigma_1 \mathbf{u}_1 \ \ldots \sigma_n \mathbf{u}_n]$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \ldots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_m]$ and

remember: $U$ is orthonormal

$m > n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$$

$m < n$

$$\text{or } \Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_m & 0 & \ldots & 0 \end{bmatrix}$$

$m = n$

$$\text{or } \Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \end{bmatrix}$$

# The Important Equality

$$AV = U\Sigma$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \dots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \dots \ \mathbf{u}_m]$ and

remember: $U$ is orthonormal

$$m > n$$

$$\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \\ 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix}$$

$$m < n$$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_m & 0 & \dots & 0 \end{bmatrix}$

$$m = n$$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \end{bmatrix}$

# The Important Equality

$$\overset{m \times n}{A} \underset{n \times n}{V} = \overset{m \times m}{U} \underset{m \times n}{\Sigma}$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \dots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \dots \ \mathbf{u}_m]$ and

remember: $U$ is orthonormal

$m > n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \\ 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix}$$

$m < n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_m & 0 & \dots & 0 \end{bmatrix}$

$m = n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \end{bmatrix}$

# The Important Equality

$$A\cancel{VV}^T = U\Sigma V^T$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \ldots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_m]$ and

**remember: $U$ is orthonormal**

$m > n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$$

or

$m < n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_m & 0 & \ldots & 0 \end{bmatrix}$$

or

$m = n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \end{bmatrix}$$

# The Important Equality

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

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \; \ldots \; \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \; \ldots \; \mathbf{u}_m]$ and

remember: $U$ is orthonormal

$m > n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$$

$m < n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_m & 0 & \ldots & 0 \end{bmatrix}$

$m = n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_n \end{bmatrix}$

# The Important Equality

$$A = U\Sigma V^T$$

Remember that $\sigma_i = \sqrt{\lambda_i}$ is the singular value, which is the length $\|A\mathbf{v}_i\|$.

Let's take $V = [\mathbf{v}_1 \ \dots \ \mathbf{v}_n]$ and $U = [\mathbf{u}_1 \ \dots \ \mathbf{u}_m]$ and

remember: $U$ is orthonormal

$m > n$

$$\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \\ 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix}$$

$m < n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_m & 0 & \dots & 0 \end{bmatrix}$

$m = n$

or $\Sigma = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \end{bmatrix}$

# Singular Value Decomposition

**Theorem.** For a $m \times n$ matrix $A$, there are *orthogonal* matrices $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ such that

$$A = \underset{m \times n}{\underset{\color{blue}{m \times m}}{U}\, \Sigma\, \overset{\color{red}{n \times n}}{V^T}}$$

where diagonal entries* of $\Sigma$ are $\sigma_1, \ldots, \sigma_n$ the singular values of $A$.

* these are diagonal entries in a <u>non-square</u> matrix.

# Singular Value Decomposition

**Theorem.** For a $m \times n$ matrix $A$, there are *orthogonal* matrices $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ such that

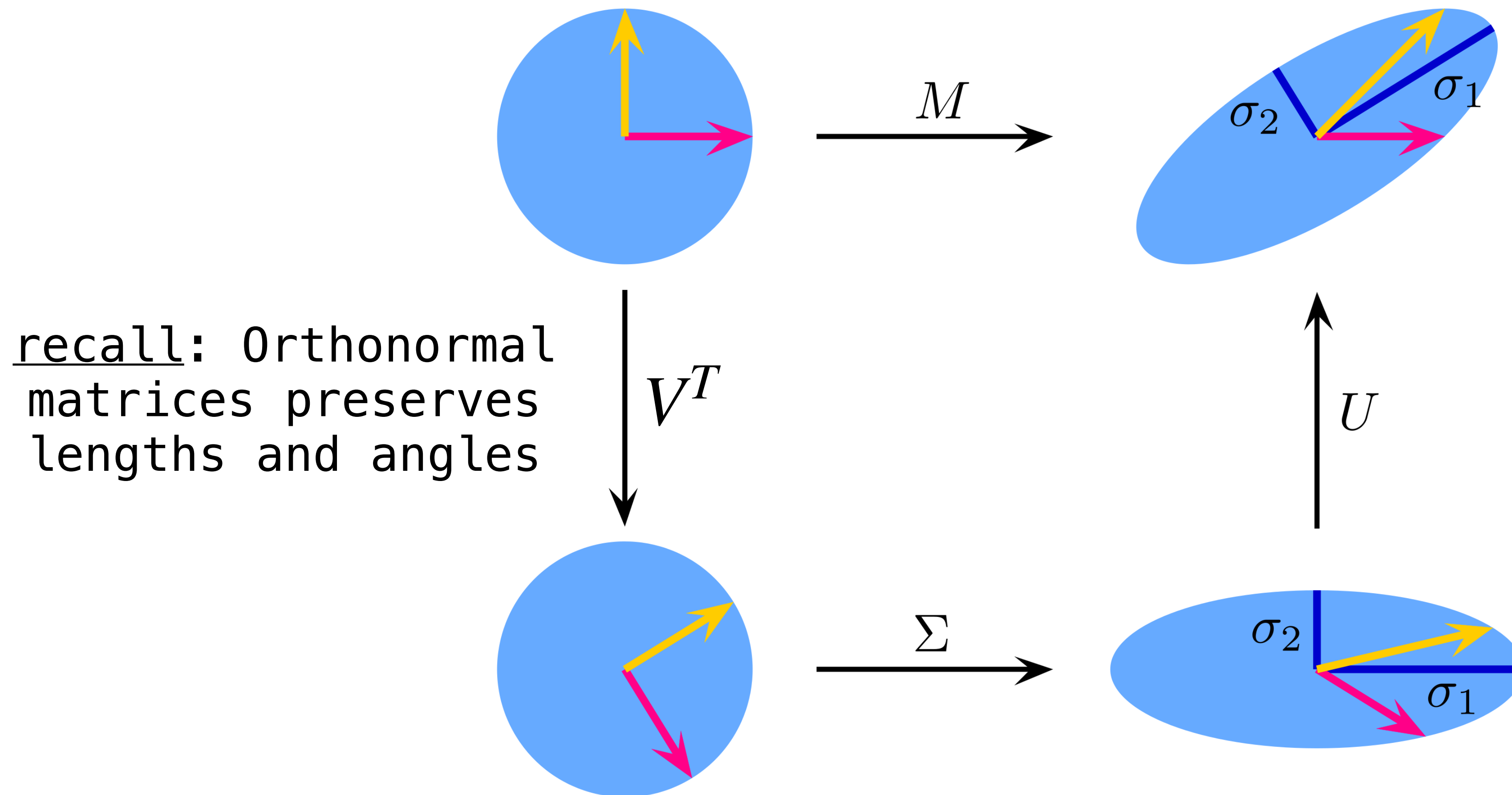<span style="color:#3399cc">left singular vectors</span>　　　　<span style="color:red">right singular vectors</span>

$$A = \overset{\color{blue}{m \times m}}{U} \underset{\color{purple}{m \times n}}{\Sigma} \overset{\color{red}{n \times n}}{V^T}$$

where diagonal entries* of $\Sigma$ are $\sigma_1, \ldots, \sigma_n$ the singular values of $A$.

\* these are diagonal entries in a <u>non-square</u> matrix.

# The Picture (Again)

recall: Orthonormal
matrices preserves
lengths and angles

$V^T$

$M$

$\Sigma$

$U$

$\sigma_1$

$\sigma_2$

$M = U \cdot \Sigma \cdot V^T$

# How To: Finding a SVD

# Step 1: Set up $\Sigma$

$$\Sigma = \begin{bmatrix} 3\sqrt{2} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

decreasing →

$$\begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix}$$

The **singular values** are the square roots of the eigenvalues of $A^T A$ (or $A A^T$):

$$A^T A = \begin{bmatrix} 1 & -2 & 2 \\ -1 & 2 & -2 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix} = \begin{bmatrix} 9 & -9 \\ -9 & 9 \end{bmatrix}$$

$$\det(A - \lambda I) = (\lambda - 9)^2 - 81 = \lambda^2 - 18\lambda = \lambda(\lambda - 18) \quad \lambda = 0, 18$$

# Step 2: Set up $V$

$$\begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix}$$

Find an orthonormal eigenbasis for $A^T A$:

$$A^T A = \begin{bmatrix} 9 & -9 \\ -9 & 9 \end{bmatrix} \qquad A - 18I \sim \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \quad v_1 = \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix}$$

$$A - 0I \sim \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix} \quad r_2 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

$$V = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

# Step 3: Set up $U$ (Part 1)

$$U = \begin{bmatrix} \frac{1}{3} & 0 & \frac{-4}{\sqrt{18}} \\ \frac{-2}{3} & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{18}} \\ \frac{2}{3} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{18}} \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix}$$

If $\mathbf{v}_1 \ldots, \mathbf{v}_n$ is an eigenbasis of $\mathbb{R}^n$ (in decreasing order of eigenvalue), then $A\mathbf{v}_1, \ldots, A\mathbf{v}_r$ is an eigenbasis of Col($A$) (where $r$ is the rank of $A$). These vectors can be normalized and made the first $r$ columns of $U$:

$$\vec{u}_1 = \frac{A v_1}{\|A v_1\|} = A \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}\left( A \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right) = \frac{1}{\sqrt{2}} \begin{bmatrix} 2 \\ -4 \\ 4 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix}$$

$$\langle \vec{w}, \vec{v} \rangle = 0 = x - 2y + 2z = 0$$

$$\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} -4 \\ -1 \\ 1 \end{bmatrix}$$

# Step 4: Set up $U$ (Part 2)

$$\begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix}$$

If $m > r$, then extend $\mathbf{u}_1, \ldots, \mathbf{u}_r$ until it has $m$ orthonormal vectors:

$$\left( \text{last slide} \right)$$

# Step 5: Put everything together

$$\begin{bmatrix} 1 & -1 \\ -2 & 2 \\ 2 & -2 \end{bmatrix}$$

$$A = U \Sigma V^T$$

# SVD in NumPy

In reality, we will almost never build SVDs by hand.  We can use:
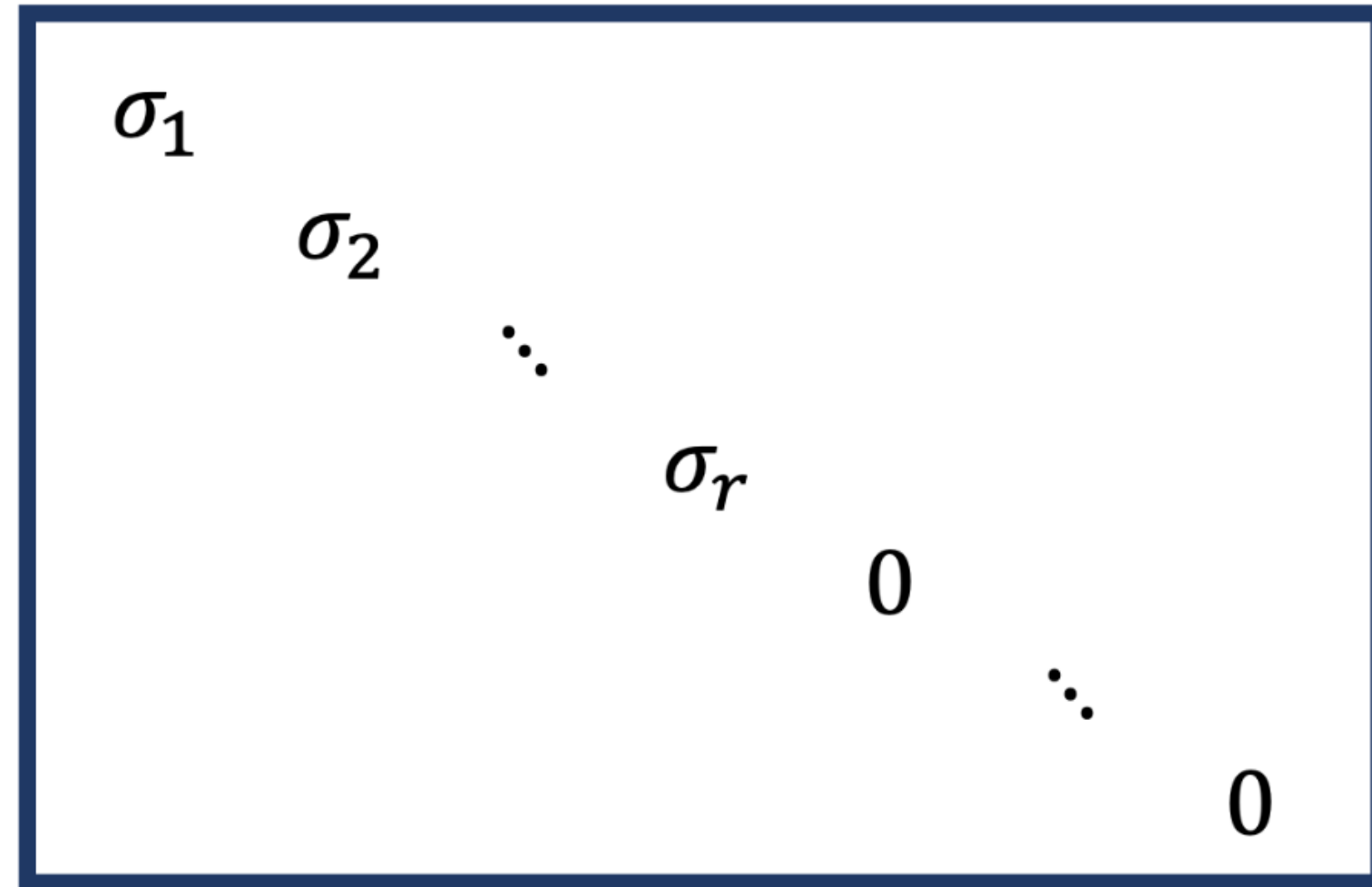
$$numpy.linalg.svd$$

Let's do a quick demo...

# Pseudoinverses

# SVD (The Picture)

$$U \qquad \Sigma \qquad V^T$$

$$\sigma_1$$
$$\sigma_2$$
$$\ddots$$
$$\sigma_r$$
$$0$$
$$\ddots$$
$$0$$

# Reduced SVD (The Picture)

$$U \qquad \Sigma \qquad V^T$$

$$\sigma_1$$
$$\sigma_2$$
$$\ddots$$
$$\sigma_r$$
$$0$$
$$\ddots$$
$$0$$

$$u_1 \quad u_2 \quad u_r$$

$$\cdots \quad v_1^T \quad \cdots$$
$$\cdots \quad v_2^T \quad \cdots$$
$$\vdots$$
$$\cdots \quad v_r^T \quad \cdots$$
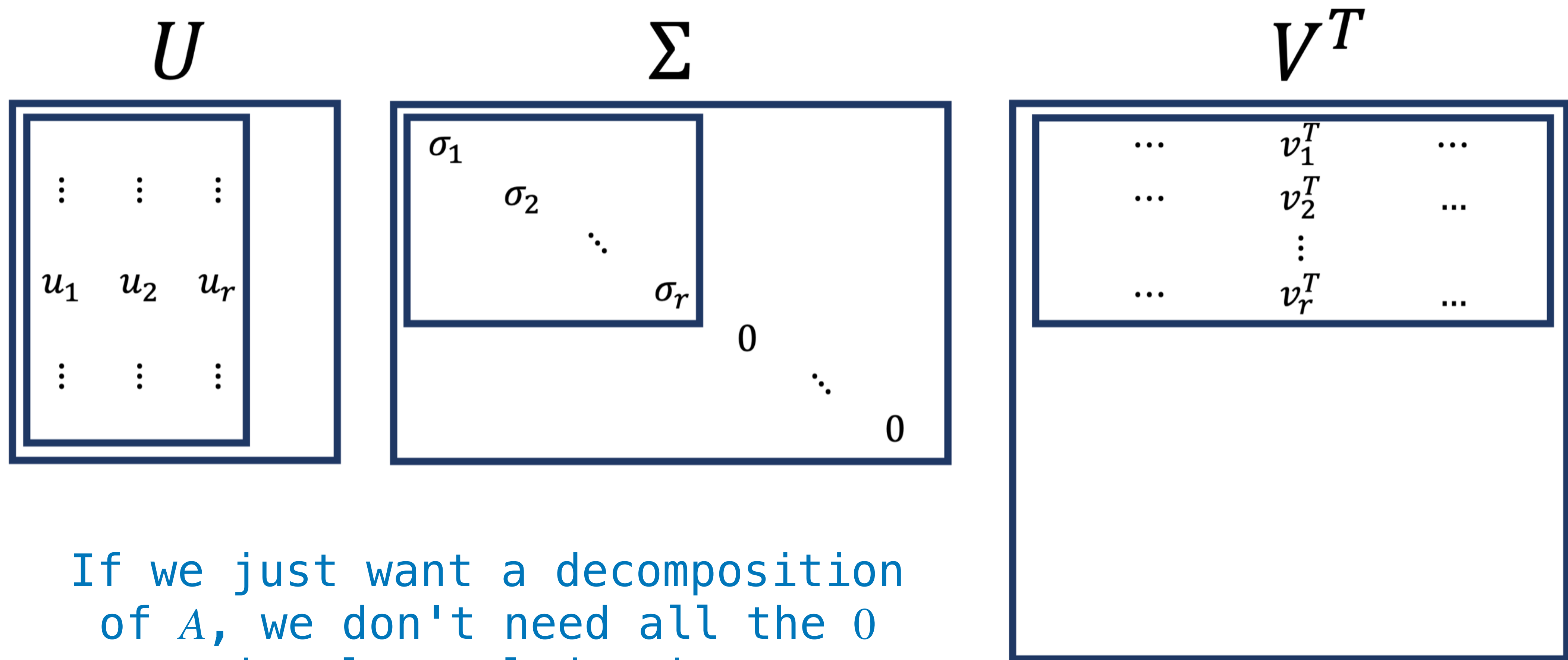
If we just want a decomposition of $A$, we don't need all the $0$ singular valudes in $\Sigma$

# The Reduced SVD

**Theorem.** For every matrix $A$ of rank $r$, there is an orthonormal matrix $U \in \mathbb{R}^{m \times r}$, a diagonal matrix $\Sigma \in \mathbb{R}^{r \times r}$ with **positive** entries on the diagonal, and an orthonormal matrix $V \in \mathbb{R}^{n \times r}$ such that

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

# The Pseudoinverse

**Definition.** Given a reduced SVD $A = U\Sigma V^T$, the **_pseudoinverse_** of $A$ is $A^+ = V\Sigma^{-1}U^T$

**Theorem.** $A^+\mathbf{b}$ is the _minimum length_ **least squares solution** of $A\mathbf{x} = \mathbf{b}$

$$AA^+b = U\Sigma V^T V \Sigma^{-1} U^T b$$

$$= U U^T \vec{b} = \text{proj}_{Col(A)}\vec{b} = U(U^T U)^{-1}U^T\vec{b}$$

*(in Python we have **numpy.linalg.pinv**)*

# Recall: Least Squares in NumPy

## numpy.linalg.lstsq

`linalg.`**`lstsq`**`(a, b, rcond='warn')`

Return the least-squares solution to a linear matrix equation.

Computes the vector *x* that approximately solves the equation `a @ x = b`. The equation may be under-, well-, or over-determined (i.e., the number of linearly independent rows of *a* can be less than, equal to, or greater than its number of linearly independent columns). If *a* is square and of full rank, then *x* (but for round-off error) is the "exact" solution of the equation. Else, *x* minimizes the Euclidean 2-norm $||b - ax||$. If there are multiple minimizing solutions, the one with the smallest 2-norm $||x||$ is returned.

**Parameters:**  **a** : *(M, N) array_like*

"Coefficient" matrix.

**b** : *{(M,), (M, K)} array_like*

Ordinate or "dependent variable" values. If *b* is two-dimensional, the least-squares solution is calculated for each of the *K* columns of *b*.

**rcond** : *float, optional*

# Recall: Least Squares in NumPy

## numpy.linalg.lstsq

linalg.**lstsq**(*a*, *b*, *rcond='warn'*)                                    **[source]**

Return the least-squares solution to a linear matrix equation.

Computes the vector *x* that approximately solves the equation a @ x = b. The equation may be under-, well-, or over-determined (i.e., the number of linearly independent rows of *a* can be less than, equal to, or greater than its number of linearly independent columns). If *a* is square and of full rank, then *x* (but for round-off error) is the "exact" solution of the equation. Else, *x* minimizes the Euclidean 2-norm $||b - ax||$. If there are multiple minimizing solutions, the one with the smallest 2-norm $||x||$ is returned.

NumPy chooses the shortest vector

**Parameters:**  **a** : *(M, N) array_like*

"Coefficient" matrix.

**b** : *{(M,), (M, K)} array_like*

Ordinate or "dependent variable" values. If *b* is two-dimensional, the least-squares solution is calculated for each of the *K* columns of *b*.
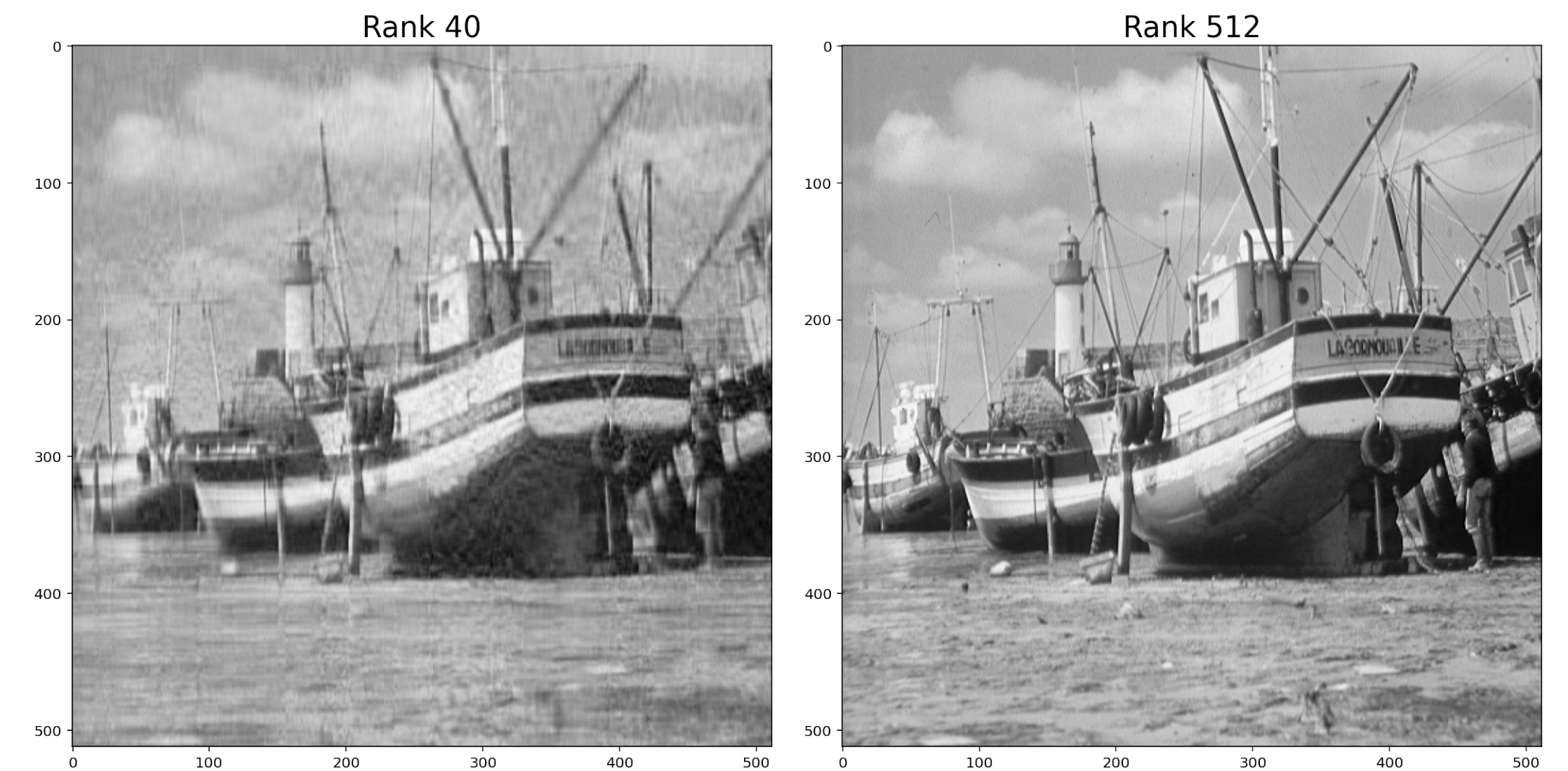
**rcond** : *float, optional*

# Recall: Least Squares in NumPy

## numpy.linalg.lstsq

linalg.**lstsq**(*a*, *b*, *rcond='warn'*)                                    [source]

> Return the least-squares solution to a linear matrix equation.
>
> Computes the vector *x* that approximately solves the equation `a @ x = b`. The equation may be under-, well-, or over-determined (i.e., the number of linearly independent rows of *a* can be less than, equal to, or greater than its number of linearly independent columns). If *a* is square and of full rank, then *x* (but for round-off error) is the "exact" solution of the equation. Else, *x* minimizes the Euclidean 2-norm $||b - ax||$. If there are multiple minimizing solutions, the one with the smallest 2-norm $||x||$ is returned.

NumPy chooses the shortest vector

(why?...)

> **Parameters:**   **a** : *(M, N) array_like*
>
>     "Coefficient" matrix.
>
> **b** : *{(M,), (M, K)} array_like*
>
>     Ordinate or "dependent variable" values. If *b* is two-dimensional, the least-squares solution is calculated for each of the K columns of *b*.
>
> **rcond** : *float, optional*

# Recall: Least Squares in NumPy

## numpy.linalg.lstsq

`linalg.`**`lstsq`**`(a, b, rcond='warn')`

Return the least-squares solution to a linear matrix equation.

Computes the vector *x* that approximately solves the equation `a @ x = b`. The equation may be under-, well-, or over-determined (i.e., the number of linearly independent rows of *a* can be less than, equal to, or greater than its number of linearly independent columns). If *a* is square and of full rank, then *x* (but for round-off error) is the "exact" solution of the equation. Else, *x* minimizes the Euclidean 2-norm $||b - ax||$. If there are multiple minimizing solutions, the one with the smallest 2-norm $||x||$ is returned.

NumPy chooses the shortest vector

(why?**...**)

**because they use SVD!**

Parameters: a : *(M, N) array_like*

"Coefficient" matrix.

b : *{(M,), (M, K)} array_like*

Ordinate or "dependent variable" values. If *b* is two-dimensional, the least-squares solution is calculated for each of the *K* columns of *b*.

rcond : *float, optional*

# What's next?
# A couple final thoughts

# Applications of SVD

image compression

Rank 40

Rank 512

2D PCA Visualization Labeled with Document Source

- comp.os.ms-windows.misc
- rec.sport.baseball
- sci.space

document
classification

# Applications of SVD

- Reduced SVD, pseudoinverses and least squares

Rank 40

Rank 512



2D PCA Visualization Labeled with Document Source

comp.os.ms-windows.misc
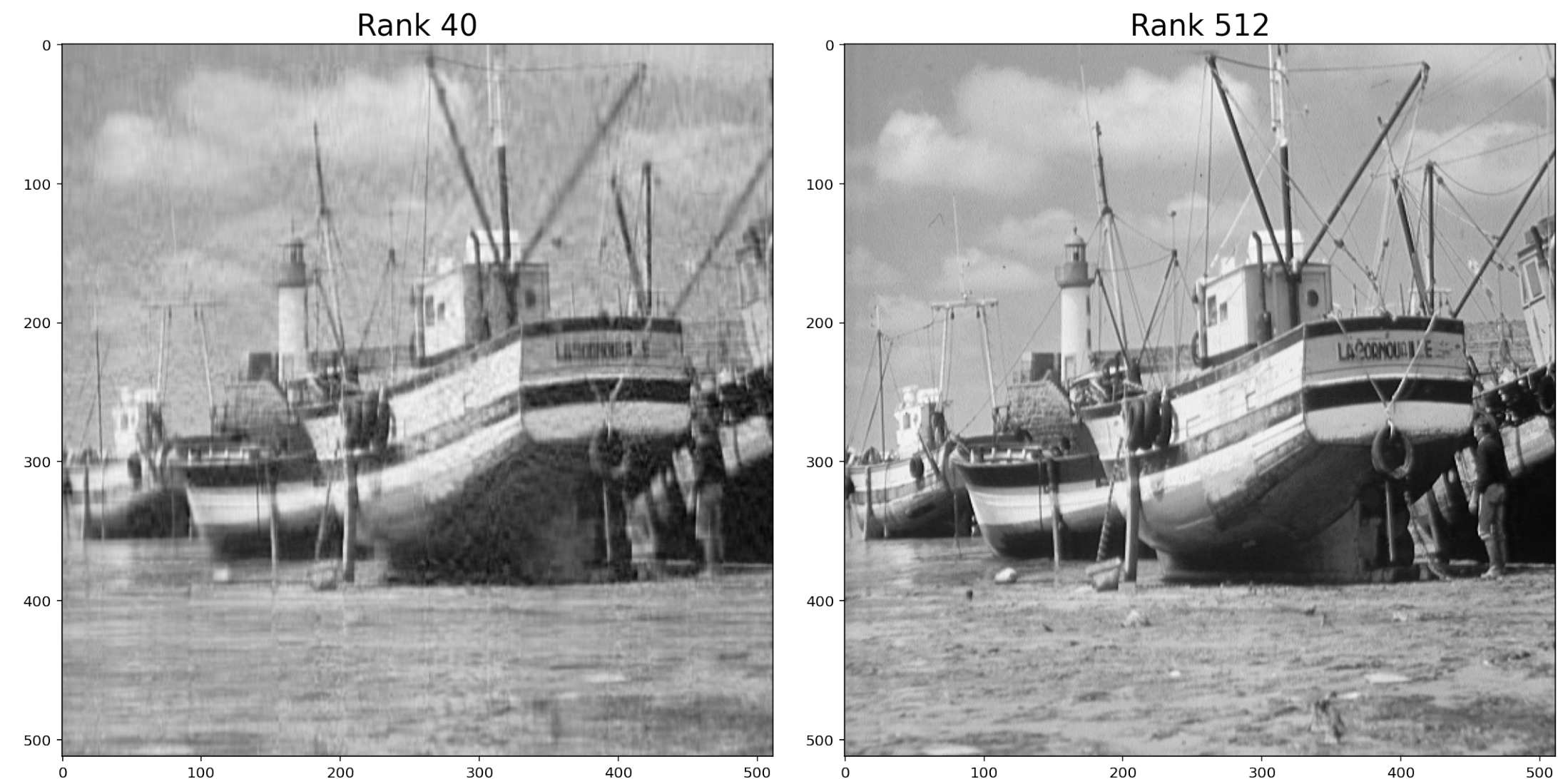rec.sport.baseball
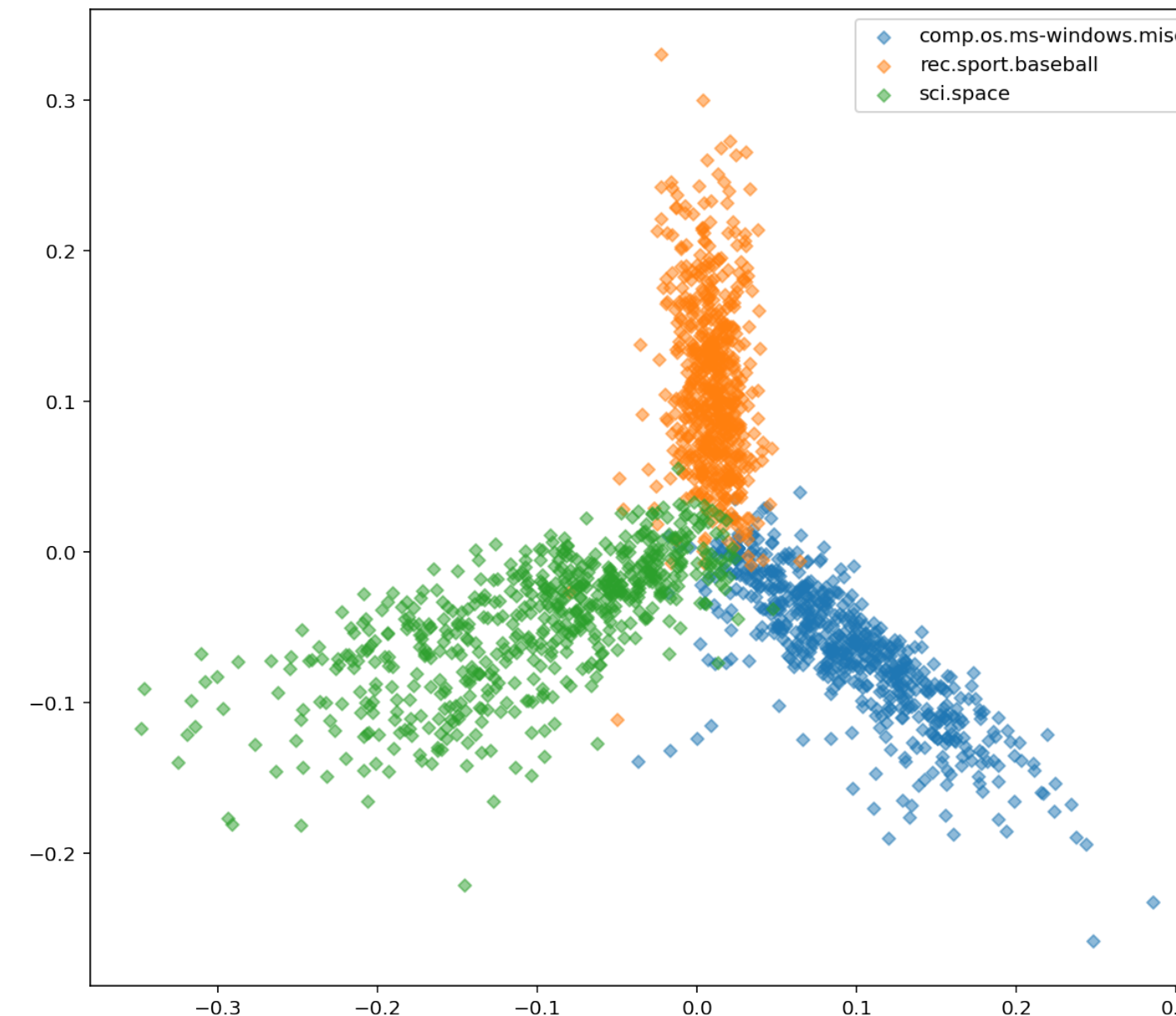sci.space

document classification

# Applications of SVD

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

Rank 40 Rank 512

2D PCA Visualization Labeled with Document Source

comp.os.ms-windows.misc
rec.sport.baseball
sci.space

document classification

# Applications of SVD

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length
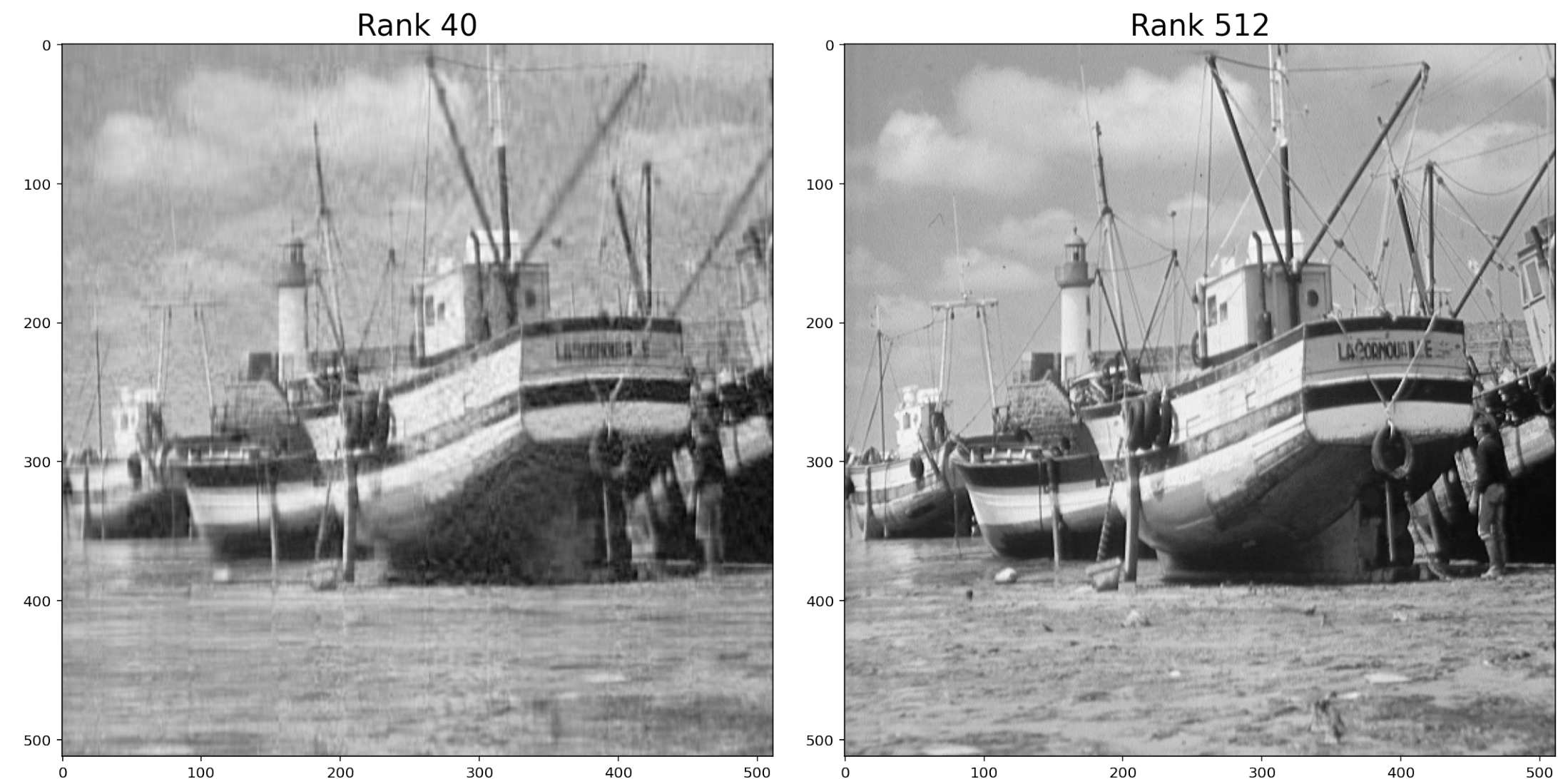
- Low Rank Approximation and Data Compression
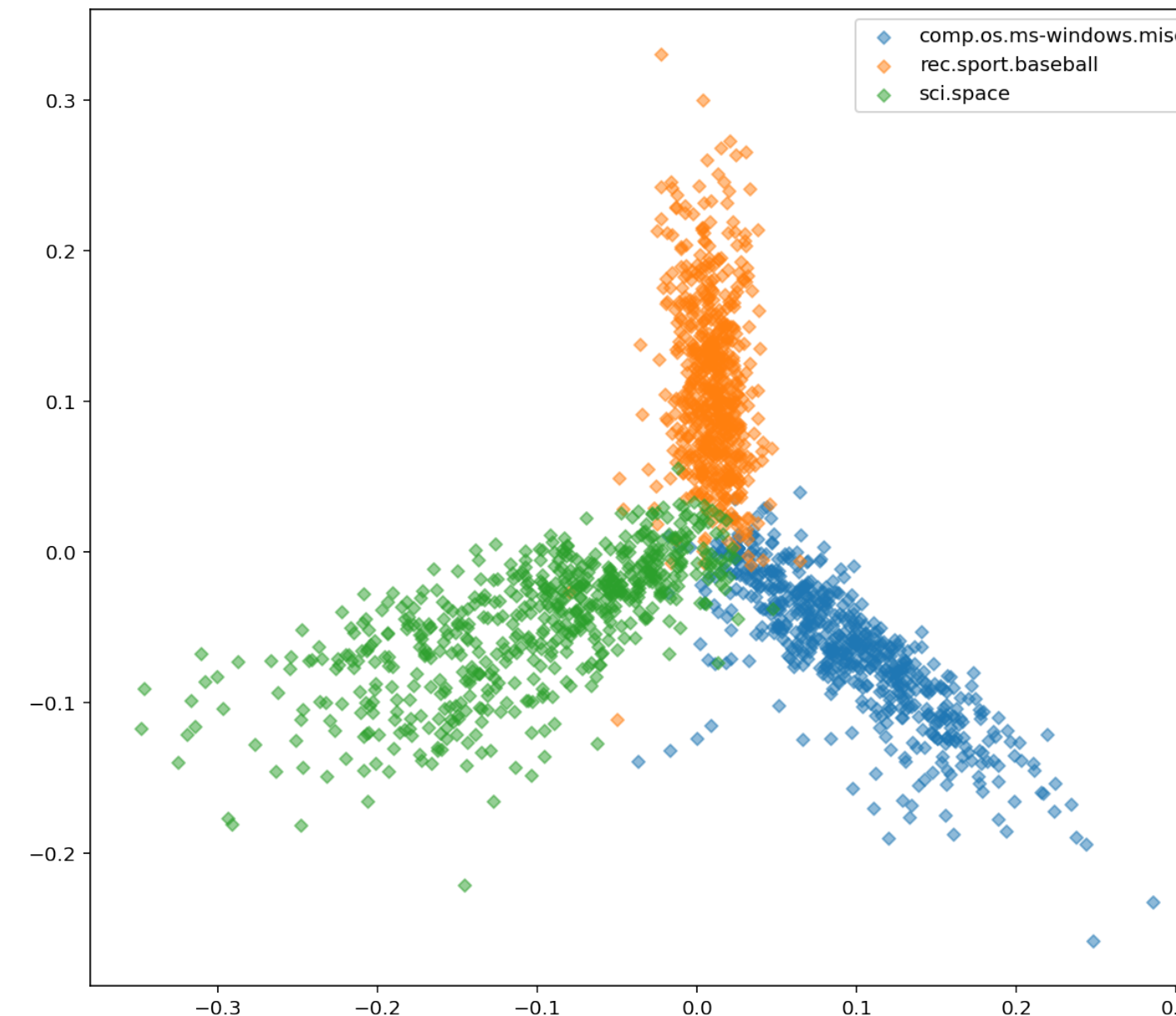
document classification

# Applications of SVD

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

- Low Rank Approximation and Data Compression

  - Replacing small singular values with zero in $\Sigma$ gives a good approximation to $A$.
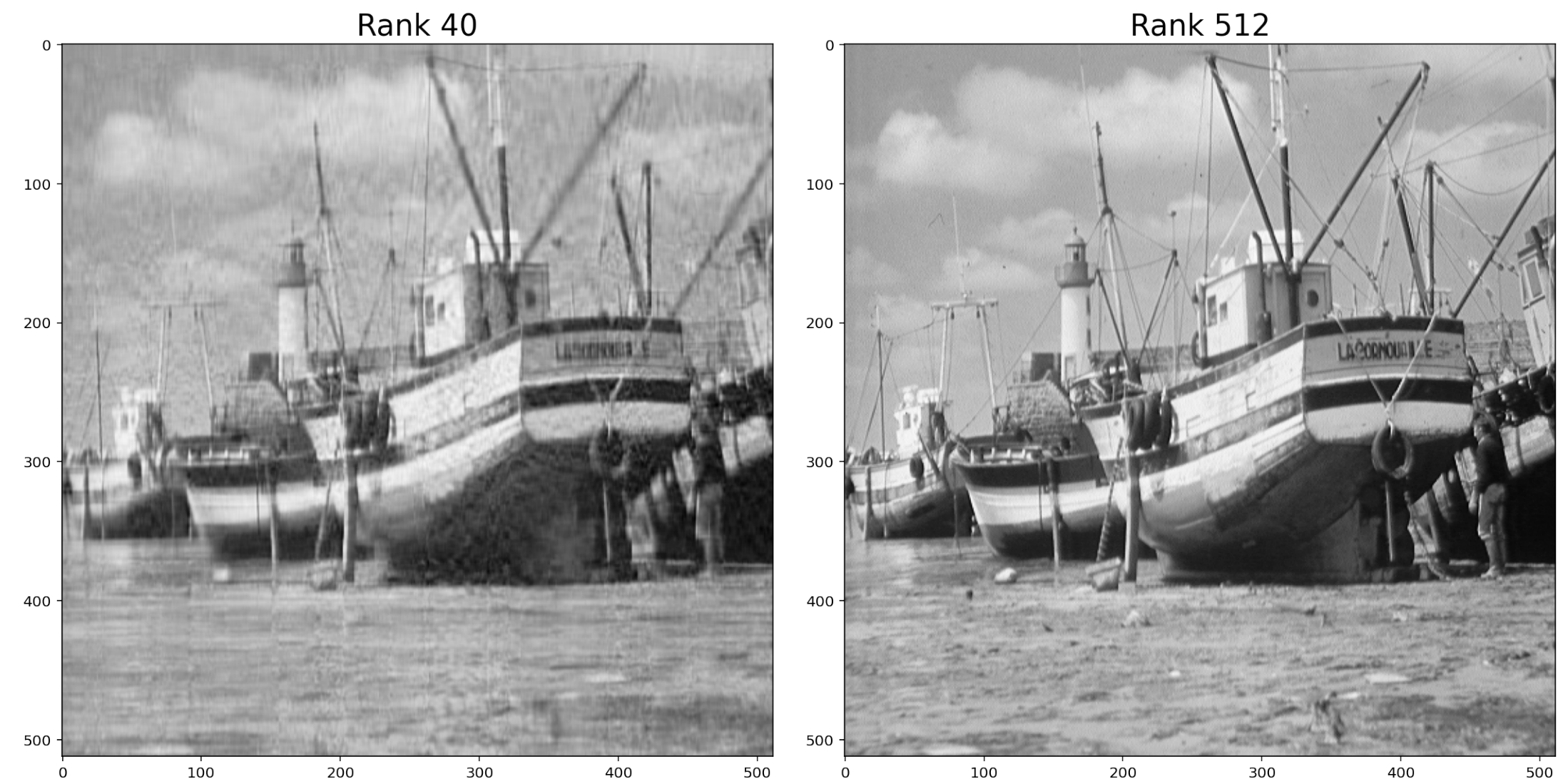
image compression



2D PCA Visualization Labeled with Document Source



document classification

# Applications of SVD

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

- Low Rank Approximation and Data Compression

  - Replacing small singular values with zero in $\Sigma$ gives a good approximation to $A$.

  - This is used for image compression



Rank 40      Rank 512



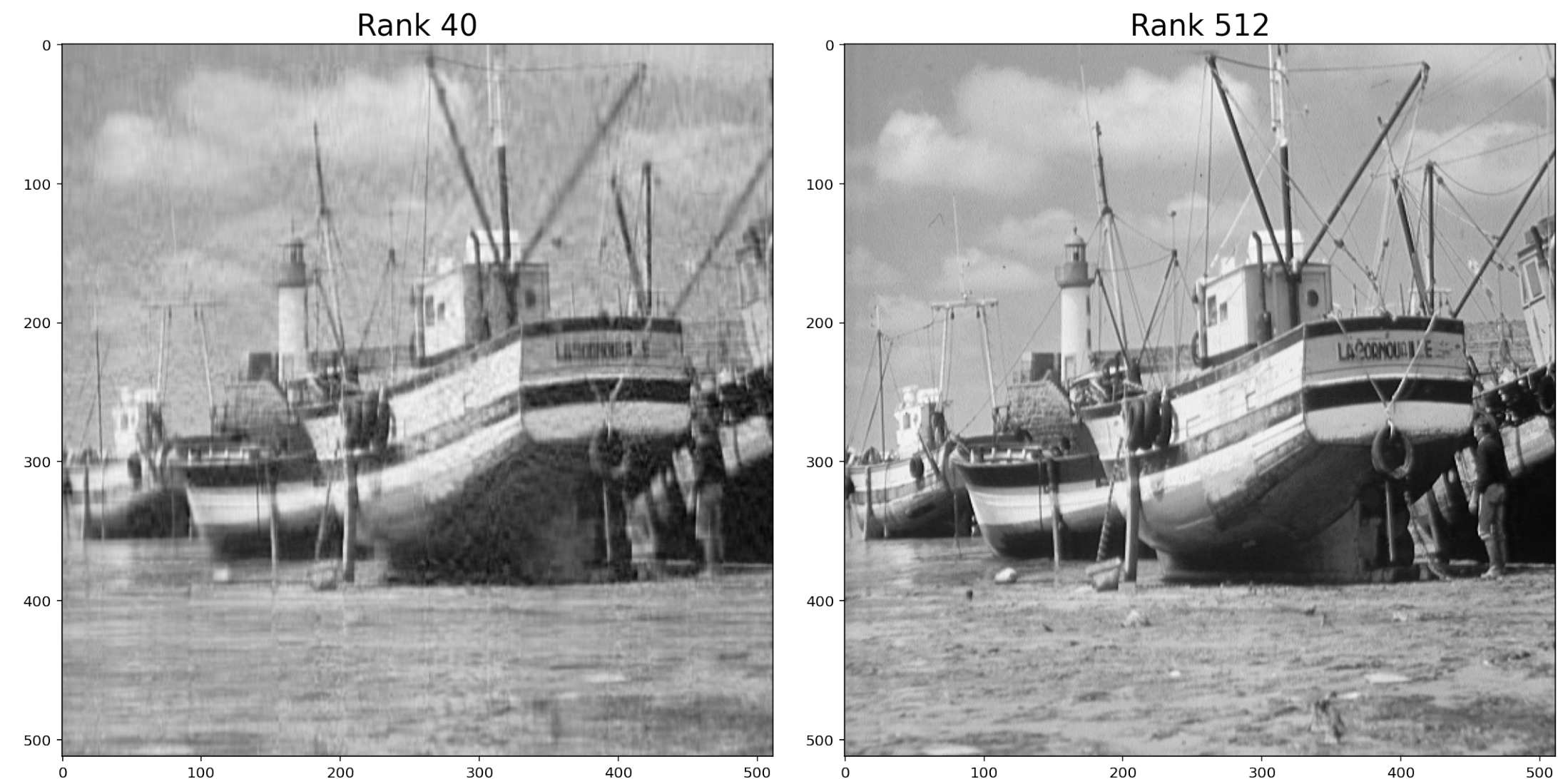2D PCA Visualization Labeled with Document Source
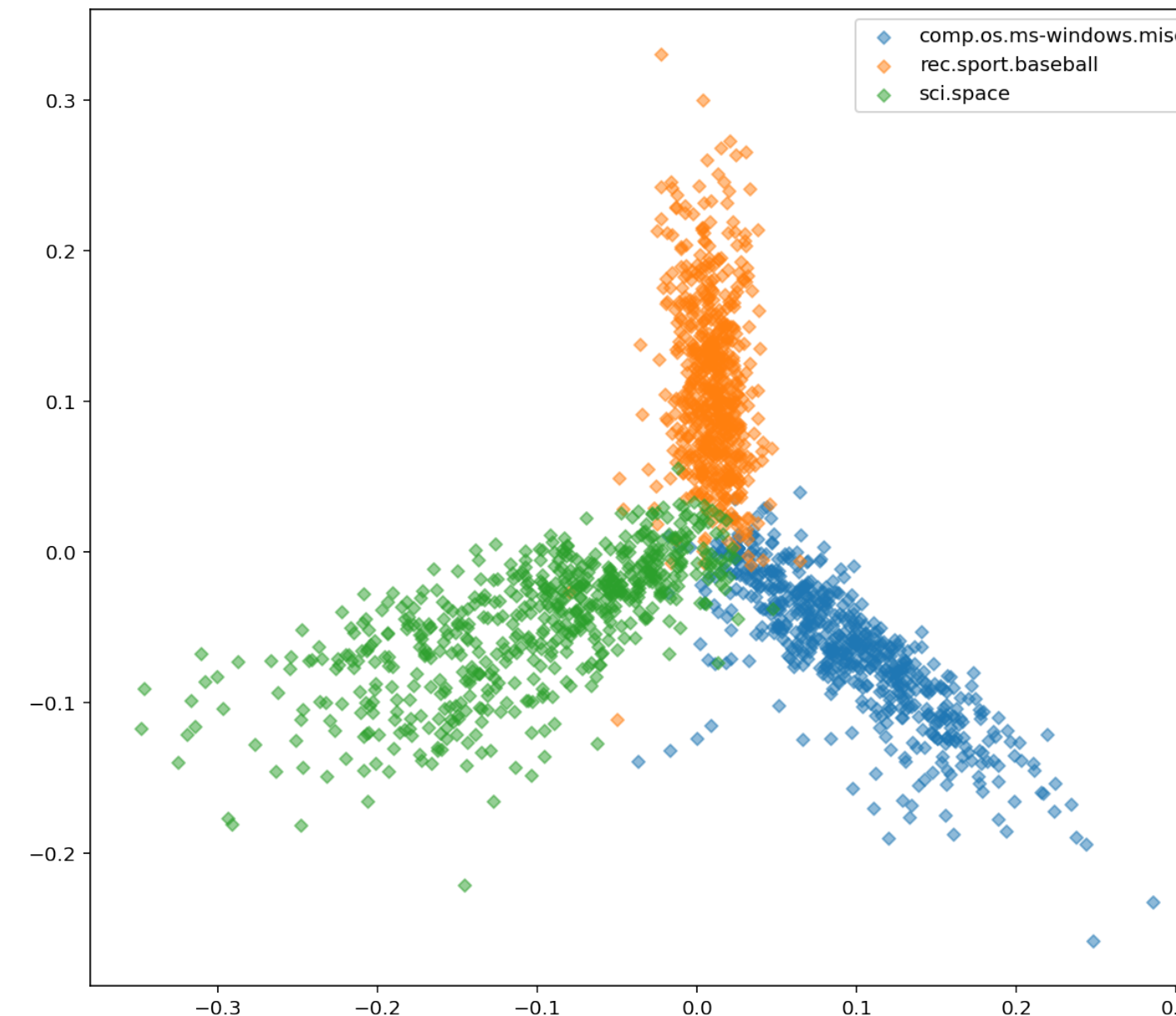
document classification

# **Applications of SVD**

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

- Low Rank Approximation and Data Compression

  - Replacing small singular values with zero in $\Sigma$ gives a good approximation to $A$.

  - This is used for image compression

- Principle Component Analysis

image compression



Rank 40      Rank 512



2D PCA Visualization Labeled with Document Source

- comp.os.ms-windows.misc
- rec.sport.baseball
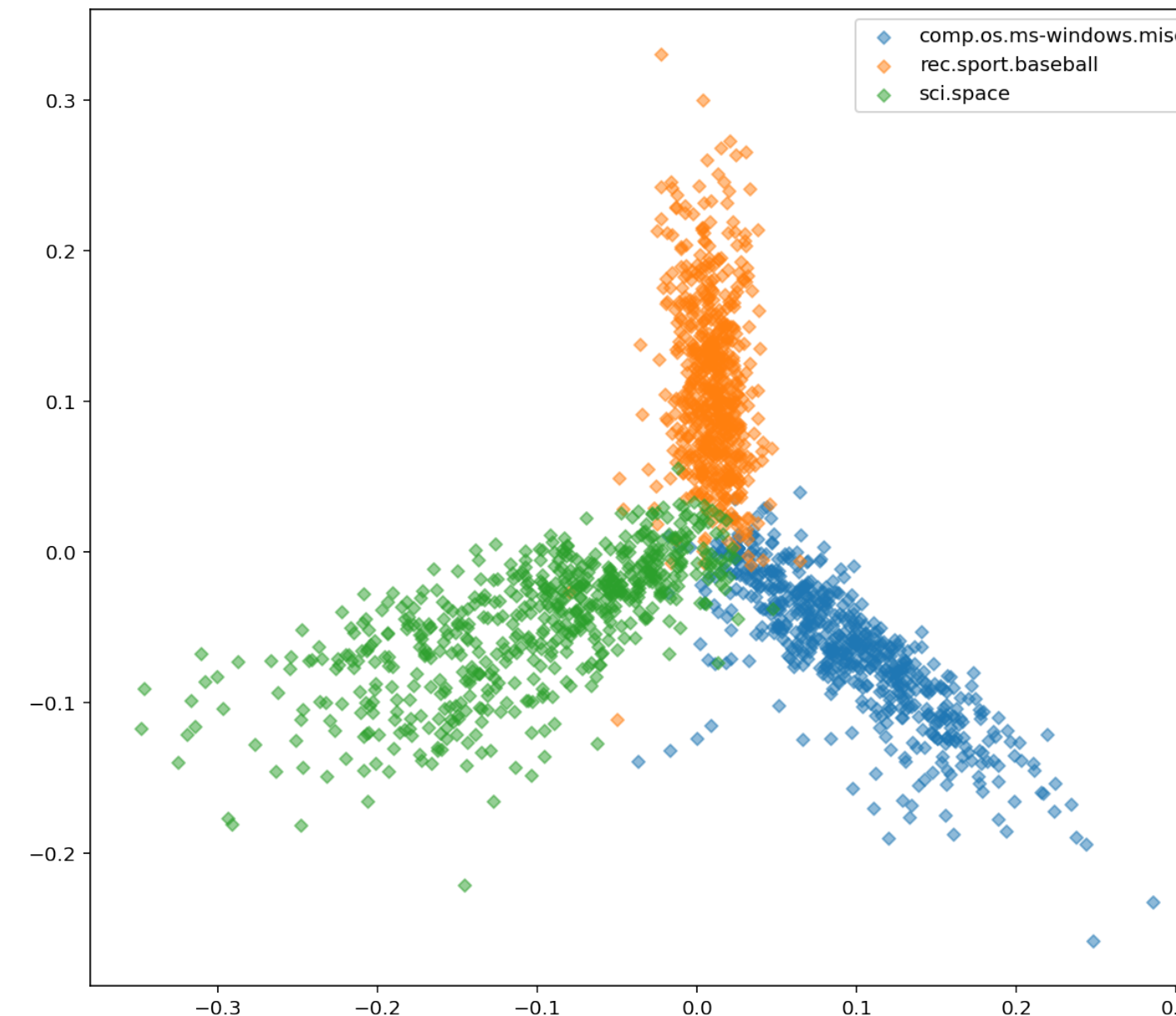- sci.space

document classification

# **Applications of SVD**

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

- Low Rank Approximation and Data Compression

  - Replacing small singular values with zero in $\Sigma$ gives a good approximation to $A$.

  - This is used for image compression

- Principle Component Analysis
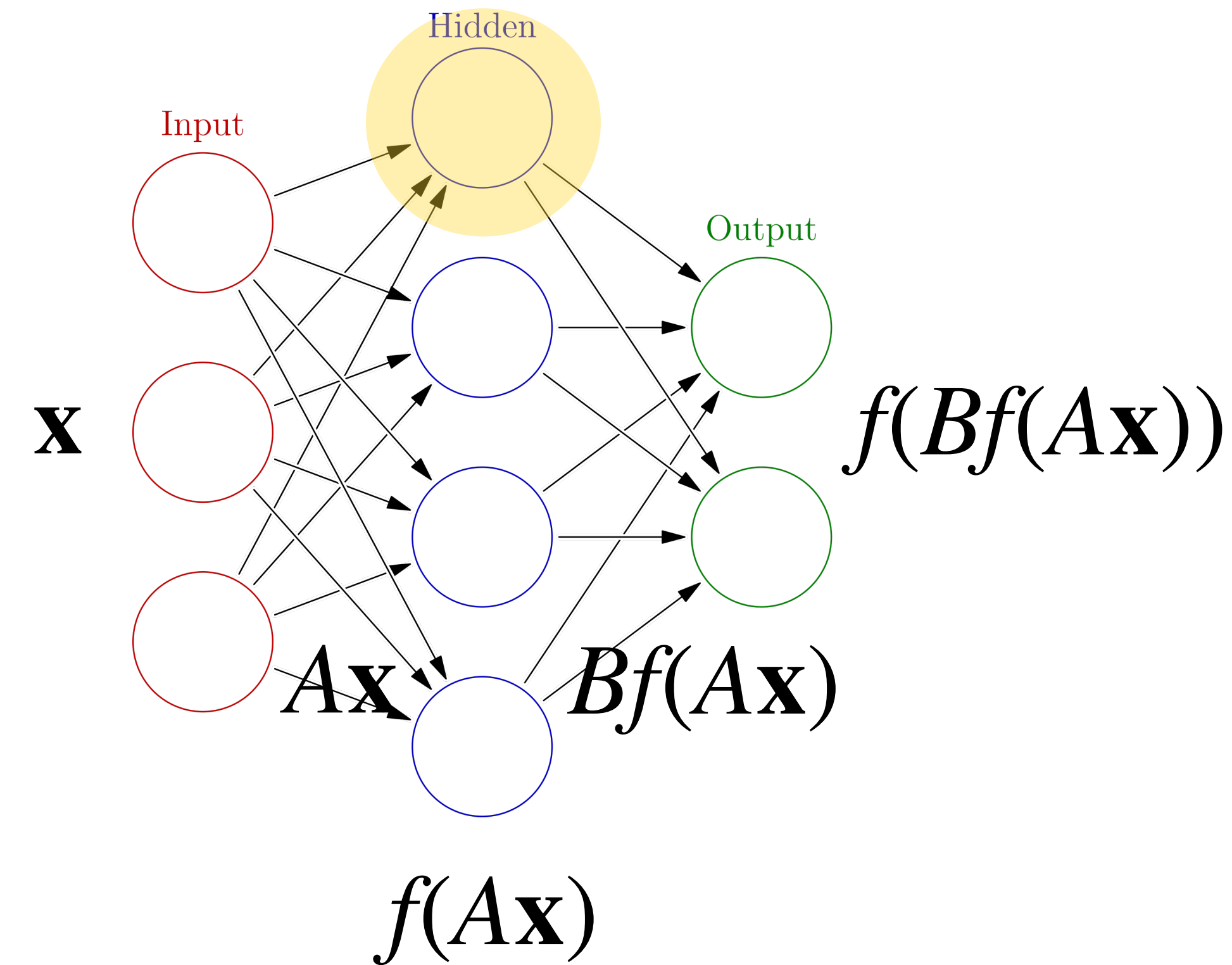
  - Large singular vectors are "most affected."



Rank 40    Rank 512



2D PCA Visualization Labeled with Document Source

comp.os.ms-windows.misc
rec.sport.baseball
sci.space

document classification

# Applications of SVD

image compression



Rank 40            Rank 512

- Reduced SVD, pseudoinverses and least squares

  - If $A^+ = V\Sigma^{-1}U^T$, then $A^+\mathbf{b}$ is a least squares solution of minimum length

- Low Rank Approximation and Data Compression

  - Replacing small singular values with zero in $\Sigma$ gives a good approximation to $A$.

  - This is used for image compression

- Principle Component Analysis

  - Large singular vectors are "most affected."

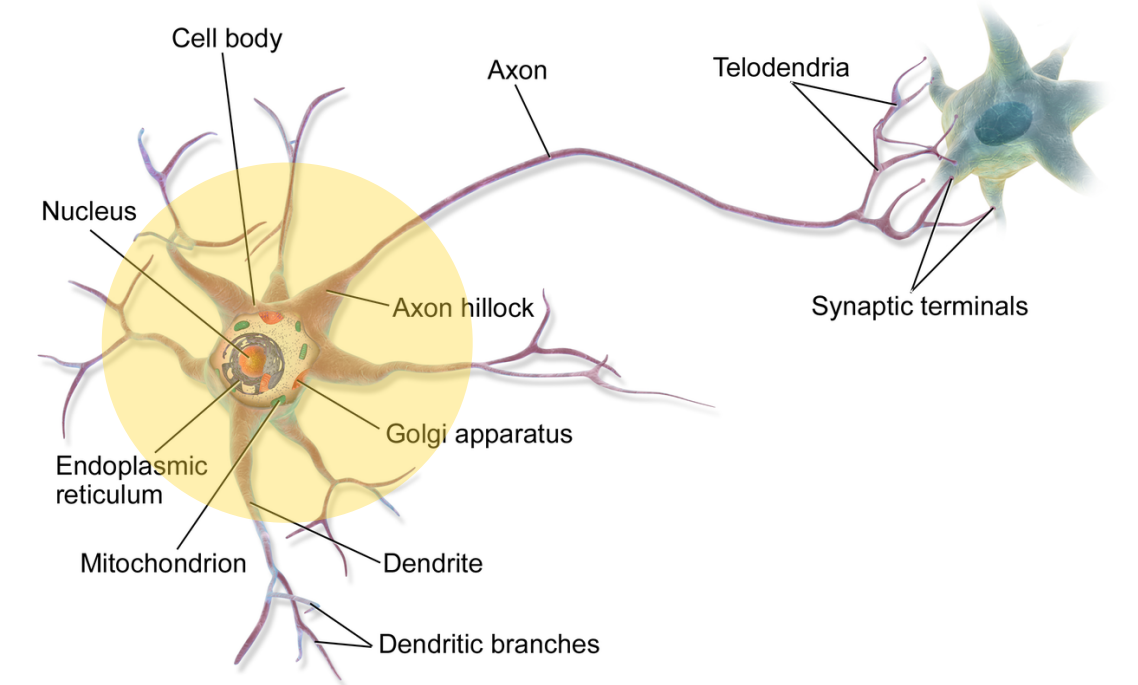  - These are good vectors to look at for classifying data
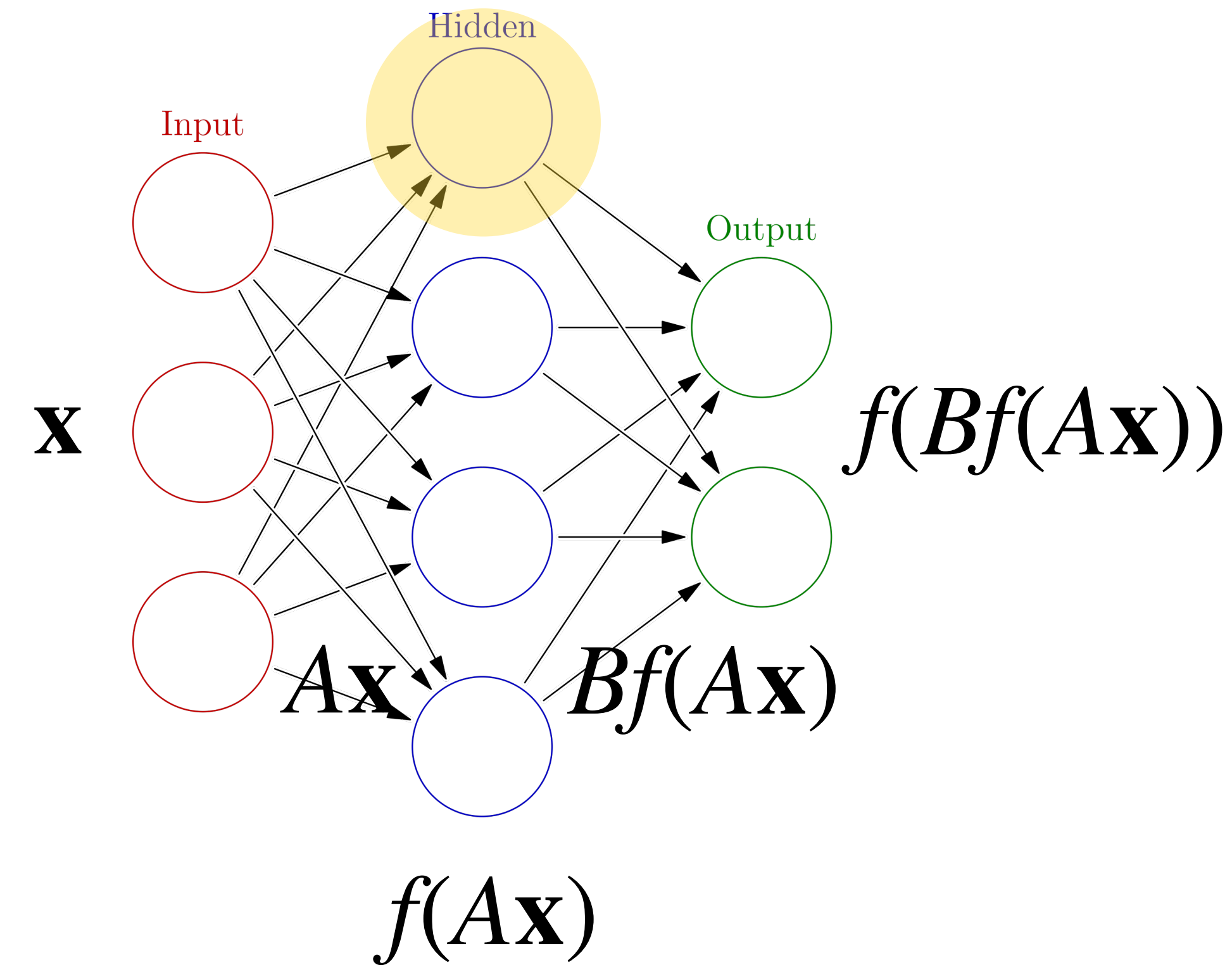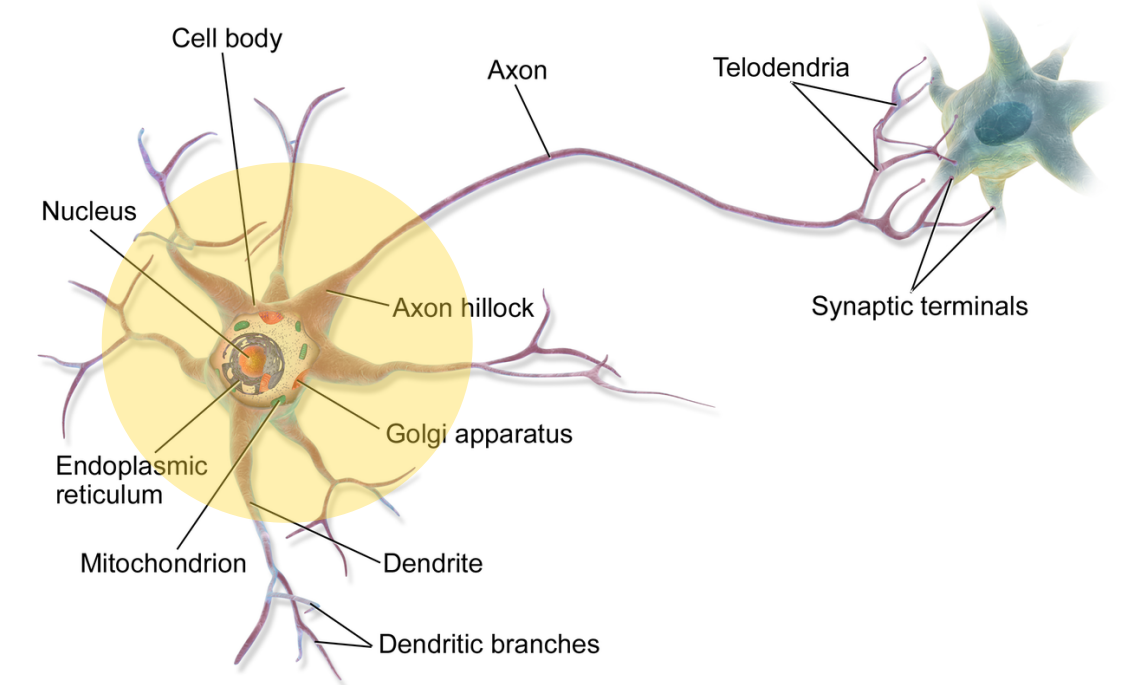


2D PCA Visualization Labeled with Document Source

- comp.os.ms-windows.misc
- rec.sport.baseball
- sci.space
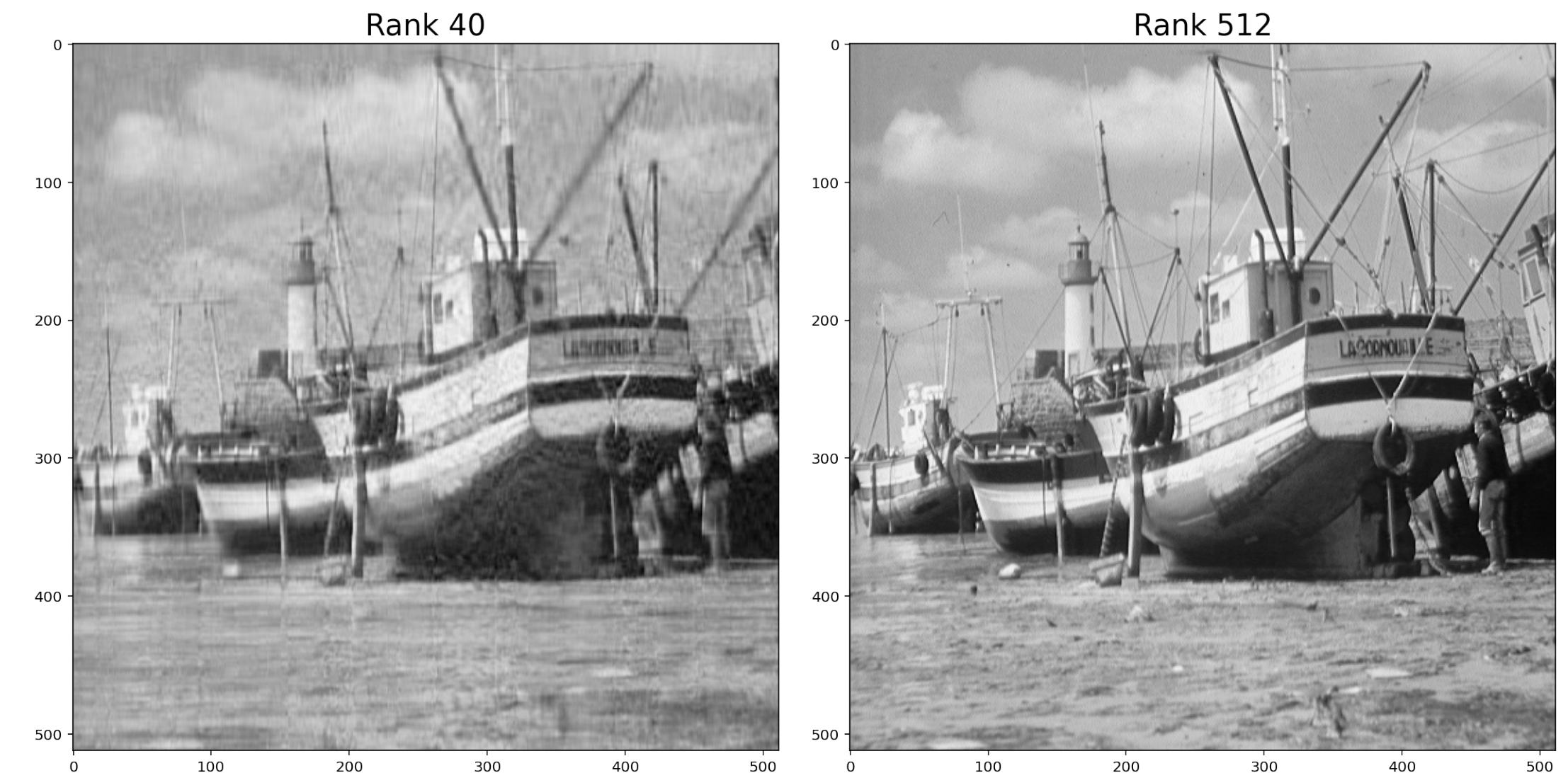
document classification

# Neural Networks (Non-Linearity)



$$\mathbf{x}$$

$$A\mathbf{x}$$

$$Bf(A\mathbf{x})$$

$$f(B f(A\mathbf{x}))$$

$$f(A\mathbf{x})$$

# Neural Networks (Non-Linearity)

Neural networks are models of
artificial neurons bundles.



$\mathbf{x}$

$A\mathbf{x}$   $Bf(A\mathbf{x})$

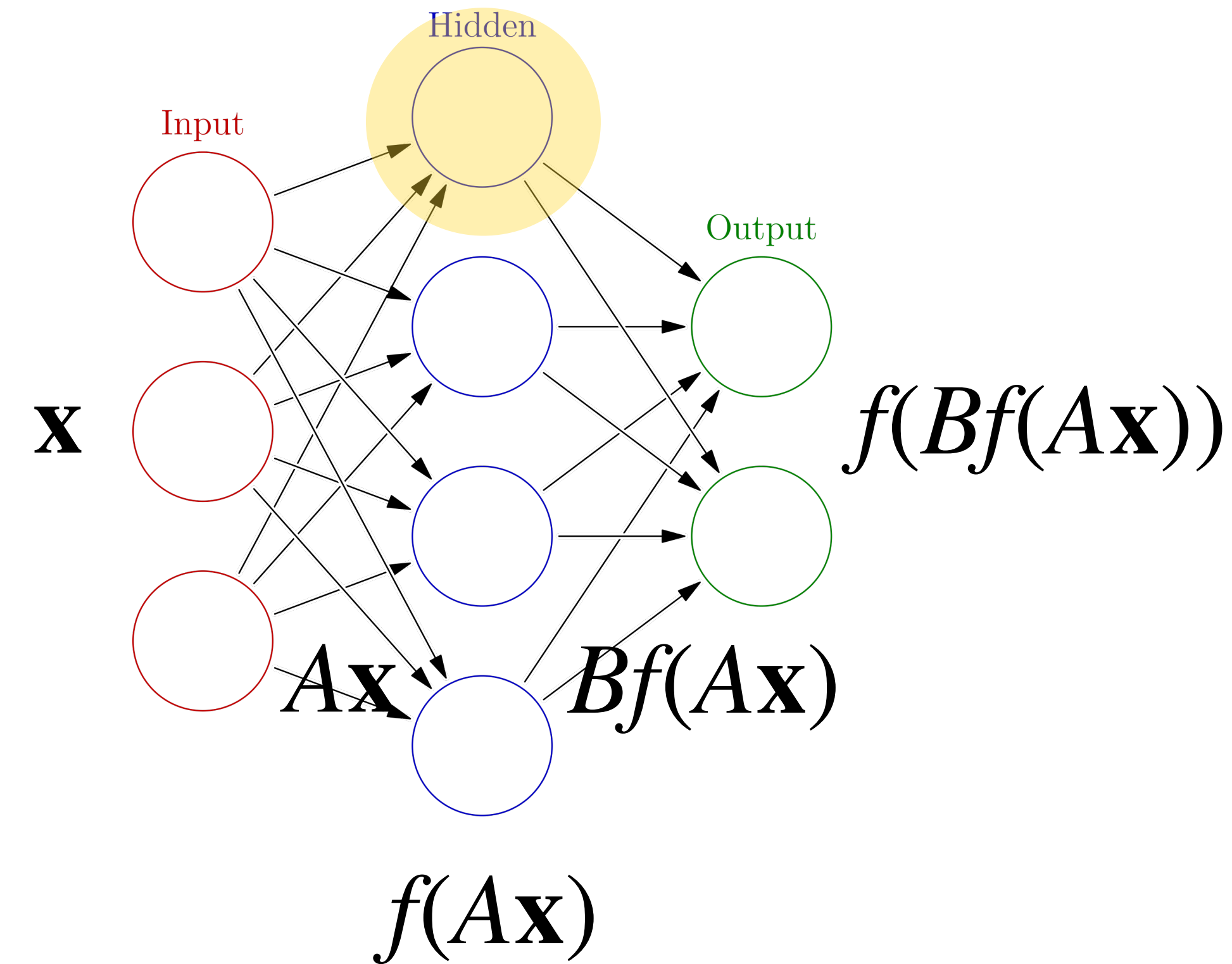$f(A\mathbf{x})$

$f(Bf(A\mathbf{x}))$

# Neural Networks (Non-Linearity)



Neural networks are models of artificial neurons bundles.

Given an input vector $\mathbf{x}$, it is transformed into a *hidden* vector $A\mathbf{x}$ by a linear transformation, and then an *activation function* $f$ is applied to the result.



$\mathbf{x}$

$A\mathbf{x}$   $Bf(A\mathbf{x})$
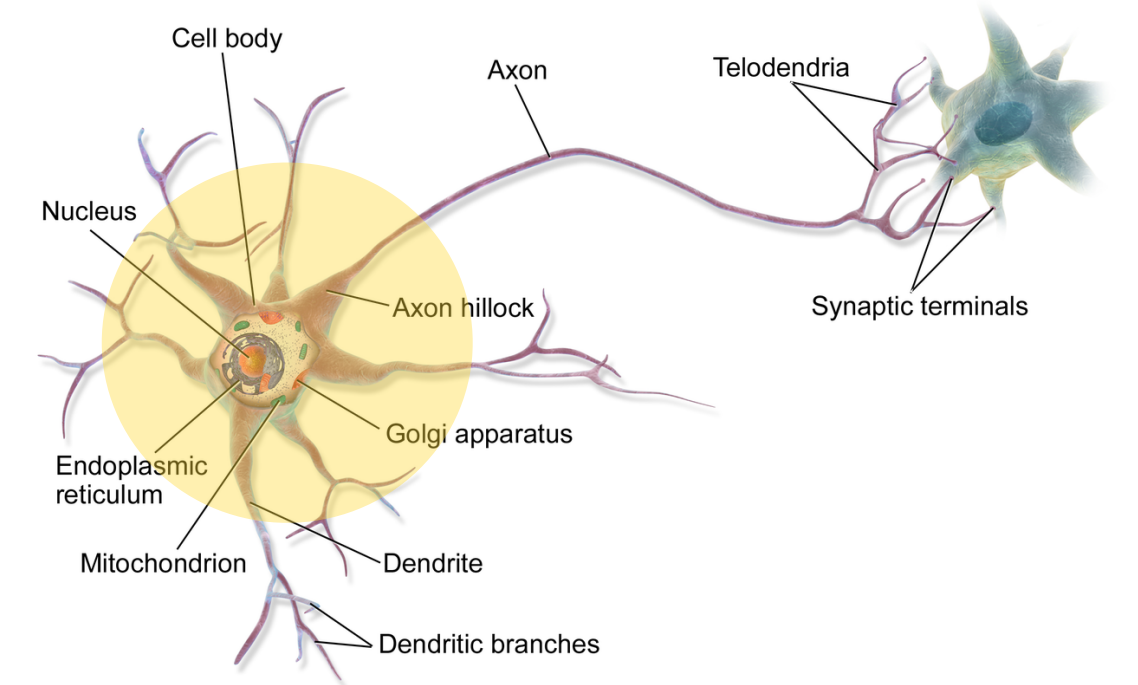
$f(A\mathbf{x})$

$f(Bf(A\mathbf{x}))$
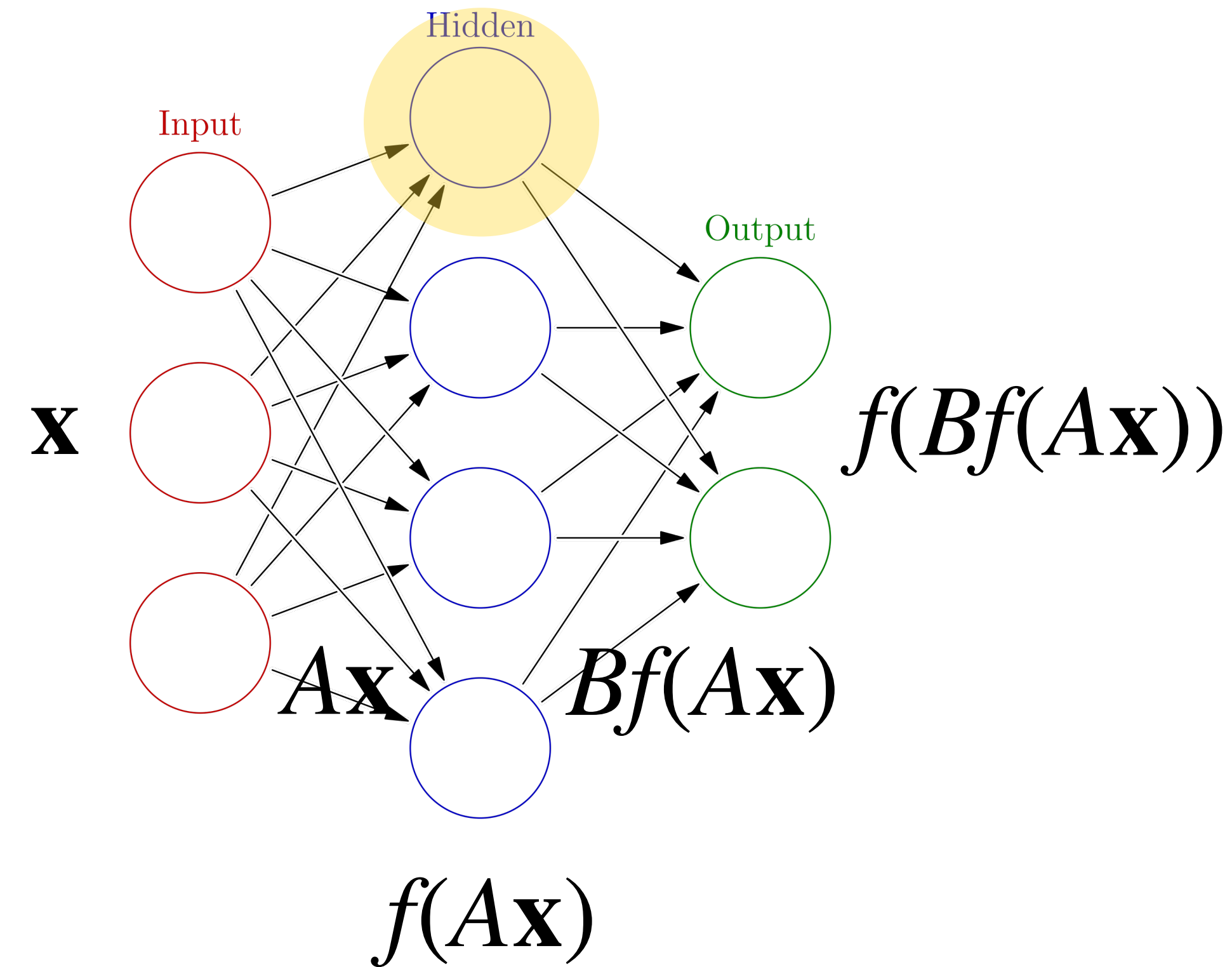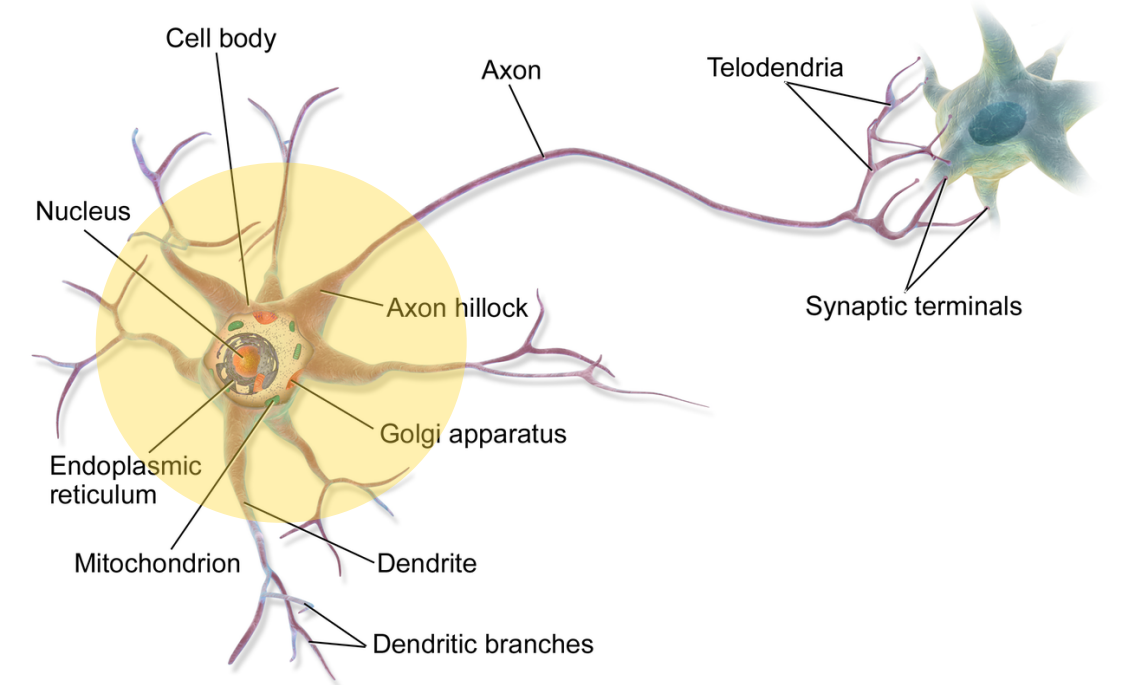
# Neural Networks (Non-Linearity)



Neural networks are models of artificial neurons bundles.

Given an input vector $\mathbf{x}$, it is transformed into a *hidden* vector $A\mathbf{x}$ by a linear transformation, and then an *activation function $f$* is applied to the result.

**Neural networks are just matrix multiplications with intermediate calls to a nonlinear function $f$.**
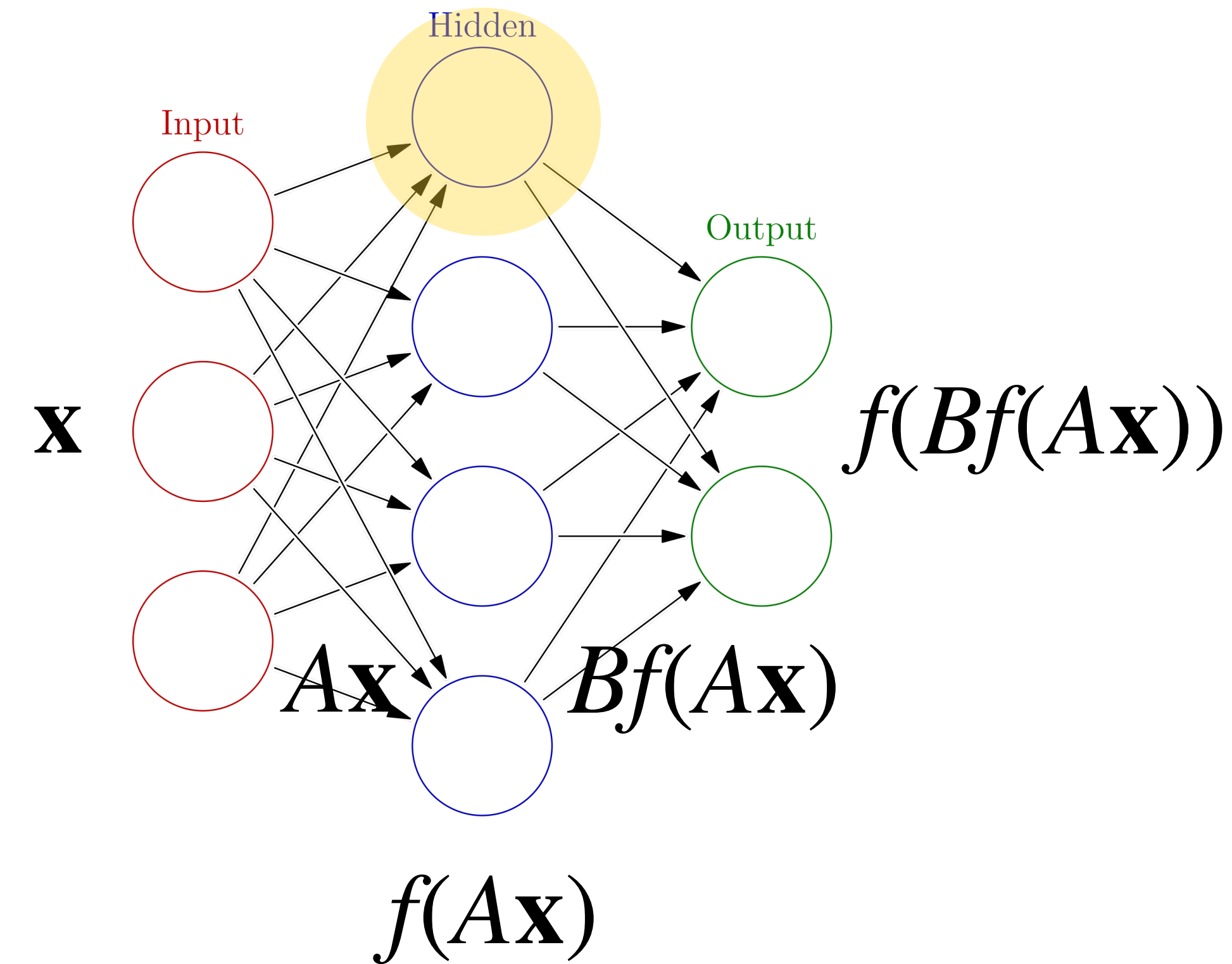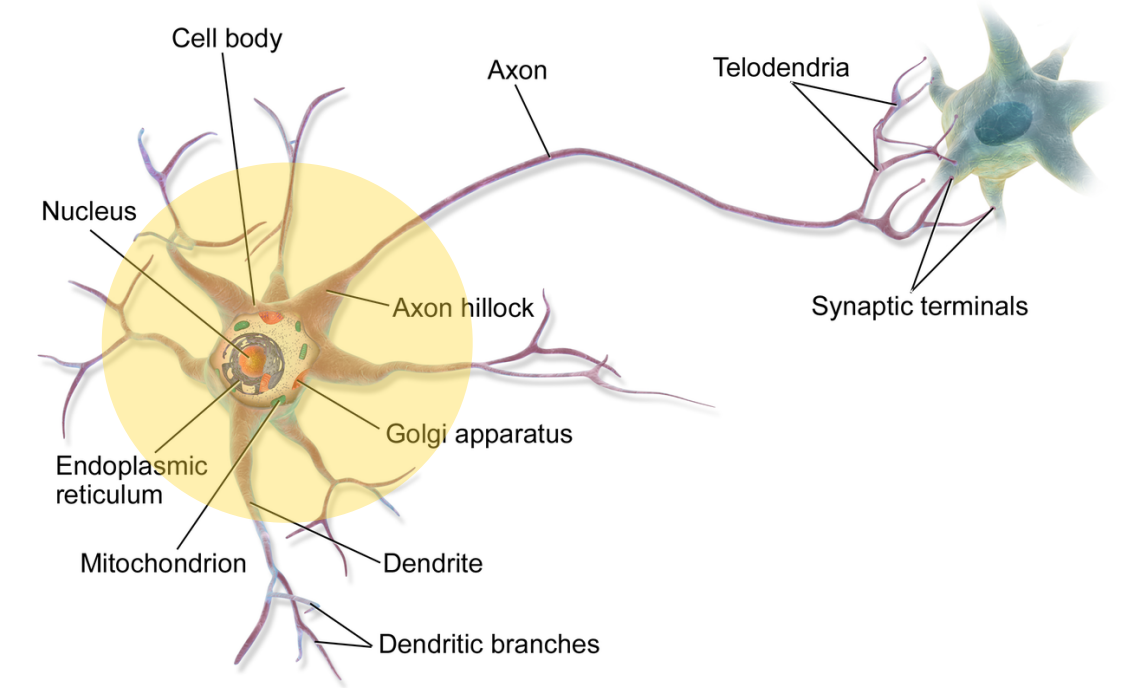
# Neural Networks (Non-Linearity)



Neural networks are models of artificial neurons bundles.

Given an input vector $\mathbf{x}$, it is transformed into a *hidden* vector $A\mathbf{x}$ by a linear transformation, and then an *activation function $f$* is applied to the result.

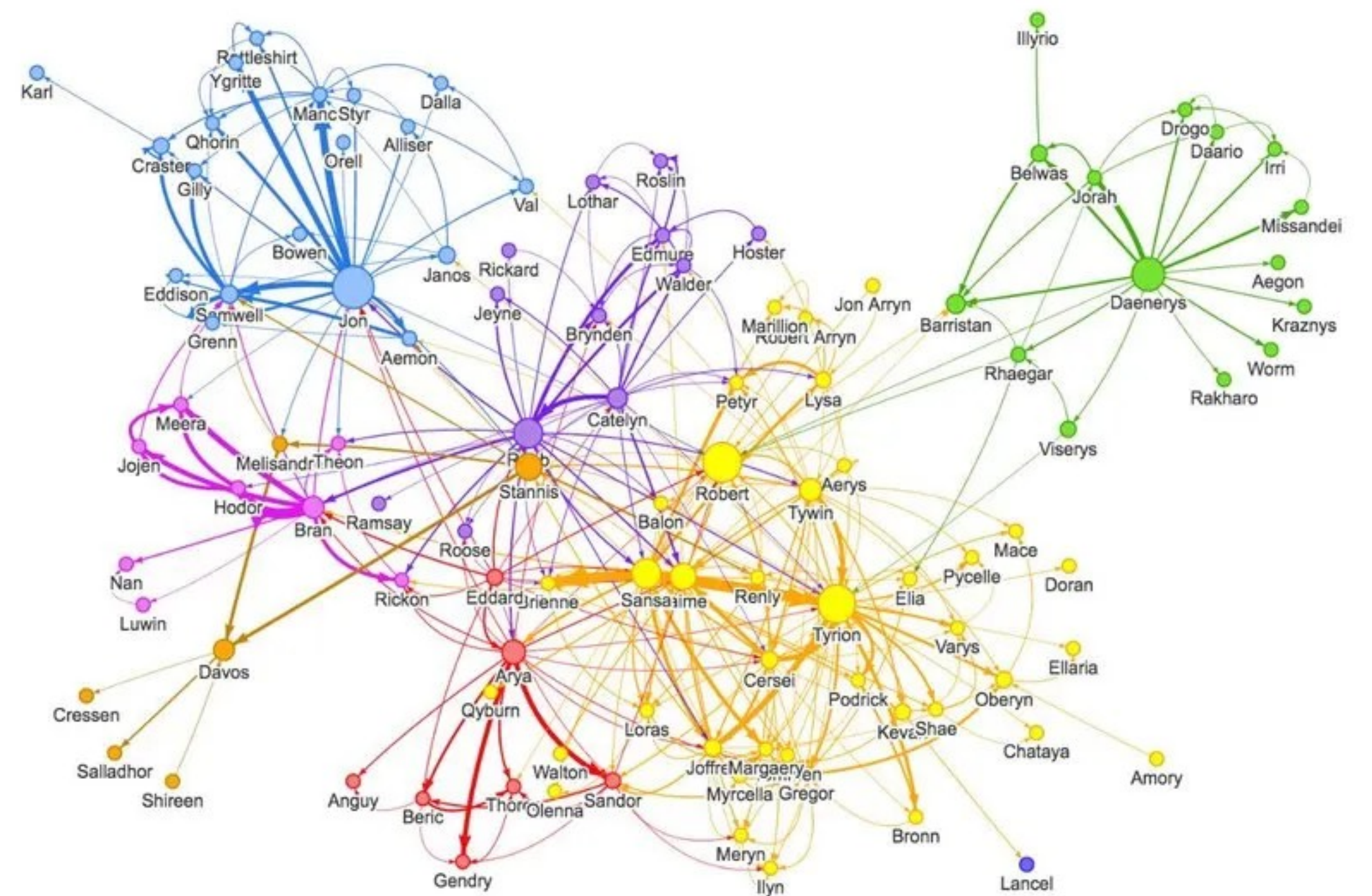**Neural networks are just matrix multiplications with intermediate calls to a nonlinear function $f$.**

$$\mathrm{NN}(\mathbf{x}) = f(A_k(f(A_{k-1}\ldots f(A_1\mathbf{x})))$$

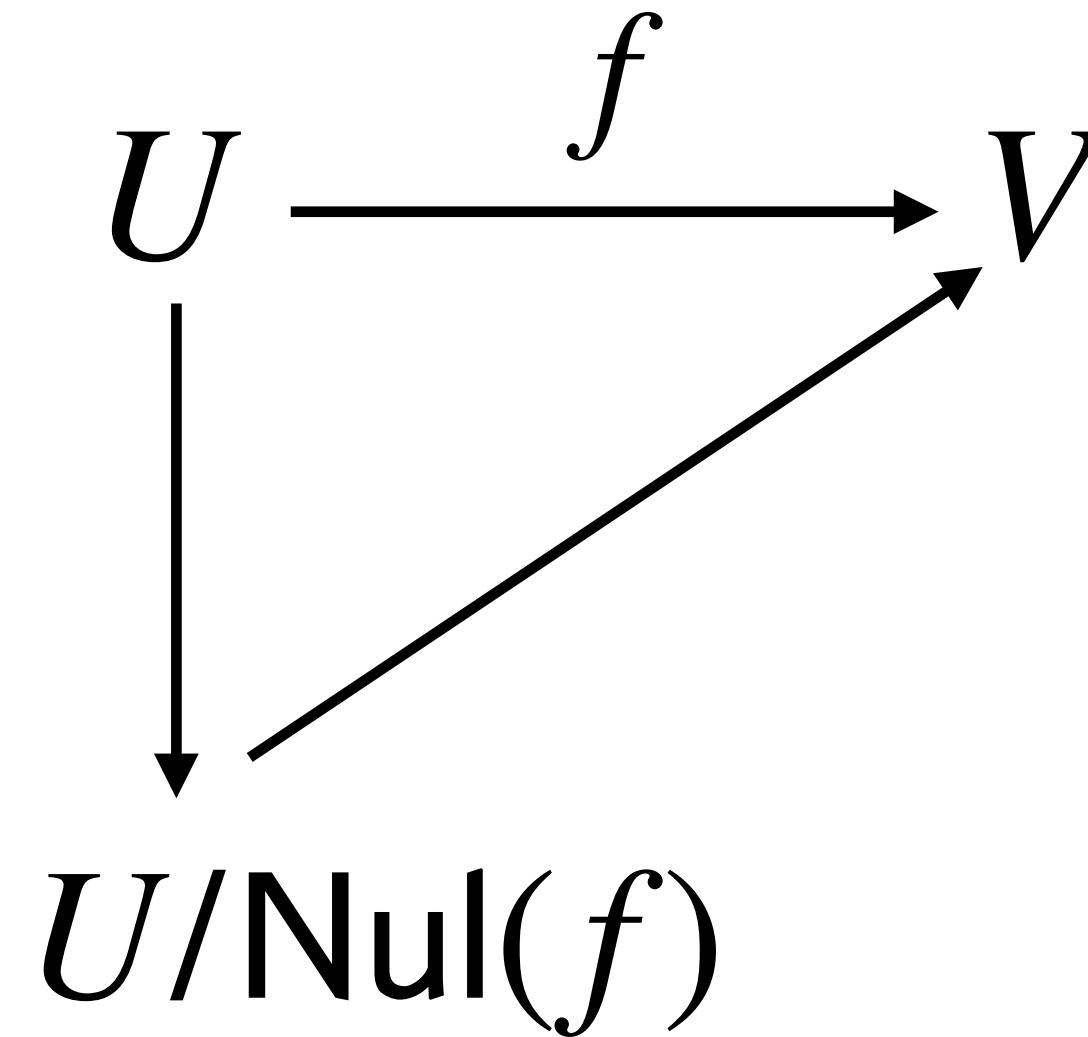# Spectral/Algebraic Graph Theory

Graphs can be viewed as matrices.

The finding eigenvalues in graphs can gives use better clustering and cutting algorithms.

# Abstract Algebra

$$\frac{U}{\mathsf{Nul}(f)} \cong \mathsf{Range}(f)$$

$$U \xrightarrow{\ f\ } V$$

$$U/\mathsf{Nul}(f)$$

There's a lot of beautiful structure in the algebra we've done in this course.

And there are lots of directions to go from here (infinite dimensional spaces, less restrictive settings like groups and modules,...)

# Course List

- CS 365 Foundations of Data Science
- CS 440 Intro to Artificial Intelligence
- CS 480 Intro to Computer Graphics
- CS 505 Intro to Natural Language Processing
- CS 506 Tools for Data Science
- CS 507 Intro to Optimization in ML
- CS 523 Deep Learning
- CS 530 Advanced Algorithms
- CS 531 Advanced Optimization Algorithms
- CS 542 Machine Learning
- CS 565 Algorithmic Data Mining
- CS 581 Computational Fabrication
- CS 583 Audio Computation

*Some of these may not exist anymore...*

# Appreciations

# The Course Staff

I'd like to thank:

**Abhinit Sati, Vishesh Jain, Ieva Sagaitis, Kevin Wrenn, Jin Zhang, Sohan Atluri, Fynn Buesnel, Aseef Imran, Eugene Jung, Chris Min, Wyatt Napier, Kyle Yung**

If you see them around you should thank them as well

# The CS Department Staff

If you're ever in the CS Department office, be kind to the people who work there. They work very hard to keep all our courses running

# The Students of CS132

Thanks for sticking with it

Thanks for giving feedback

Thanks for participating

fin