

# MLSP linear algebra refresher

**"YOU LEARN SOMETHING NEW EVERYDAY"**

**FALSE.**

**YOU LEARN SOMETHING OLD EVERY DAY. JUST  
BECAUSE YOU'VE JUST LEARNED IT DOESN'T MEAN  
IT'S NEW, OTHER PEOPLE ALREADY KNEW IT.**

[quickmeme.com](http://quickmeme.com)

I learned  
something old  
today!

# Book

- Fundamentals of Linear Algebra, Gilbert Strang
- Important to be very comfortable with linear algebra
  - Appears repeatedly in the form of Eigen analysis, SVD, Factor analysis
  - Appears through various properties of matrices that are used in machine learning
    - Often used in the processing of data of various kinds
    - Will use sound and images as examples
- Today's lecture: Definitions
  - Very small subset of all that's used
  - Important subset, intended to help you recollect



# Incentive to use linear algebra

- Simplified notation!

$$\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{y} \quad \longleftrightarrow \quad \sum_j y_j \sum_i x_i a_{ij}$$

- Easier intuition

– *Really convenient geometric interpretations*

- Easy code translation!

```
for i=1:n
    for j=1:m
        c(i)=c(i)+y(j)*x(i)*a(i,j)
    end
end
```



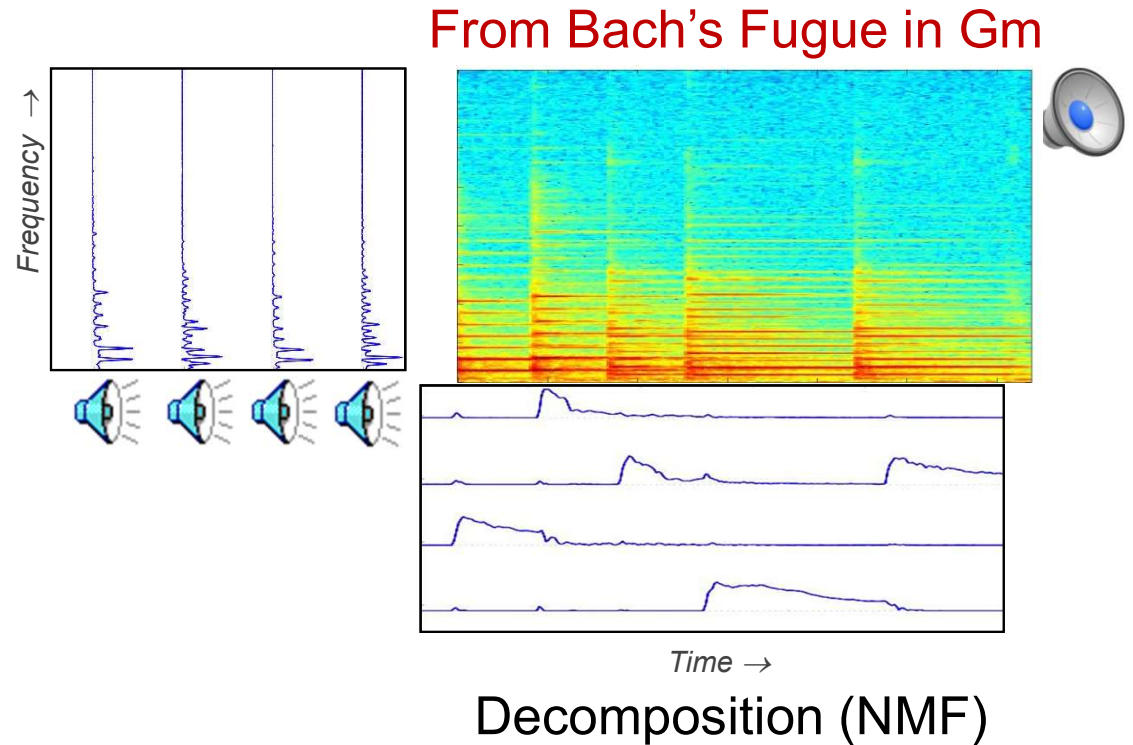
```
C=x*A*y
```

# And other things you can do



Rotation + Projection +  
Scaling + Perspective

- Manipulate Data
- Extract information from data
- Represent data..
- Etc.



# Overview

- Vectors and matrices
- Basic vector/matrix operations
- Various matrix types
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- Solving simultaneous equations
- Projections
- Eigen decomposition
- SVD

# Overview

- **Vectors and matrices**
- **Basic vector/matrix operations**
- Various matrix types
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- Solving simultaneous equations
- Projections
- Eigen decomposition
- SVD

# What is a vector

Column vector

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

An Nx1 vector

$$[a \quad b \quad c]$$

Row vector

A 1xN vector

- A rectangular or horizontal arrangement of numbers

# What is a vector

Column vector

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

An Nx1 vector

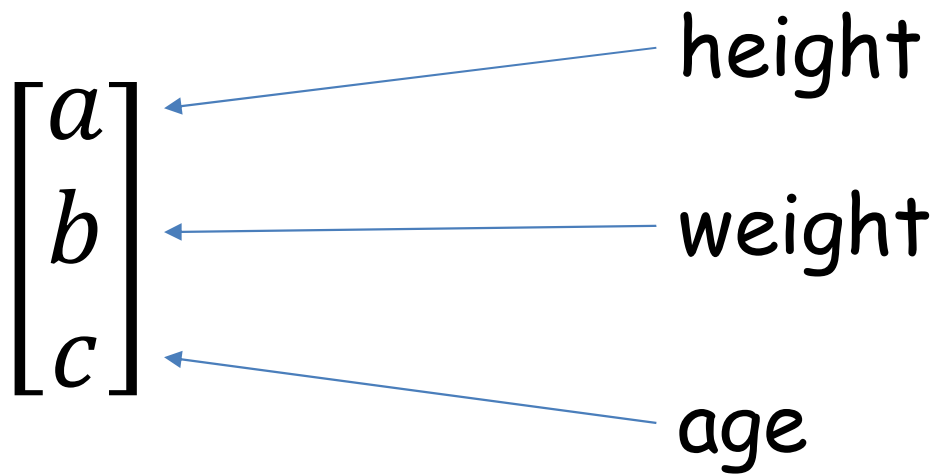
$$[a \quad b \quad c]$$

Row vector

A 1xN vector

- A rectangular or horizontal arrangement of numbers
- Which, without additional context, is actually a useless and meaningless mathematical object

# *A meaningful vector*

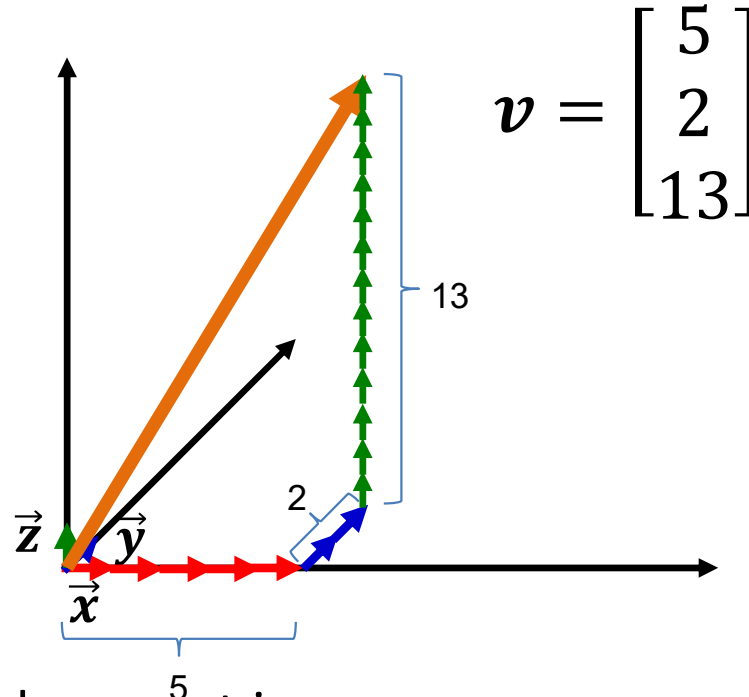


- A rectangular or horizontal arrangement of numbers
- Where each number refers to a different quantity

# What is a vector

$$\mathbf{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\mathbf{v} = a\vec{x} + b\vec{y} + c\vec{z}$$



- A vector represents a net displacement in space
  - If the displacement is from origin, it represents a location in space
- Each component of the vector actually represents the *number of steps* along a set of *basis* directions
  - The vector cannot be interpreted without reference to the bases!!!!
  - The bases are often *implicit* – we all just agree upon them and don't have to mention them

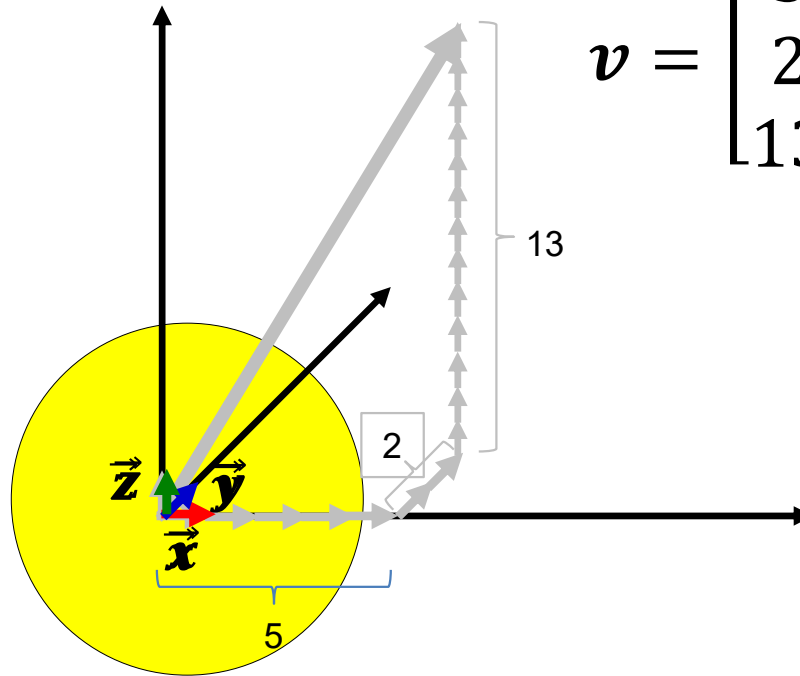


# Standard Bases

$$\mathbf{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\mathbf{v} = \begin{bmatrix} 5 \\ 2 \\ 13 \end{bmatrix}$$

$$\mathbf{v} = a\vec{x} + b\vec{y} + c\vec{z}$$

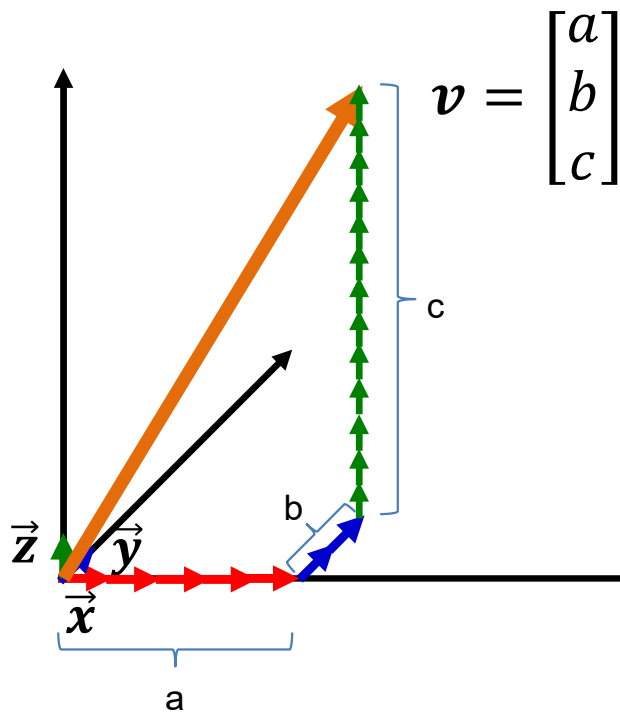


- “Standard” bases are “Orthonormal”
  - Each of the bases is at  $90^\circ$  to every other basis
    - Moving in the direction of one basis results in *no* motion along the directions of other bases
  - All bases are unit length

# A vector by another basis..

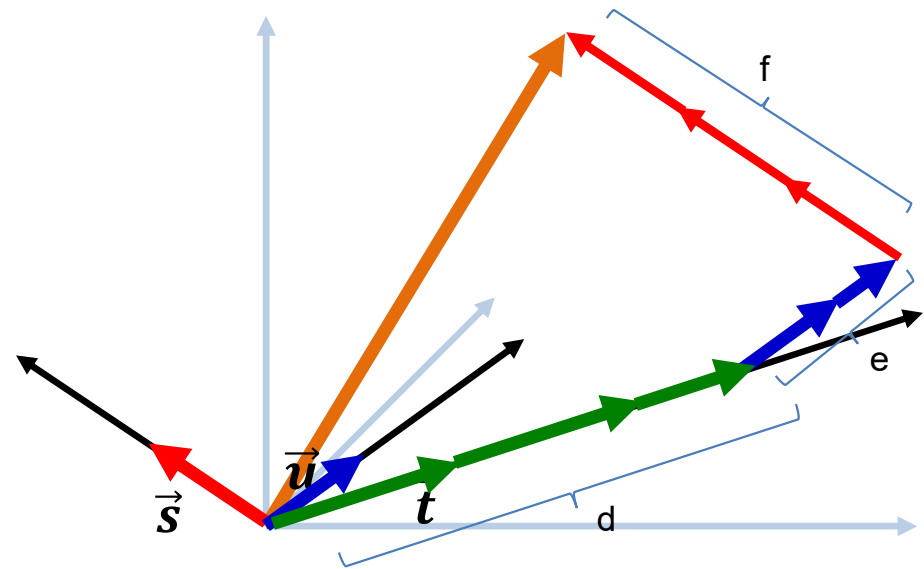
$$v = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ using } \vec{x}, \vec{y}, \vec{z}$$

$$v = a\vec{x} + b\vec{y} + c\vec{z}$$



$$v = d\vec{s} + e\vec{t} + f\vec{u}$$

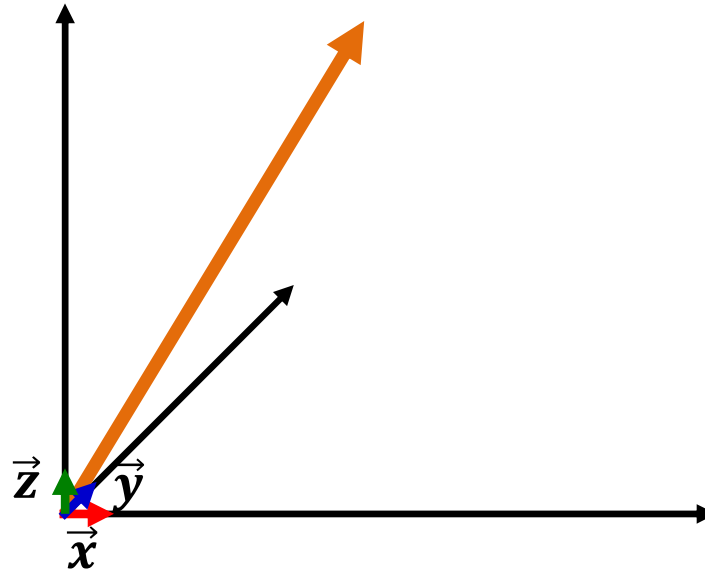
$$v = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$



- For non-standard bases we will generally *have* to specify the bases to be understood

# Length of a vector

$$\mathbf{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$



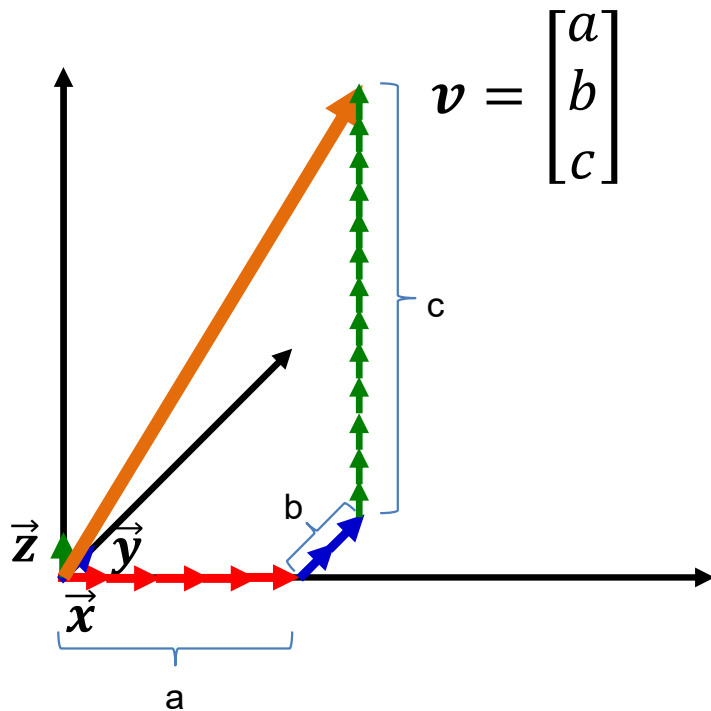
$$|\mathbf{v}| = \sqrt{a^2 + b^2 + c^2}$$

- The Euclidean length of the displacement assuming standard orthonormal bases
  - The Euclidean distance from origin to the location of the vector

# Length of a vector..

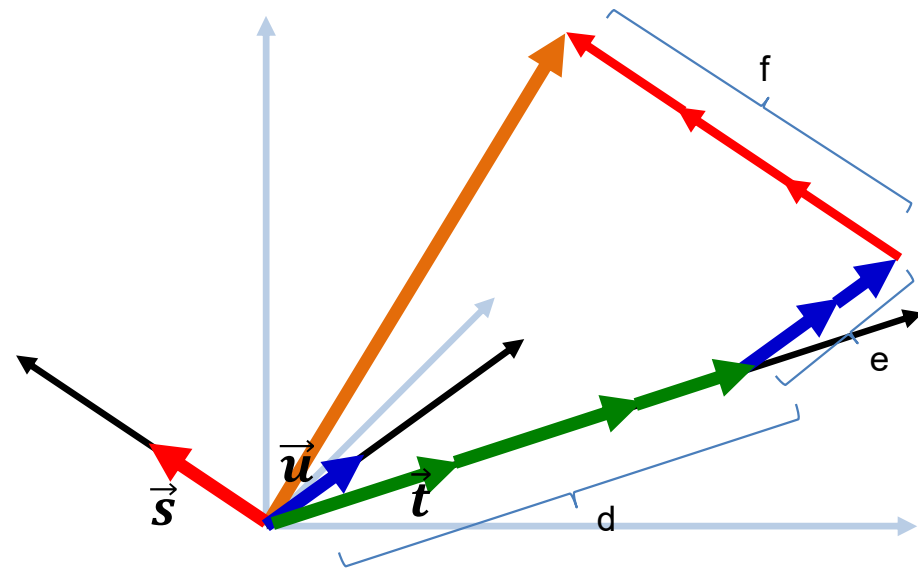
$$\mathbf{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ using } \vec{x}, \vec{y}, \vec{z}$$

$$\mathbf{v} = a\vec{x} + b\vec{y} + c\vec{z}$$



$$\mathbf{v} = d\vec{s} + e\vec{t} + f\vec{u}$$

$$\mathbf{v} = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$

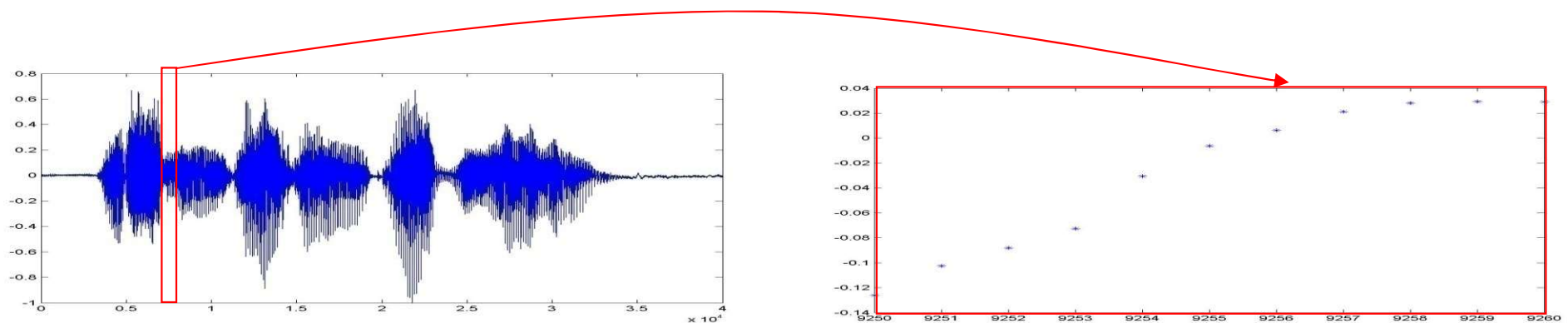


The norm of a vector depends on the bases used to specify it

$$|\mathbf{v}| = \sqrt{a^2 + b^2 + c^2} \quad \text{OR} \quad |\mathbf{v}| = \sqrt{d^2 + e^2 + f^2}$$

# Representing signals as vectors

- Signals are frequently represented as vectors for manipulation
- E.g. A segment of an audio signal

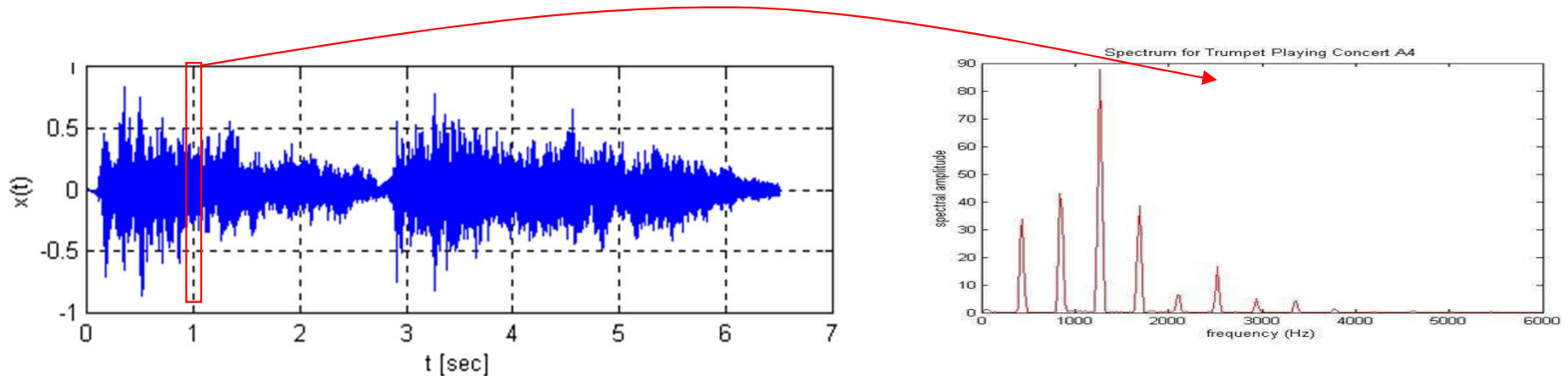


- Represented as a vector of sample values

$$[s_1 \ s_2 \ s_3 \ s_4 \ \dots \ s_N]$$

# Representing signals as vectors

- Signals are frequently represented as vectors for manipulation
- E.g. The *spectrum* segment of an audio signal



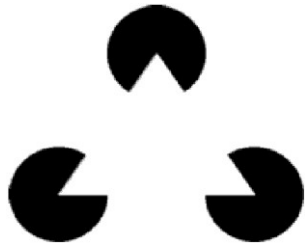
- Represented as a vector of sample values

$$[S_1 \ S_2 \ S_3 \ S_4 \ \dots \ S_M]$$

- Each component of the vector represents a frequency component of the spectrum

# Representing an image as a vector

- 3 pacmen
- A 321 x 399 grid of pixel values
  - Row and Column = position
- A 1 x 128079 vector
  - “Unraveling” the image


$$[1 \ 1 \ . \ 1 \ 1 \ . \ 0 \ 0 \ 0 \ . \ . \ 1]$$

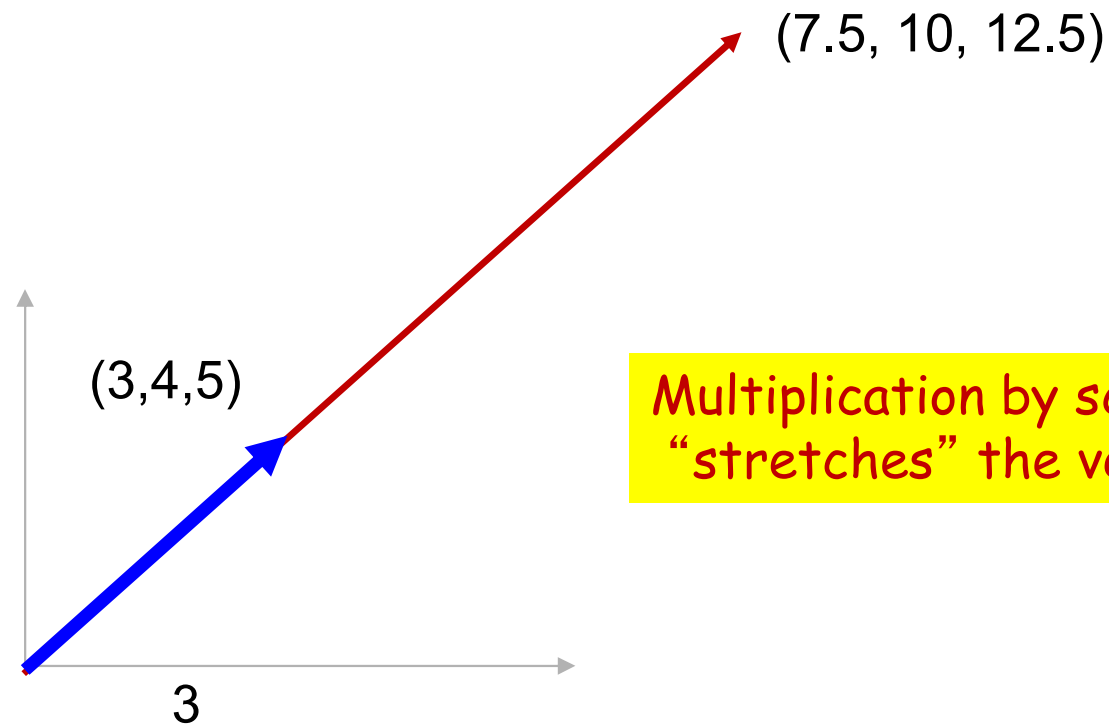
- Note: This can be recast as the grid that forms the image

# Vector operations

- Addition
- Multiplication
- Inner product
- Outer product

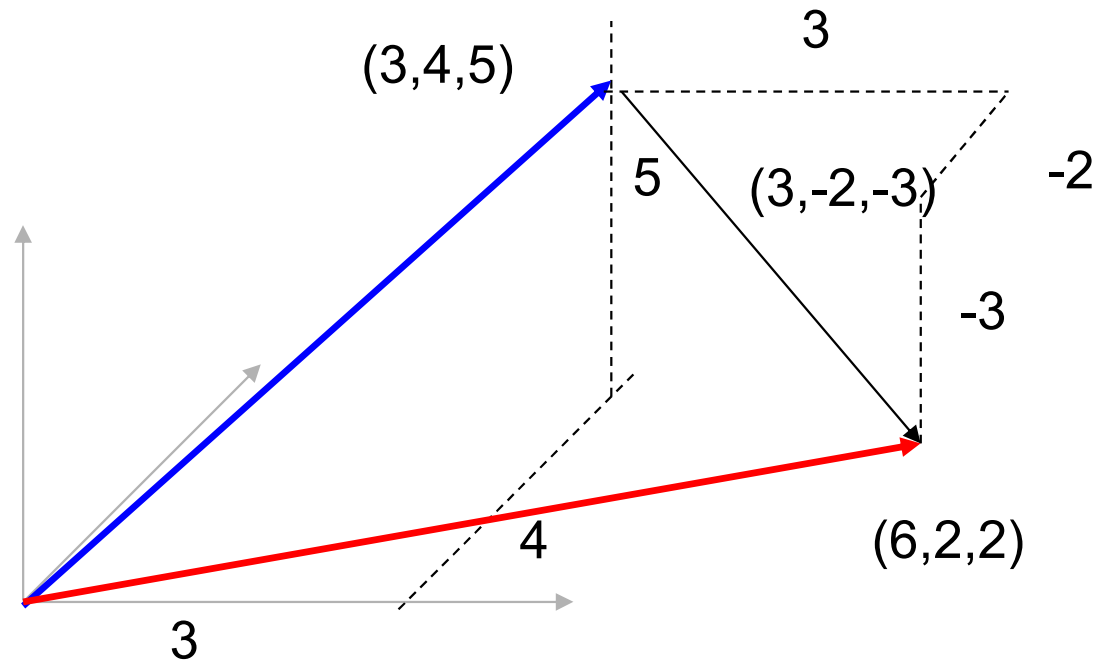


# Vector Operations: Multiplication by scalar



- Vector multiplication by scalar: each component multiplied by scalar
  - $2.5 \times [3, 4, 5] = [7.5, 10, 12.5]$
- Note: as a result, vector norm is also multiplied by the scalar
  - $||2.5 \times [3, 4, 5]|| = 2.5 \times ||[3, 4, 5]||$

# Vector Operations: Addition



- Vector addition: individual components add  
–  $[3, 4, 5] + [3, -2, -3] = [6, 2, 2]$

# Vector operation: Inner product

- Multiplication of a row vector by a column vector to result in a scalar
  - Note order of operation
  - The *inner* product between two column vectors  $\mathbf{u}$  and  $\mathbf{v}$  is the product of  $\mathbf{u}^T$  and  $\mathbf{v}$
  - Also called the “dot” product

$$\mathbf{u} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^T \mathbf{v} = [a \quad b \quad c] \begin{bmatrix} d \\ e \\ f \end{bmatrix} = a \cdot d + b \cdot e + c \cdot f$$

# Vector operation: Inner product

- The inner product of a vector with itself is its squared norm
  - This will be the squared length

$$\mathbf{u} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\mathbf{u} \cdot \mathbf{u} = \mathbf{u}^T \mathbf{u} = a^2 + b^2 + c^2 = \|\mathbf{u}\|^2$$

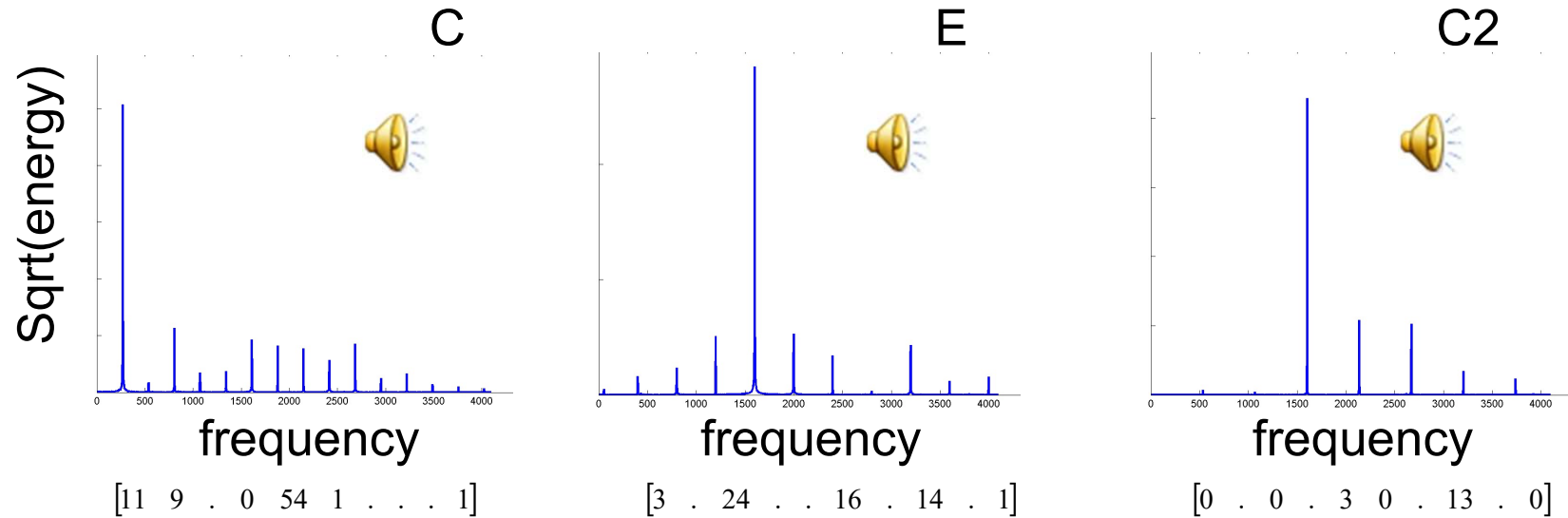
# Vector dot product

- Example:
  - Coordinates are yards, not ave/st
  - $\mathbf{a} = [200 \ 1600]$ ,
  - $\mathbf{b} = [770 \ 300]$
- The dot product of the two vectors relates to the length of a *projection*
  - How much of the first vector have we covered by following the second one?
  - Must normalize by the length of the “target” vector

$$\frac{\mathbf{a} \cdot \mathbf{b}^T}{\|\mathbf{a}\|} = \frac{[200 \ 1600] \cdot \begin{bmatrix} 770 \\ 300 \end{bmatrix}}{\|[200 \ 1600]\|} \approx 393\text{yd}$$

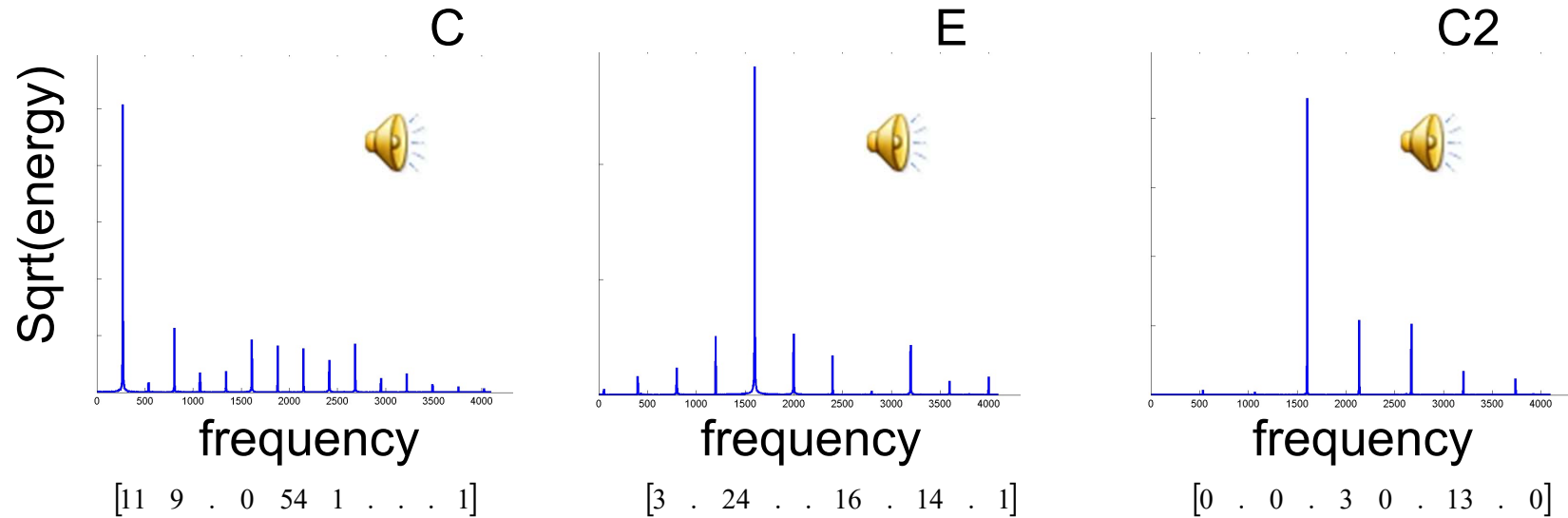


# Vector dot product



- Vectors are spectra
  - Energy at a discrete set of frequencies
  - Actually  $1 \times 4096$
  - X axis is the *index* of the number in the vector
    - Represents frequency
  - Y axis is the value of the number in the vector
    - Represents magnitude

# Vector dot product

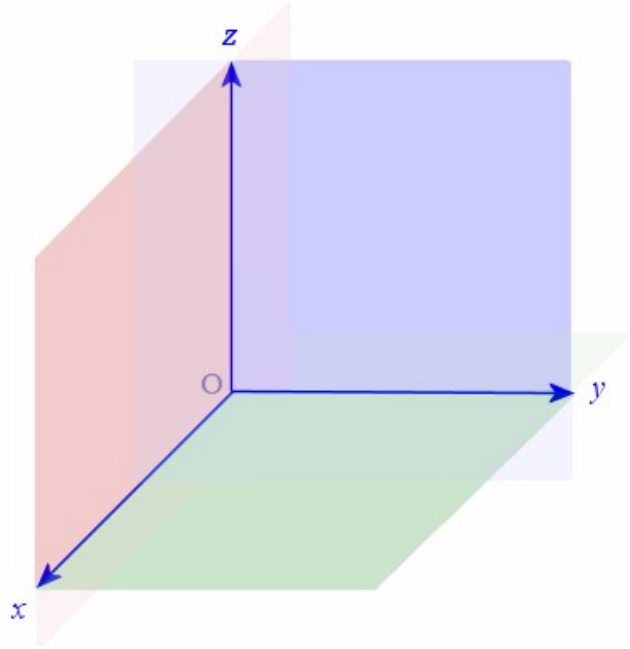


- How much of C is also in E
  - How much can you fake a C by playing an E
  - $C.E / |C| |E| = 0.1$
  - Not very much
- How much of C is in C2?
  - $C.C2 / |C| / |C2| = 0.5$
  - Not bad, you can fake it

# The notion of a “Vector Space”

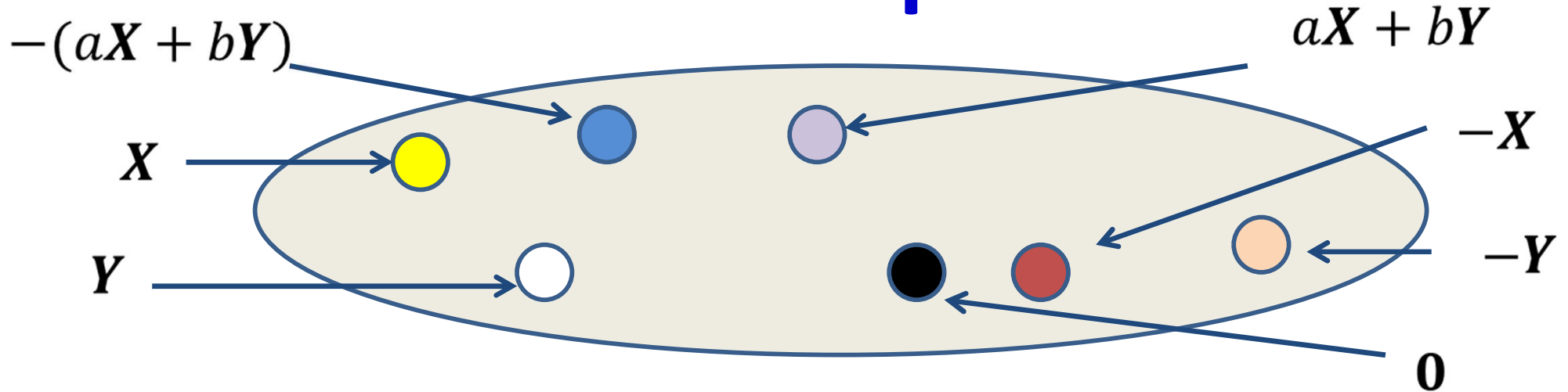


# An introduction to *spaces*



- Conventional notion of “space”: a geometric construct of a certain number of “dimensions”
  - E.g. the 3-D space that this room and every object in it lives in

# A vector space



- A *vector space* is an infinitely large set of vectors with the following properties
  - The set includes the zero vector (of all zeros)
  - The set is “closed” under addition
    - If  $X$  and  $Y$  are in the set,  $aX + bY$  is also in the set for any two scalars  $a$  and  $b$
  - For every  $X$  in the set, the set also includes the additive inverse  $Y = -X$ , such that  $X + Y = 0$

# Additional Properties

- Additional requirements:
  - Scalar multiplicative identity element exists:  
 $1X = X$
  - Addition is associative:  $X + Y = Y + X$
  - Addition is commutative:  $(X+Y)+Z = X+(Y+Z)$
  - Scalar multiplication is commutative:  
 $a(bX) = (ab) X$
  - Scalar multiplication is distributive:  
 $(a+b)X = aX + bX$   
 $a(X+Y) = aX + aY$

# Example of vector space

$$\mathbf{S} = \left\{ \begin{bmatrix} x \\ y \\ z \end{bmatrix} \text{ for all } x, y, z \in \mathcal{R} \right\}$$

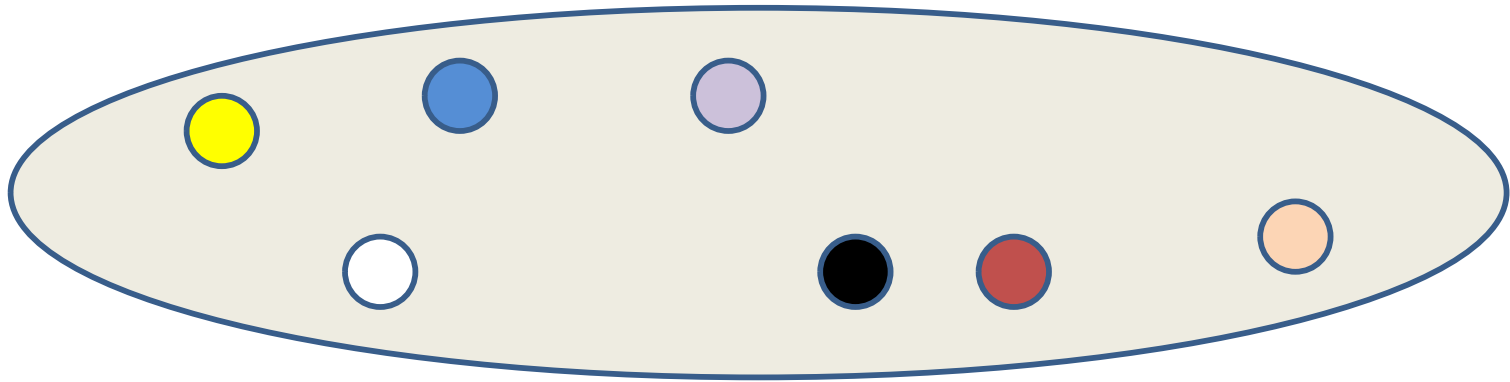
- Set of *all* three-component column vectors
  - Note we used the term *three-component*, rather than *three-dimensional*
- The set includes the zero vector
- For every  $\mathbf{X}$  in the set  $\alpha \in \mathcal{R}$ , every  $\alpha\mathbf{X}$  is in the set
- For every  $\mathbf{X}, \mathbf{Y}$  in the set,  $\alpha\mathbf{X} + \beta\mathbf{Y}$  is in the set
- $-\mathbf{X}$  is in the set
- Etc.

# Example : a function space

$$\mathcal{S} = \left\{ \begin{array}{l} a \cos(x) + b \sin 2x, \text{ for all } a, b \in \mathcal{R} \\ x \in [-\pi, \pi] \end{array} \right\}$$

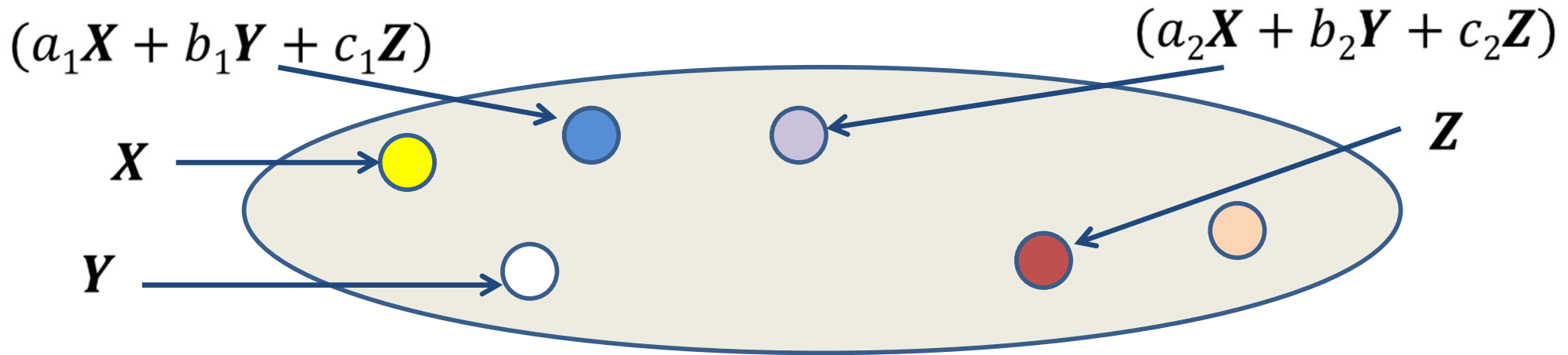
- Entries are *functions* from  $[-\pi, \pi]$  to  $\mathcal{R}$   
 $f: [-\pi, \pi] \rightarrow \mathcal{R}$
- Define  $(f + g)(x) = f(x) + g(x)$  for any  $f$  and  $g$  in this set
- Verify that this is a space!

# Dimension of a space



- Every element in the space can be composed of linear combinations of some other elements in the space
  - For any  $\mathbf{X}$  in  $\mathbf{S}$  we can write  $\mathbf{X} = a\mathbf{Y}_1 + b\mathbf{Y}_2 + c\mathbf{Y}_3..$  for some other  $\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3 ..$  in  $\mathbf{S}$ 
    - Trivial to prove..

# Dimension of a space



- What is the smallest subset of elements that can compose the entire set?
  - There may be multiple such sets
- The elements in this subset are called “bases”
  - The subset is a “basis” set
- The number of elements in the subset is the “dimensionality” of the space

# Dimensions: Example

$$\mathbf{S} = \left\{ \begin{bmatrix} x \\ y \\ z \end{bmatrix} \text{ for all } x, y, z \in \mathcal{R} \right\}$$

- What is the dimensionality of this vector space



# Dimensions: Example

$$\mathbf{Z} = \left\{ a \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + b \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}, \text{ for all } a, b \in \mathcal{R} \right\}$$

- What is the dimensionality of this vector space?
  - First confirm this is a proper vector space
- Note: all elements in  $\mathbf{Z}$  are also in  $\mathbf{S}$  (slide 36)
  - $\mathbf{Z}$  is a *subspace* of  $\mathbf{S}$

# Poll 1

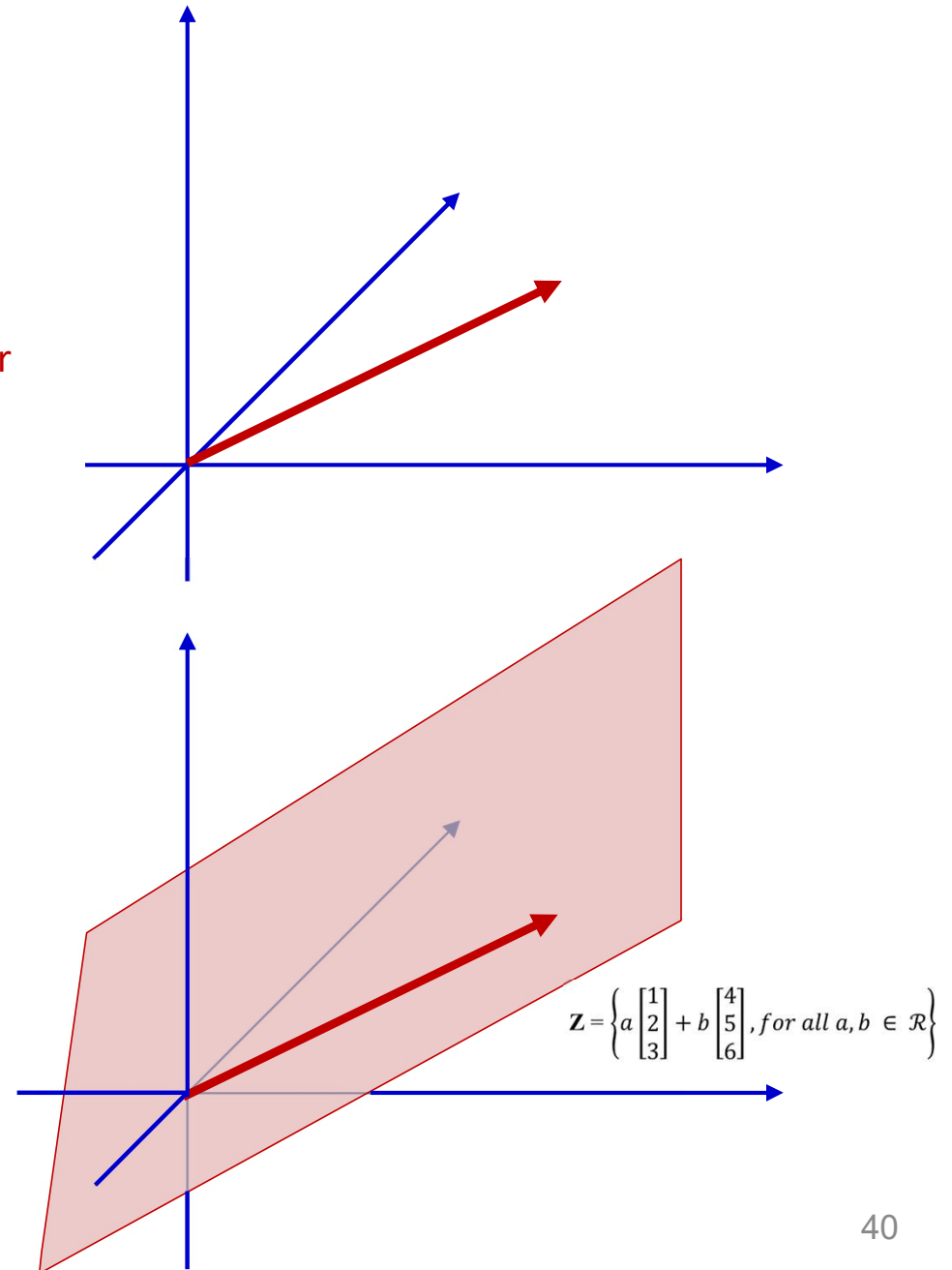
$$\mathbf{S} = \left\{ \begin{array}{l} a\cos(x) + b\sin(3x) \text{ for all } a, b, \in \mathcal{R}, \\ x \in [-\pi, \pi] \end{array} \right\}$$

- What is the dimensionality of this space?

- Return to reality..

# Returning to dimensions..

- Two interpretations of “dimension”
- The *spatial* dimension of a vector:
  - The number of components in the vector
  - An N-component vector “lives” in an N-dimensional space
  - Essentially a “stand-alone” definition of a vector against “standard” bases
- The *embedding* dimension of the vector
  - The dimensionality of the *subspace* the vector actually lives in
  - Only makes sense in the context where the vector is one element of a restricted set, e.g. a subspace or hyperplane
- Much of machine learning and signal processing is aimed at finding the latter from collections of vectors



# Matrices..

# What is a *matrix*

A 2x3 matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 2.2 & 6 \\ 3.1 & 1 & 5 \end{bmatrix}$$

A 3x2 matrix

$$\mathbf{B} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- Rectangular (or square) arrangement of numbers

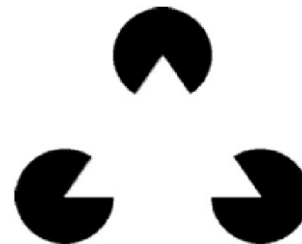
# Dimensions of a matrix

- The matrix size is specified by the number of rows and columns

$$\mathbf{c} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \mathbf{r} = [a \quad b \quad c]$$

- $\mathbf{c} = 3 \times 1$  matrix: 3 rows and 1 column (vectors are matrices too)
- $\mathbf{r} = 1 \times 3$  matrix: 1 row and 3 columns

$$\mathbf{S} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \mathbf{R} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$



- $\mathbf{S} = 2 \times 2$  matrix
- $\mathbf{R} = 2 \times 3$  matrix
- Pacman =  $321 \times 399$  matrix

# Dimensionality and Transposition

- A transposed matrix gets all its row (or column) vectors transposed in order
  - An NxM matrix becomes an MxN matrix

$$\mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad \mathbf{x}^T = [a \quad b \quad c] \quad \mathbf{y} = [a \quad b \quad c], \quad \mathbf{y}^T = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}, \quad \mathbf{X}^T = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix} \quad \mathbf{M} = \begin{bmatrix} \text{img} \end{bmatrix}, \quad \mathbf{M}^T = \begin{bmatrix} \text{img} \end{bmatrix}$$



# What is a *matrix*

A 2x3 matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 2.2 & 6 \\ 3.1 & 1 & 5 \end{bmatrix}$$

A 3x2 matrix

$$\mathbf{B} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- A matrix by itself is uninformative, except through its relationship to vectors

# Interpreting matrices

- Matrices as transforms
- Matrices as data containers
- Matrices as compositional building blocks for vector spaces

# Interpreting matrices

- Matrices as transforms
- Matrices as data containers
- Matrices as compositional building blocks for vector spaces

# Matrices as transforms

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$$

- Multiplying a vector by a matrix *transforms* the vector

$$- \mathbf{A}\mathbf{b} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} a_{11}b_1 + a_{12}b_2 + a_{13}b_3 + a_{14}b_4 \\ a_{21}b_1 + a_{22}b_2 + a_{23}b_3 + a_{24}b_4 \\ a_{31}b_1 + a_{32}b_2 + a_{33}b_3 + a_{34}b_4 \end{bmatrix}$$

- A matrix is a *transform* that *transforms* a vector
  - Above example: *left multiplication*. Matrix transforms a column vector
  - Dimensions must match!!
    - No. of columns of matrix = size of vector
    - Result inherits the number of rows from the matrix

# Matrices as transforms

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$$

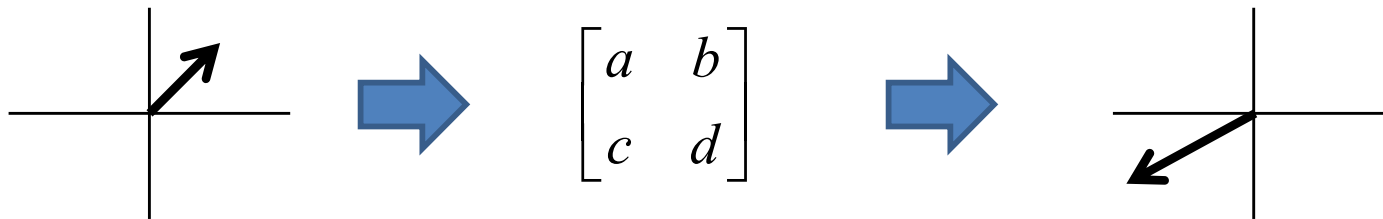
- Multiplying a vector by a matrix *transforms* the vector

$$- \mathbf{bA} = [b_1 \quad b_2 \quad b_3] \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} = \begin{bmatrix} a_{11}b_1 + a_{21}b_2 + a_{31}b_3 \\ a_{12}b_1 + a_{22}b_2 + a_{32}b_3 \\ a_{13}b_1 + a_{23}b_2 + a_{33}b_3 \\ a_{14}b_1 + a_{24}b_2 + a_{34}b_3 \end{bmatrix}^T$$

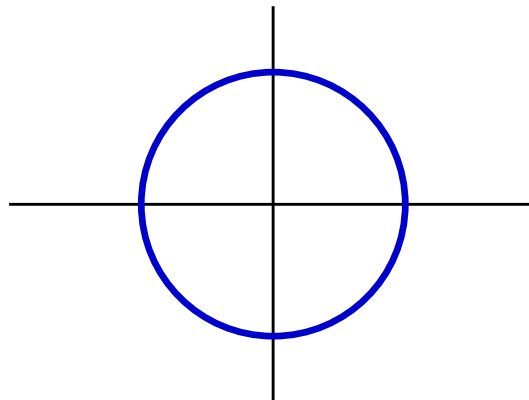
- A matrix is a *transform* that *transforms* a vector
  - Example: *right multiplication*. Matrix transforms a row vector
  - Dimensions must match!!
    - No. of rows of matrix = size of vector
    - Result inherits the number of columns from the matrix

# Matrices transform a space

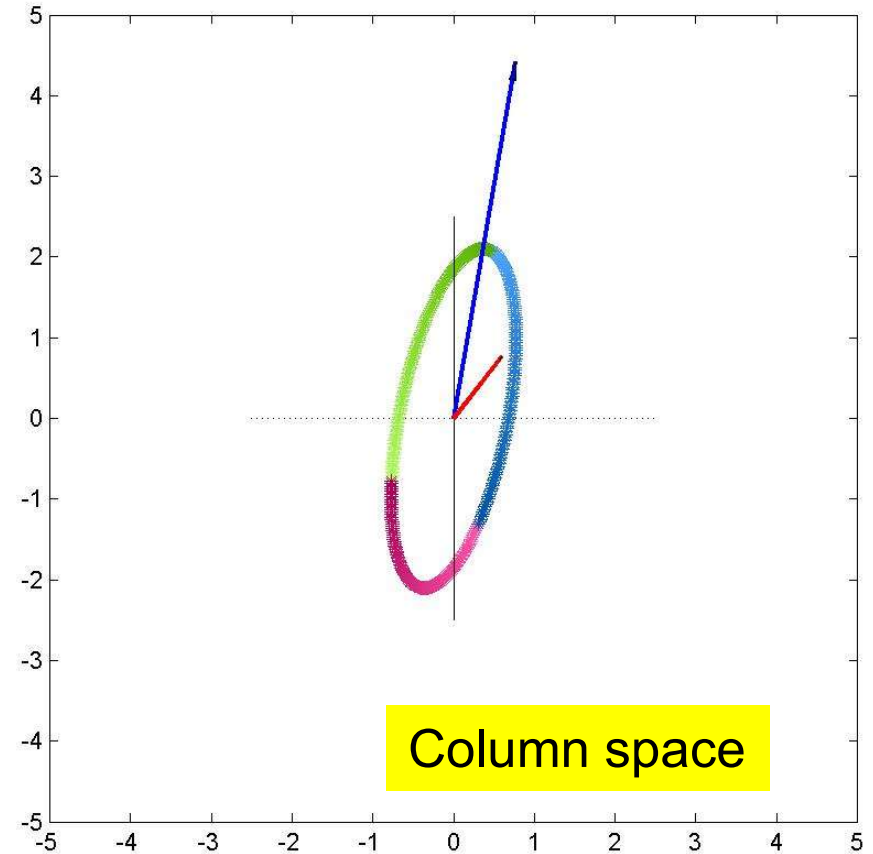
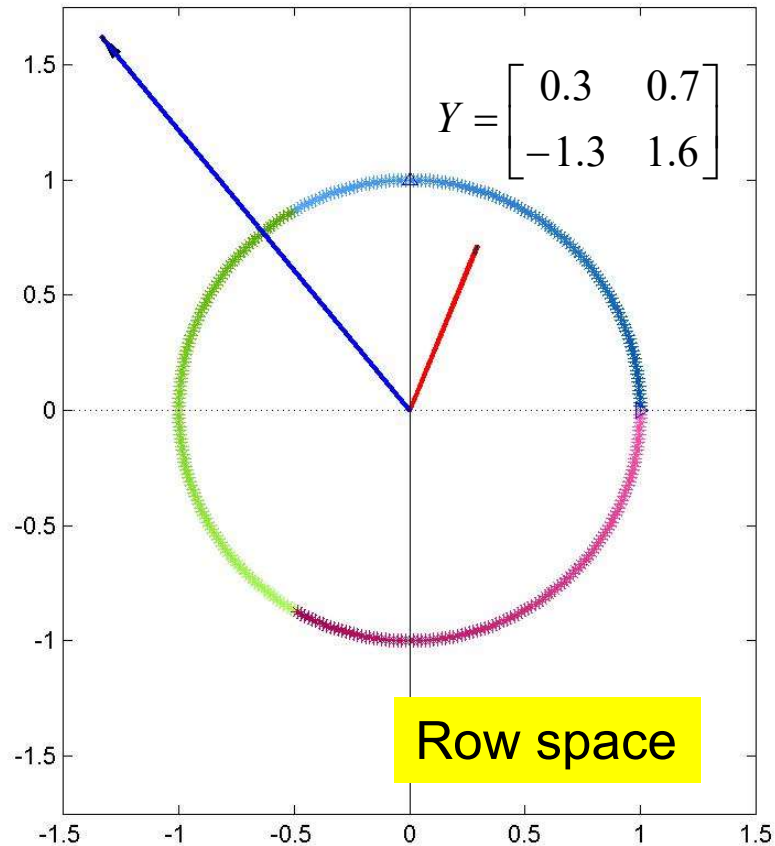
- A matrix is a **transform** that modifies vectors and vector spaces



- So how does it transform the *entire space*?
- E.g. how will it transform the following figure?

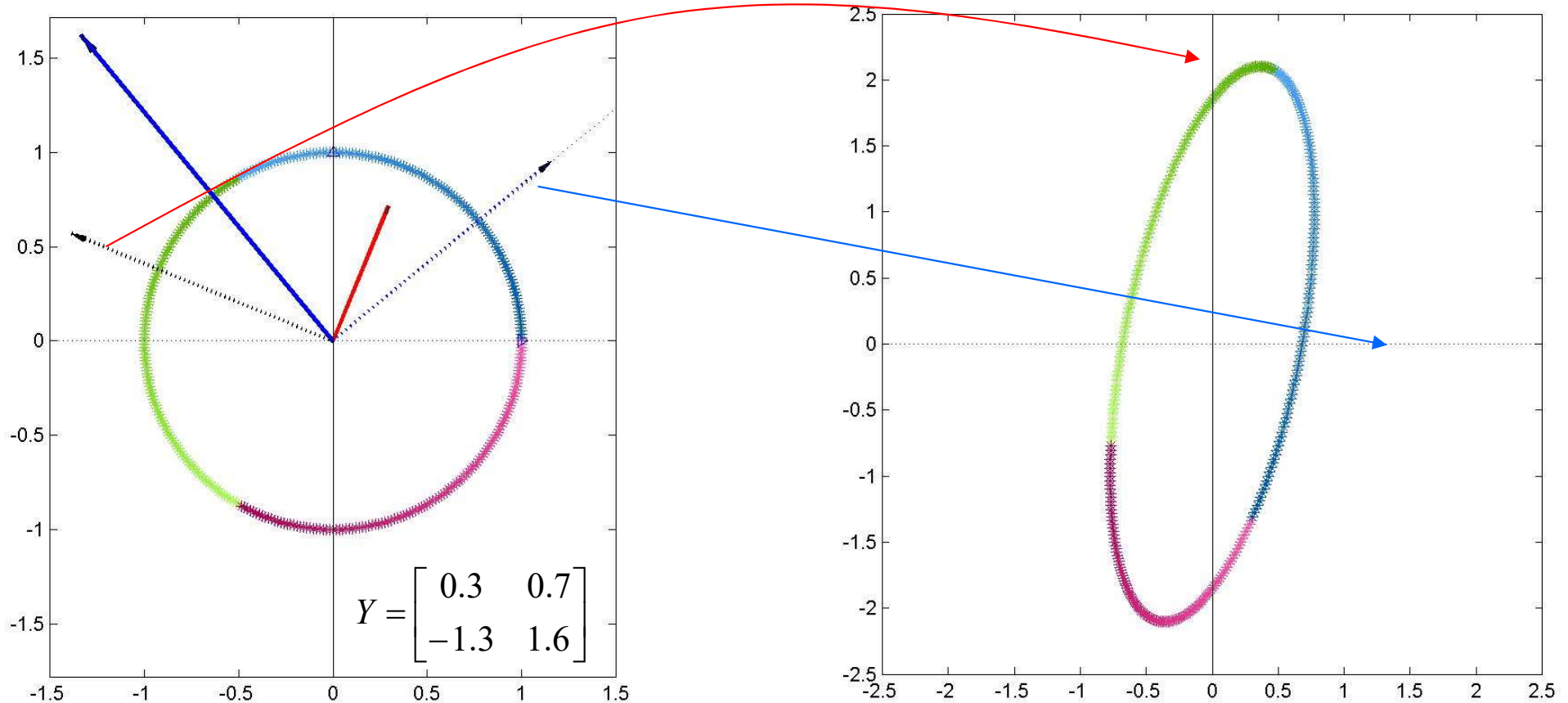


# Multiplication of vector space by matrix



- The matrix rotates and scales the space
  - Including its own row vectors

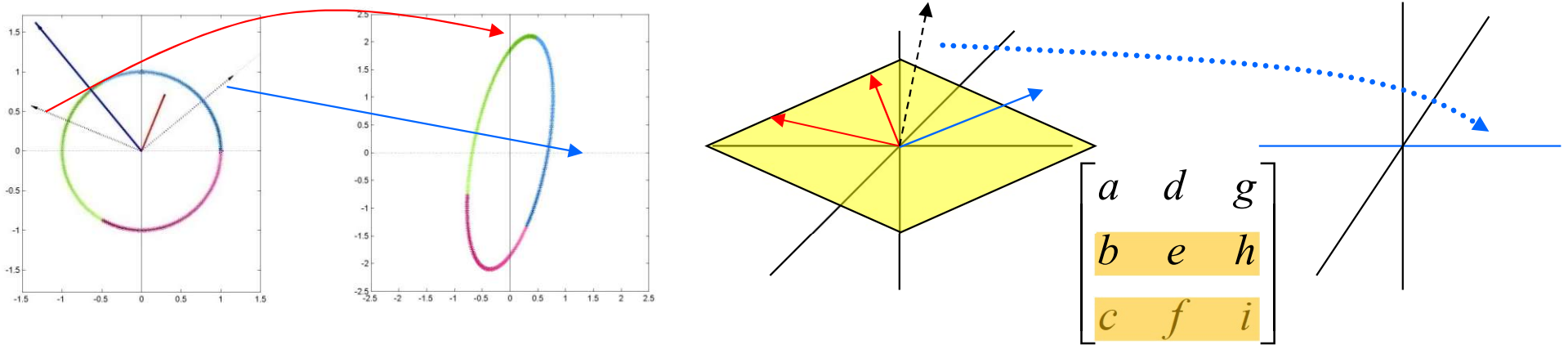
# Multiplication of vector space by matrix



- The *normals* to the row vectors in the matrix become the new axes
  - X axis = normal to the *second* row vector
    - Scaled by the inverse of the length of the *first* row vector



# Matrix Multiplication



- The  $k$ -th axis corresponds to the normal to the hyperplane represented by the  $1..k-1, k+1..N$ -th row vectors in the matrix
  - Any set of  $K-1$  vectors represent a hyperplane of dimension  $K-1$  or less
- The distance along the new axis equals the length of the projection on the  $k$ -th row vector
  - Expressed in inverse-lengths of the vector

# Interpreting matrices

- Matrices as transforms
- Matrices as data containers
- Matrices as compositional building blocks for vector spaces

# Matrices as data containers

- A matrix can be vertical stacking of row vectors

$$\mathbf{R} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$

- The space of all vectors that can be composed from the rows of the matrix is the *row space* of the matrix

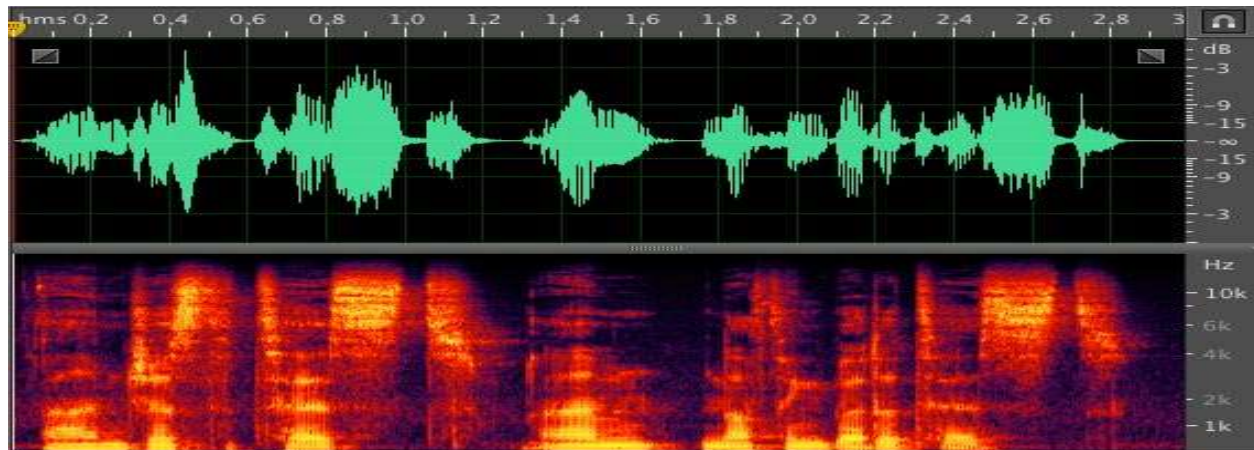
- Or a horizontal arrangement of column vectors

$$\mathbf{R} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$

- The space of all vectors that can be composed from the columns of the matrix is the *column space* of the matrix

# Representing a signal as a matrix

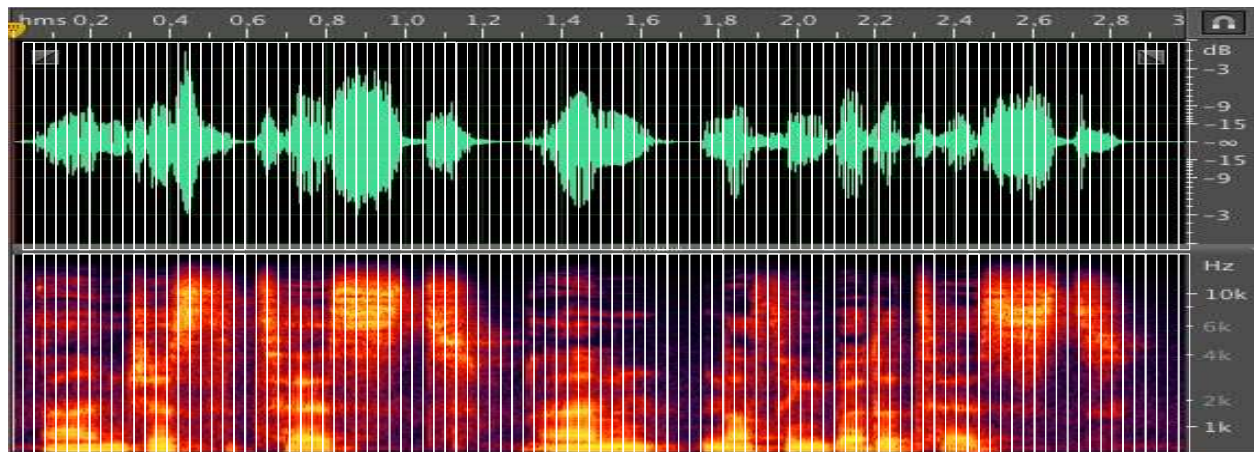
- Time series data like audio signals are often represented as spectrographic matrices



- Each column is the spectrum of a short segment of the audio signal

# Representing a signal as a matrix

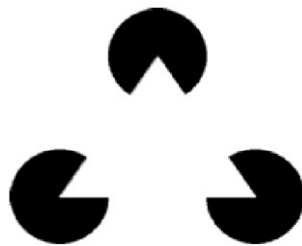
- Time series data like audio signals are often represented as spectrographic matrices



- Each column is the spectrum of a short segment of the audio signal

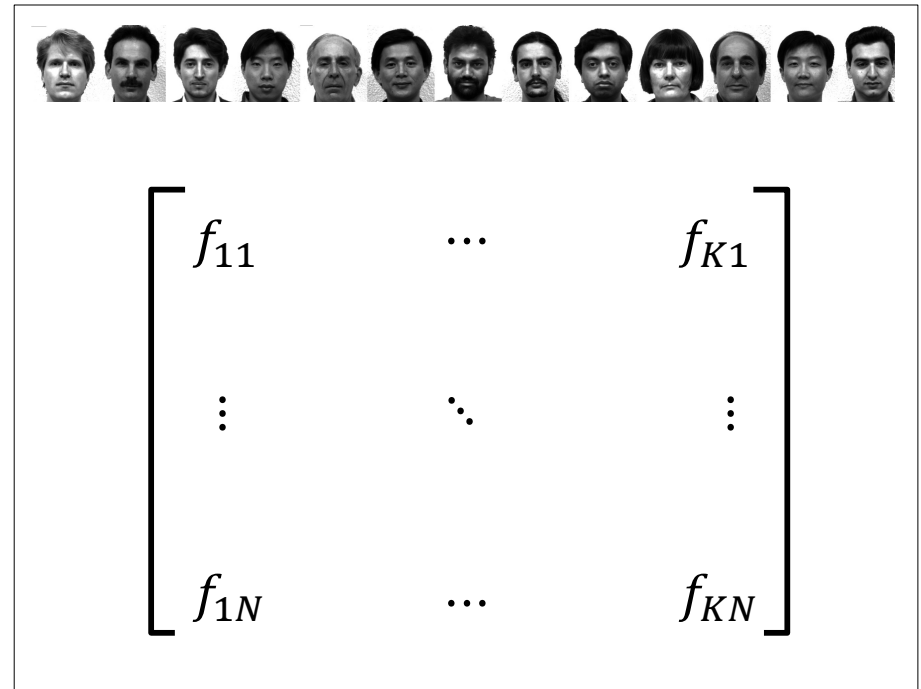
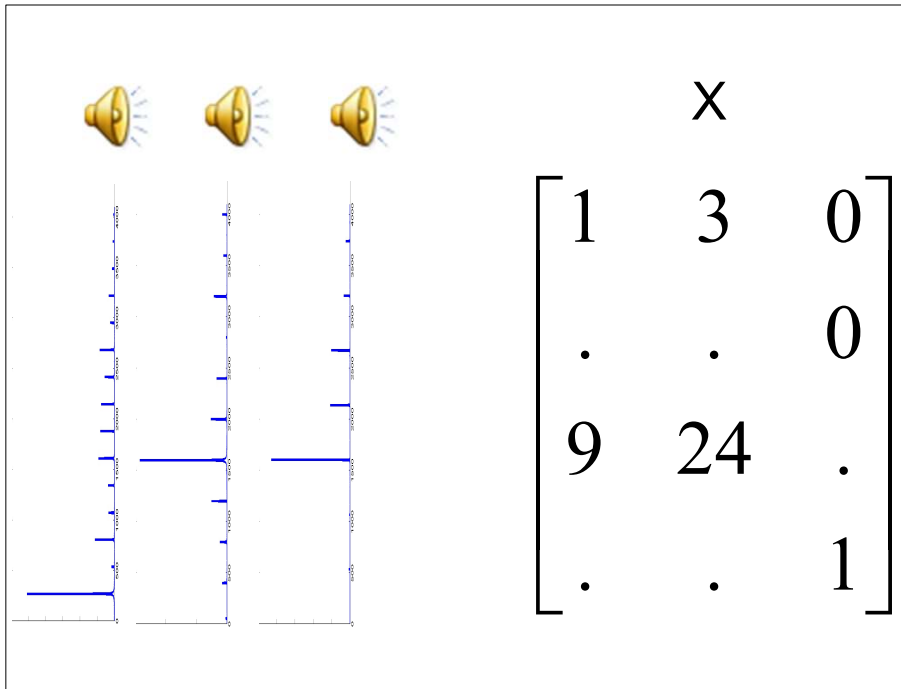
# Representing a signal as a matrix

- Images are often just represented as matrices



```
>> X(1:32:end, 1:40:end)
ans =
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    0    0    0    1    1    1
    1    1    1    1    0    0    0    1    1    1
    1    1    1    1    0    1    0    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    0    1    1    1    1    1    0    1
    1    0    0    1    1    1    1    1    0    0
    1    0    0    0    1    1    1    0    0    0
    1    0    0    0    1    1    1    0    0    0
    1    1    1    1    1    1    1    1    1    1
```

# Storing collections of data



- Individual data instances can be packed into columns (or rows) of a matrix
  - A “data container” matrix

# Interpreting matrices

- Matrices as transforms
- Matrices as data containers
- Matrices as compositional building blocks for vector spaces



# Matrices as space constructors

- Right multiplying a matrix by a column vector mixes the columns of the matrix according to the numbers in the vector

$$- \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

$$\mathbf{Ab} = b_1 \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix} + b_2 \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix} + b_3 \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix} + b_4 \begin{bmatrix} a_{14} \\ a_{24} \\ a_{34} \end{bmatrix}$$

- “Mixes” the columns
  - “Transforms” row space to column space
- “Generates” the space of vectors that can be formed by mixing its own columns

# Multiplying a vector by a matrix

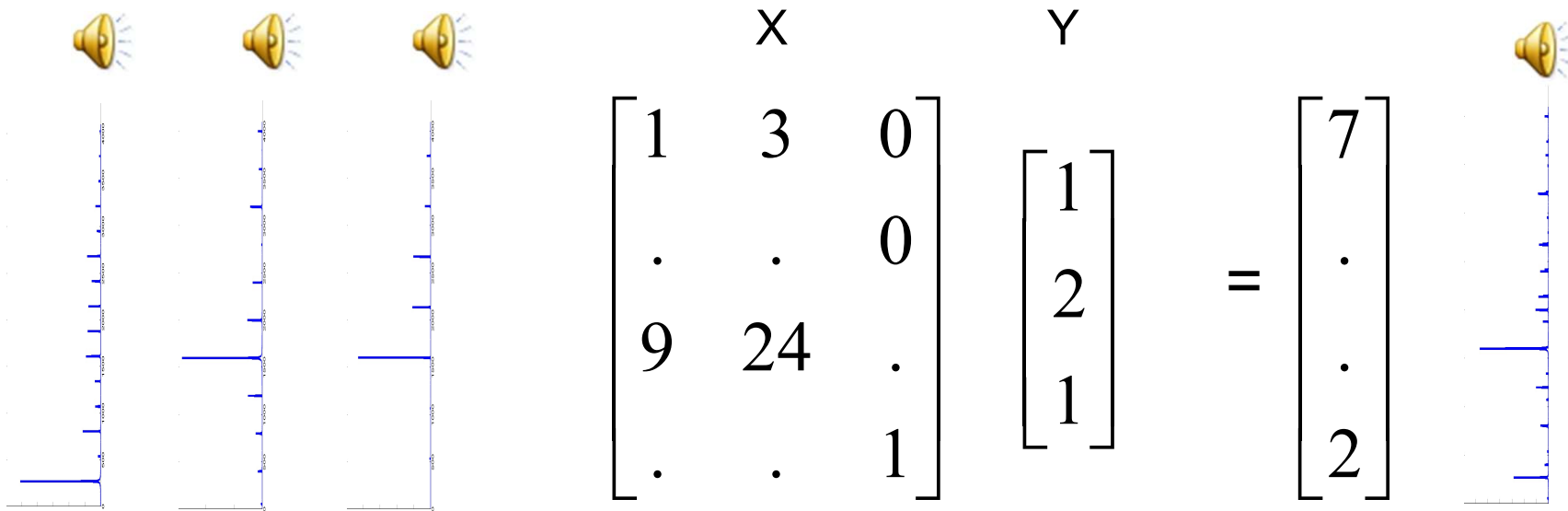
- Left multiplying a matrix by a row vector mixes the rows of the matrix according to the numbers in the vector

$$- \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad \mathbf{b} = [b_1 \quad b_2 \quad b_3]$$

$$\mathbf{bA} = b_1 [a_{11} \quad a_{12} \quad a_{13} \quad a_{14}] + b_2 [a_{21} \quad a_{22} \quad a_{23} \quad a_{24}] \\ + b_3 [a_{31} \quad a_{32} \quad a_{33} \quad a_{34}]$$

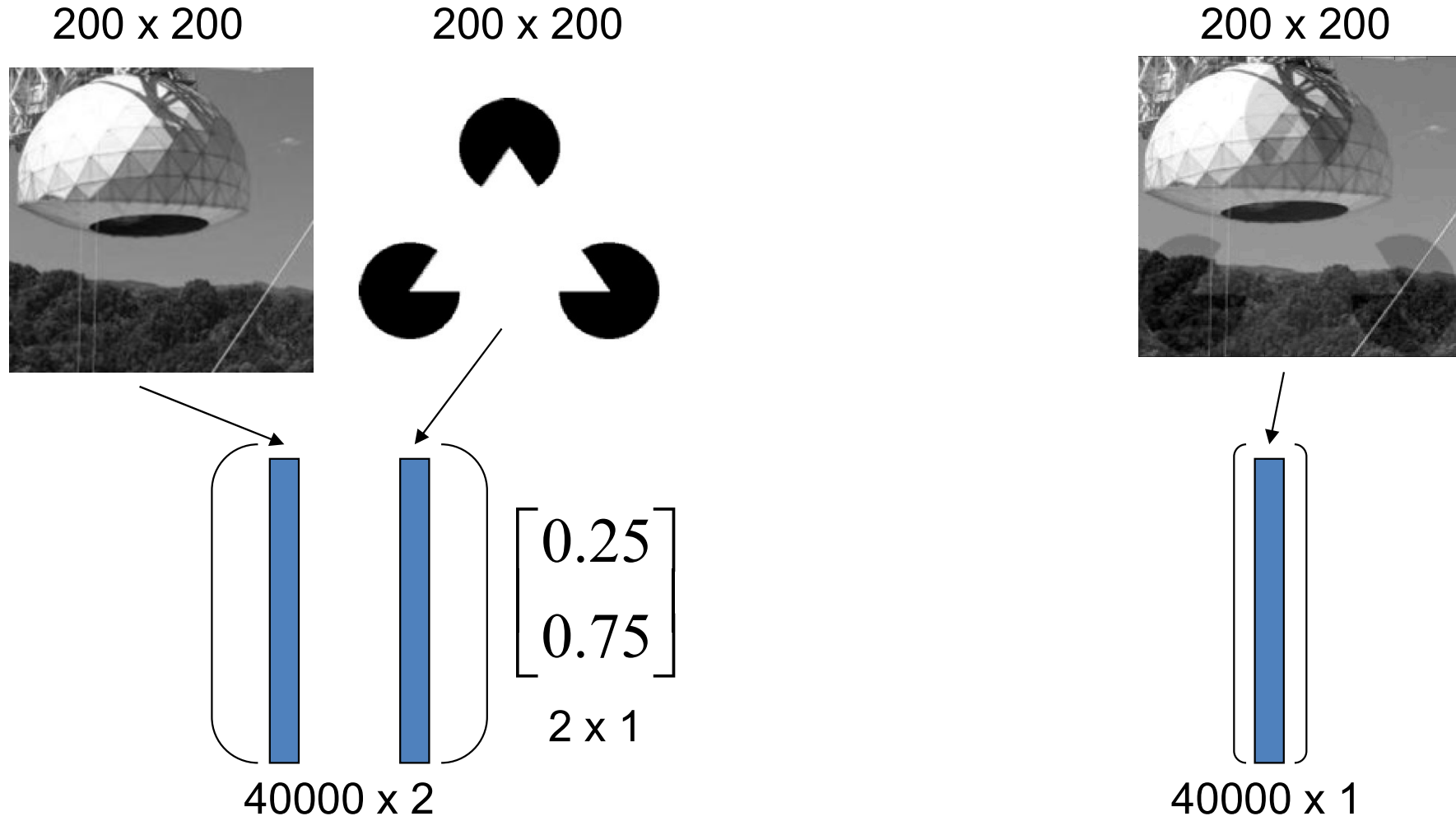
- “Mixes” the rows
  - “Transforms” column space to row space
- “Generates” the space of vectors that can be formed by mixing its own rows

# Matrix multiplication: Mixing vectors



- A physical example
  - The three column vectors of the matrix  $X$  are the spectra of three notes
  - The multiplying column vector  $Y$  is just a mixing vector
  - The result is a sound that is the mixture of the three notes

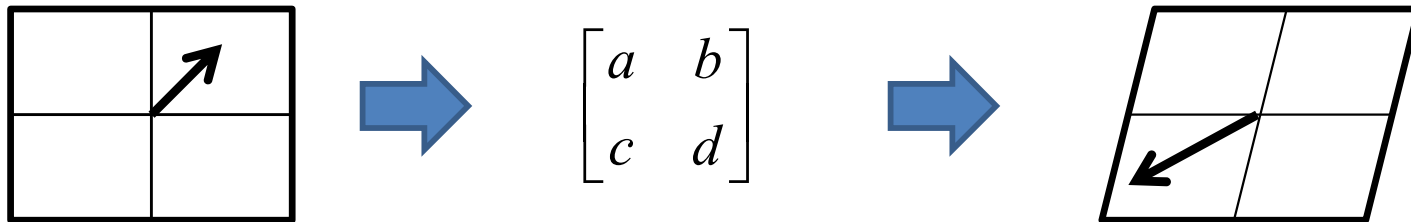
# Matrix multiplication: Mixing vectors



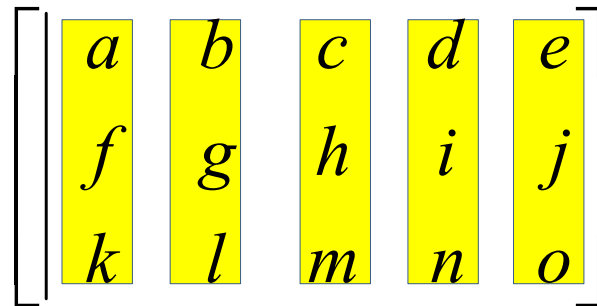
- Mixing two images
  - The images are arranged as columns
    - position value not included
  - The result of the multiplication is rearranged as an image

# Interpretations of a matrix

- As a **transform** that modifies vectors and vector spaces



- As a **container** for data (vectors)



- As a **generator** of vector spaces..

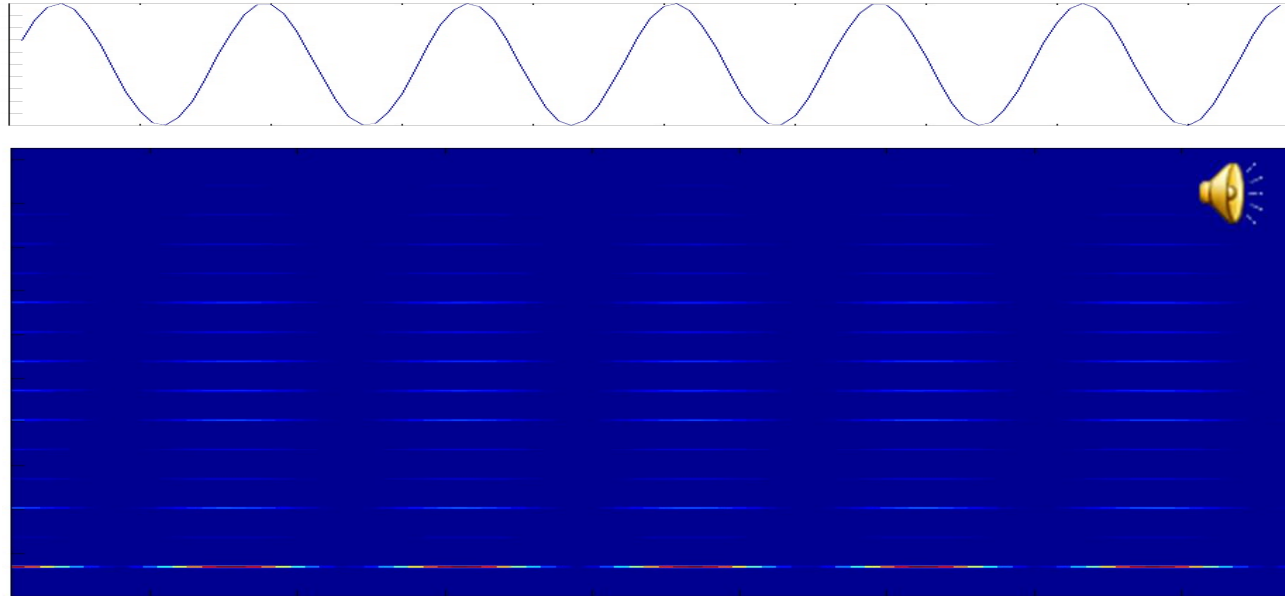
# Matrix ops..

# Vector multiplication: Outer product

- Product of a column vector by a row vector
- Also called vector *direct* product
- Results in a *matrix*
- *Transform or collection of vectors?*

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} d & e & f \end{bmatrix} = \begin{bmatrix} a \cdot d & a \cdot e & a \cdot f \\ b \cdot d & b \cdot e & b \cdot f \\ c \cdot d & c \cdot e & c \cdot f \end{bmatrix}$$

# Vector outer product



- The column vector is the spectrum
- The row vector is an amplitude modulation
- The outer product is a spectrogram
  - Shows how the energy in each frequency varies with time
  - The pattern in each column is a scaled version of the spectrum
  - Each row is a scaled version of the modulation



# Matrix multiplication

$$\begin{bmatrix} a_{11} & \cdot & \cdot & a_{1N} \\ a_{21} & \cdot & \cdot & a_{2N} \\ \cdot & \cdot & \cdot & \cdot \\ a_{M1} & \cdot & \cdot & a_{MN} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & \cdot & b_{1K} \\ \cdot & \cdot & \cdot \\ b_{N1} & \cdot & b_{NK} \end{bmatrix} = \begin{bmatrix} \sum_j a_{1j} b_{j1} & \cdot & \cdot & \sum_j a_{1j} b_{jK} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \sum_j a_{Mj} b_{j1} & \cdot & \cdot & \sum_j a_{Mj} b_{jK} \end{bmatrix}$$

- Standard formula for matrix multiplication

# Matrix multiplication

$$\begin{bmatrix} \mathbf{a}_{11} & \cdot & \cdot & \mathbf{a}_{1N} \\ \mathbf{a}_{21} & \cdot & \cdot & \mathbf{a}_{2N} \\ \cdot & \cdot & \cdot & \cdot \\ \mathbf{a}_{M1} & \cdot & \cdot & \mathbf{a}_{MN} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & \cdot & b_{1K} \\ \cdot & \cdot & \cdot \\ b_{N1} & \cdot & b_{NK} \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \cdot \mathbf{b}_1 & \cdot & \cdot & \mathbf{a}_1 \cdot \mathbf{b}_K \\ \mathbf{a}_2 \cdot \mathbf{b}_1 & \cdot & \cdot & \mathbf{a}_2 \cdot \mathbf{b}_K \\ \cdot & \cdot & \cdot & \cdot \\ \mathbf{a}_M \cdot \mathbf{b}_1 & \cdot & \cdot & \mathbf{a}_M \cdot \mathbf{b}_K \end{bmatrix}$$

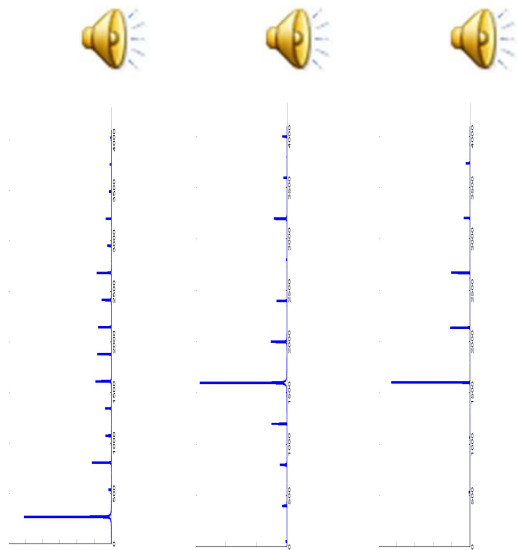
- Matrix  $\mathbf{A}$  : A column of row vectors
- Matrix  $\mathbf{B}$  : A row of column vectors
- $\mathbf{AB}$  : A matrix of inner products
  - Mimics the vector outer product rule

# Matrix multiplication: another view

$$\begin{bmatrix} a_{11} & \cdot & \cdot & a_{1N} \\ a_{21} & \cdot & \cdot & a_{2N} \\ \cdot & \cdot & \cdot & \cdot \\ a_{M1} & \cdot & \cdot & a_{MN} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & \cdot & b_{NK} \\ \cdot & \cdot & \cdot \\ b_{N1} & \cdot & b_{NK} \end{bmatrix} = \begin{bmatrix} a_{11} \\ \cdot \\ \cdot \\ a_{M1} \end{bmatrix} [b_{11} \quad \cdot \quad b_{1K}] + \begin{bmatrix} a_{12} \\ \cdot \\ \cdot \\ a_{M2} \end{bmatrix} [b_{21} \quad \cdot \quad b_{2K}] + \dots + \begin{bmatrix} a_{1N} \\ \cdot \\ \cdot \\ a_{MN} \end{bmatrix} [b_{N1} \quad \cdot \quad b_{NK}]$$

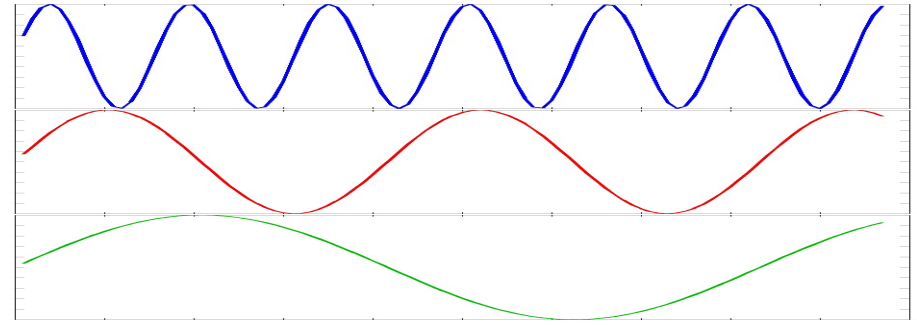
- The outer product of the first column of A and the first row of B + outer product of the second column of A and the second row of B + ....
- *Sum of outer products*

# Why is that useful?



$$X = \begin{bmatrix} 1 & 3 & 0 \\ \cdot & \cdot & 0 \\ 9 & 24 & \cdot \\ \cdot & \cdot & 1 \end{bmatrix}$$

X

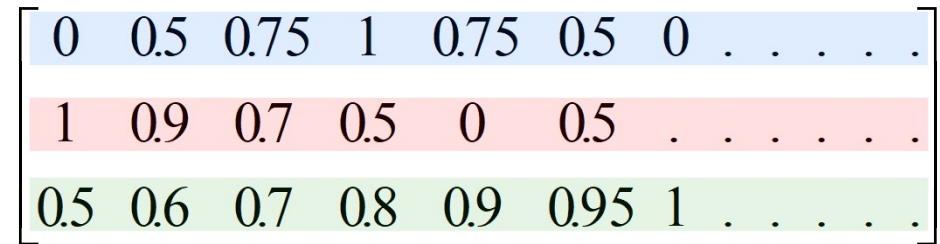
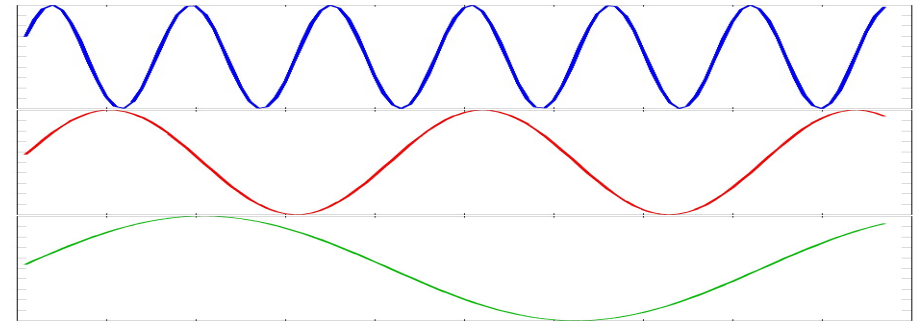
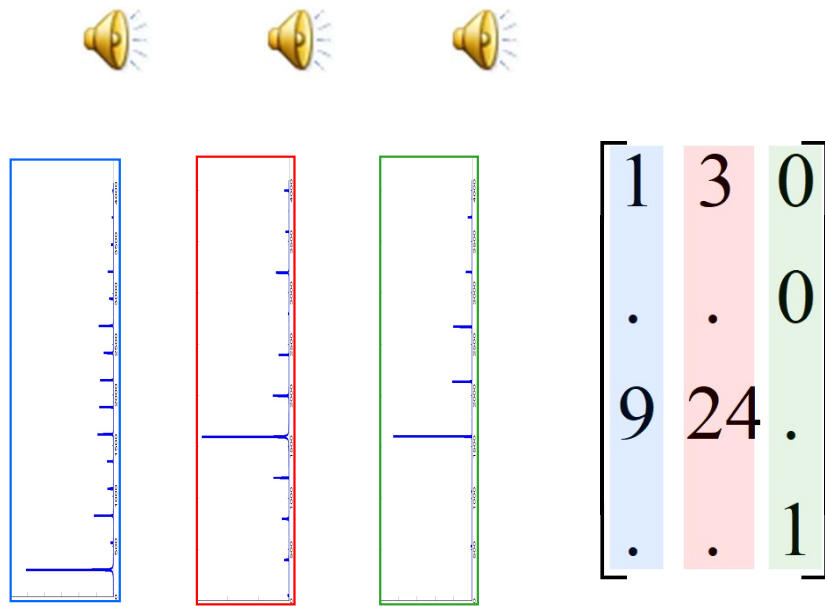


$$Y = \begin{bmatrix} 0 & 0.5 & 0.75 & 1 & 0.75 & 0.5 & 0 & \dots & \dots & \dots \\ 1 & 0.9 & 0.7 & 0.5 & 0 & 0.5 & \dots & \dots & \dots & \dots \\ 0.5 & 0.6 & 0.7 & 0.8 & 0.9 & 0.95 & 1 & \dots & \dots & \dots \end{bmatrix}$$

Y

- Sounds: Three notes modulated independently

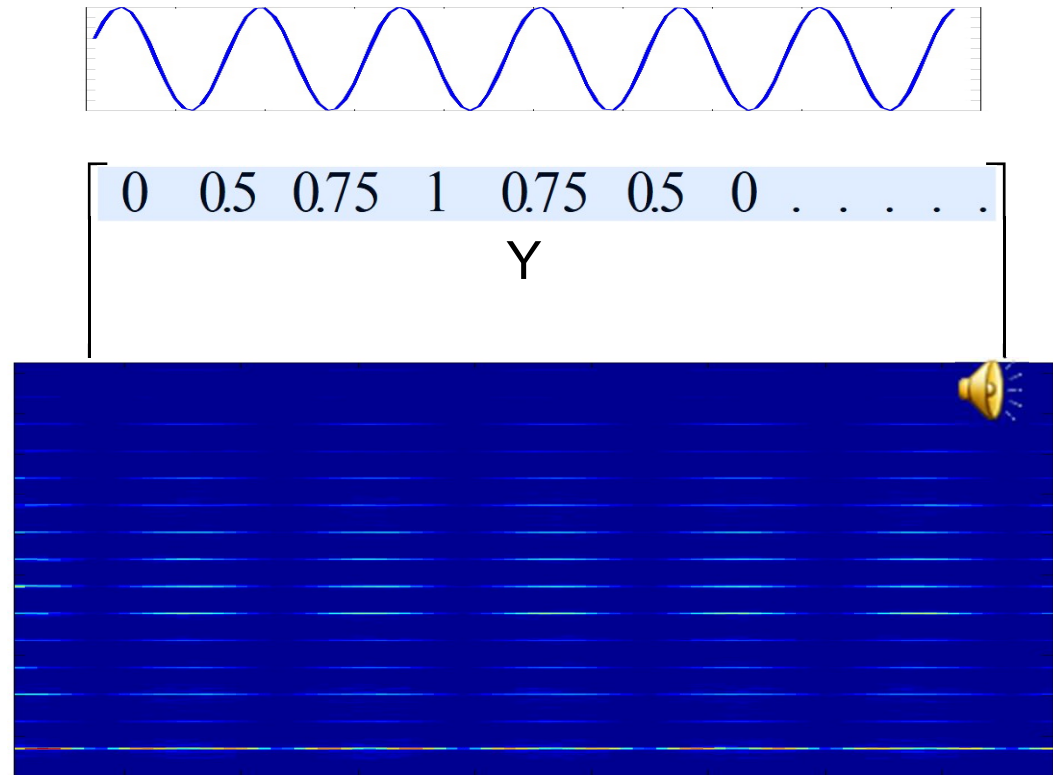
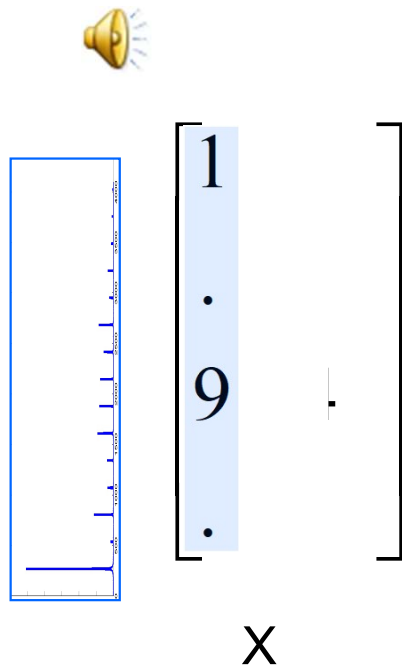
# Matrix multiplication: Mixing modulated spectra



Y

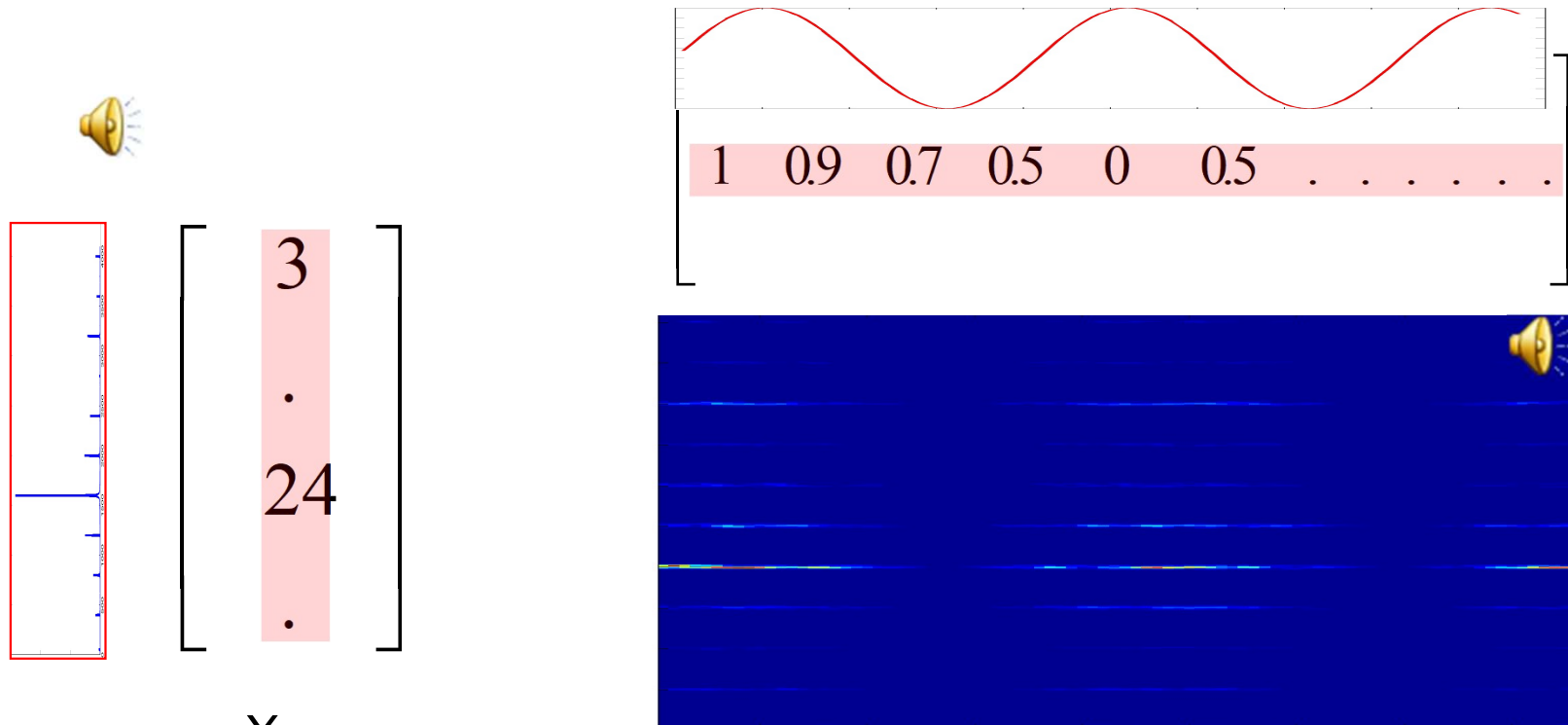
- Sounds: Three notes modulated independently

# Matrix multiplication: Mixing modulated spectra



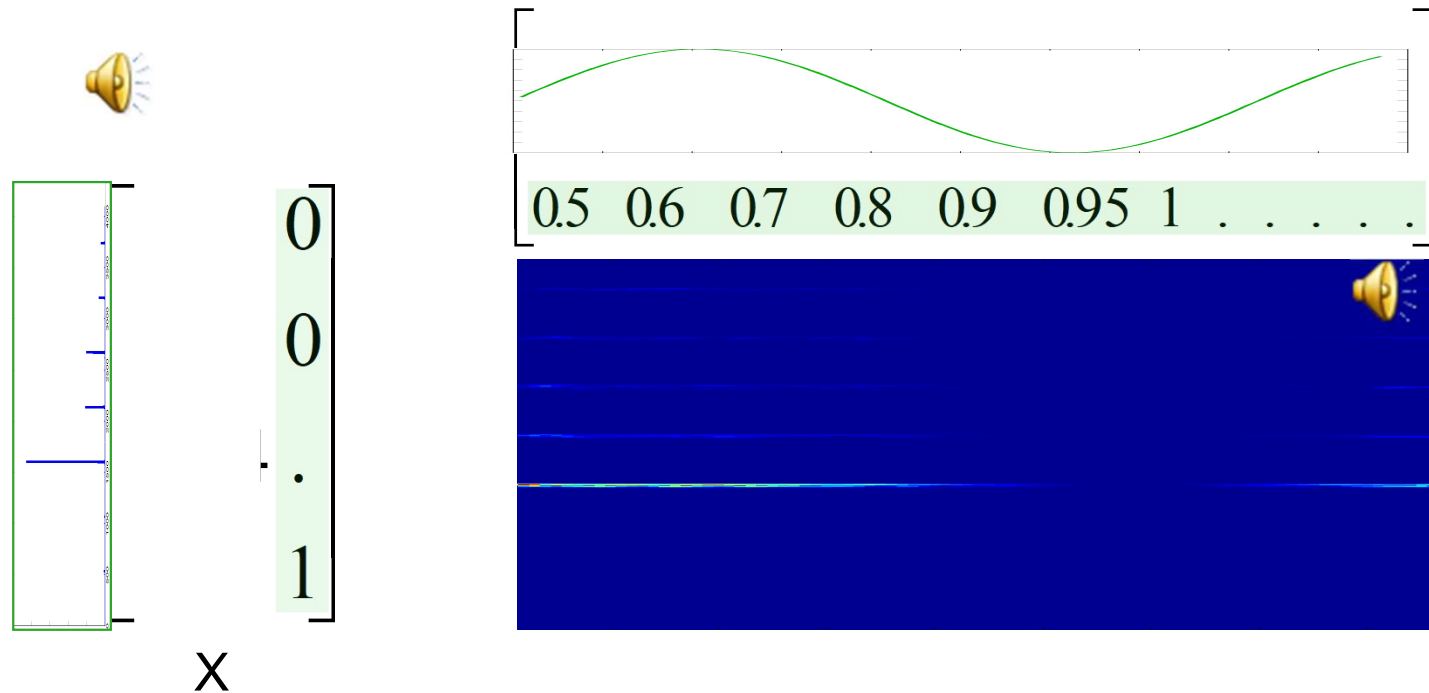
- Sounds: Three notes modulated independently

# Matrix multiplication: Mixing modulated spectra



- Sounds:  $\times$  Three notes modulated independently

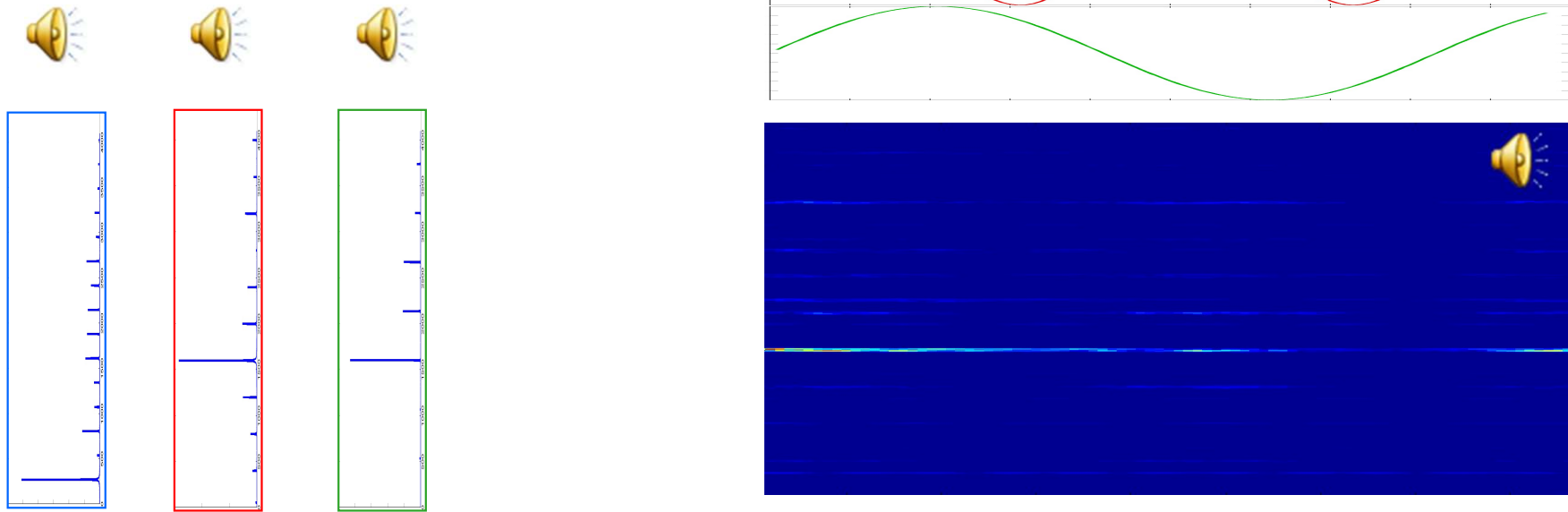
# Matrix multiplication: Mixing modulated spectra



- Sounds: Three notes modulated independently



# Matrix multiplication: Mixing modulated spectra

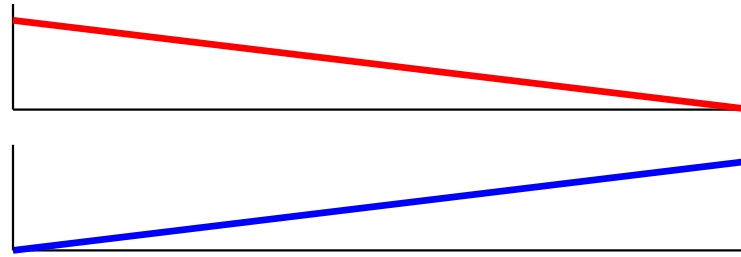


- Sounds: Three notes modulated independently

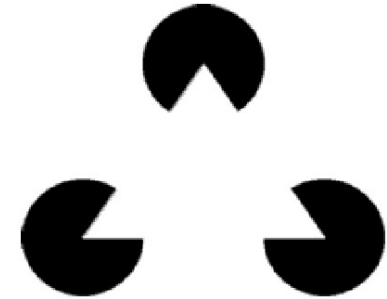
# Matrix multiplication: Image transition



$$\begin{bmatrix} i_1 & j_1 \\ i_2 & j_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}$$



1	.9	.8	.7	.6	.5	.4	.3	.2	.1	0
0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1



- Image1 fades out linearly
- Image 2 fades in linearly

# Matrix multiplication: Image transition



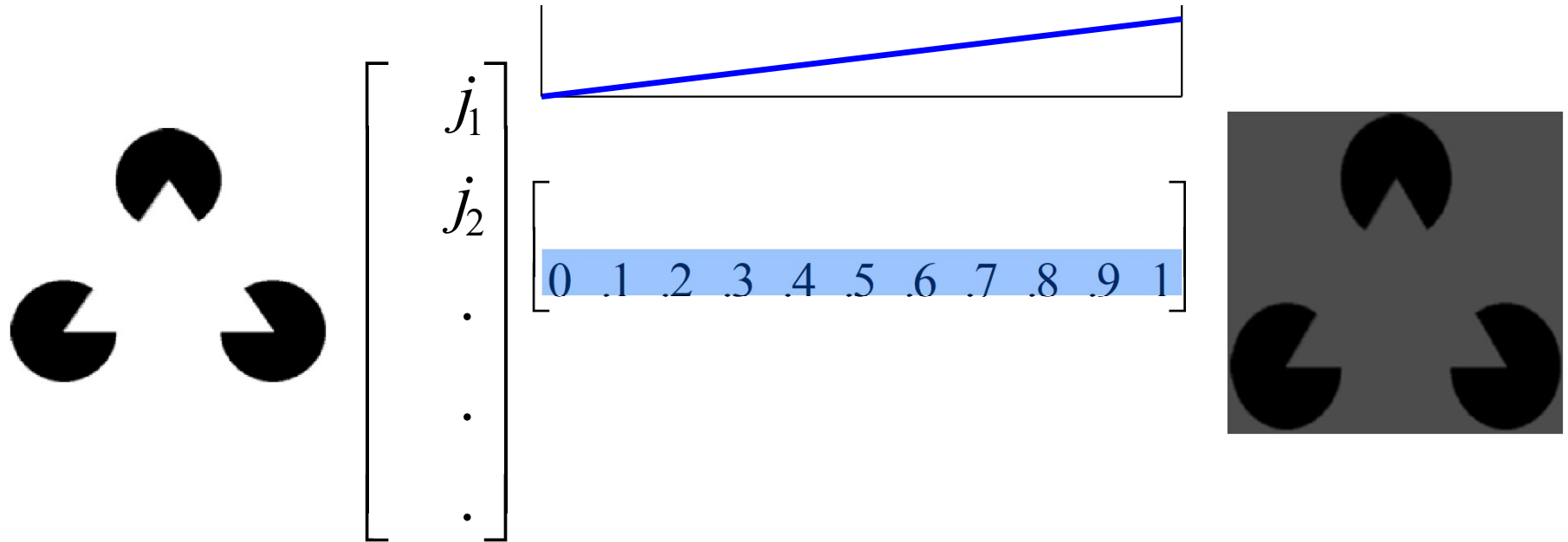
$$\begin{bmatrix} i_1 \\ i_2 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}$$

1	.9	.8	.7	.6	.5	.4	.3	.2	.1	0
$i_1$	$0.9i_1$	$0.8i_1$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0
$i_2$	$0.9i_2$	$0.8i_2$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0
$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0
$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0
$i_N$	$0.9i_N$	$0.8i_N$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0



- Each column is one image
  - The columns represent a sequence of images of decreasing intensity
- Image1 fades out linearly

# Matrix multiplication: Image transition

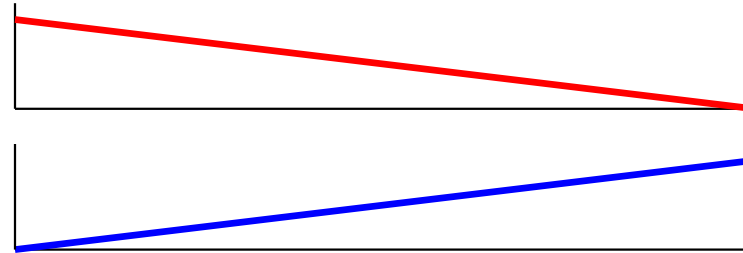


- Image 2 fades in linearly

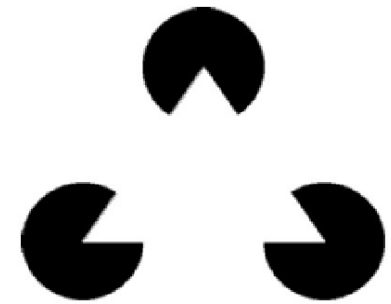
# Matrix multiplication: Image transition



$$\begin{bmatrix} i_1 & j_1 \\ i_2 & j_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}$$



1	.9	.8	.7	.6	.5	.4	.3	.2	.1	0
0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1



- Image1 fades out linearly
- Image 2 fades in linearly

# Matrix Operations: Properties

- $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$ 
  - Actual interpretation: for any vector  $\mathbf{x}$ 
    - $(\mathbf{A} + \mathbf{B})\mathbf{x} = (\mathbf{B} + \mathbf{A})\mathbf{x}$  (column vector  $\mathbf{x}$  of the right size)
    - $\mathbf{x}(\mathbf{A} + \mathbf{B}) = \mathbf{x}(\mathbf{B} + \mathbf{A})$  (row vector  $\mathbf{x}$  of the appropriate size)
- $\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}$

# Multiplication properties

- Properties of vector/matrix products

- Associative

$$\mathbf{A} \cdot (\mathbf{B} \cdot \mathbf{C}) = (\mathbf{A} \cdot \mathbf{B}) \cdot \mathbf{C}$$

- Distributive

$$\mathbf{A} \cdot (\mathbf{B} + \mathbf{C}) = \mathbf{A} \cdot \mathbf{B} + \mathbf{A} \cdot \mathbf{C}$$

- NOT commutative!!!

$$\mathbf{A} \cdot \mathbf{B} \neq \mathbf{B} \cdot \mathbf{A}$$

- *left multiplications  $\neq$  right multiplications*

- Transposition

$$(\mathbf{A} \cdot \mathbf{B})^T = \mathbf{B}^T \cdot \mathbf{A}^T$$

# Poll 2



# Poll 2

- What properties are true for matrix multiplication?
  - **Transposition property**
  - **Distributive property**
  - **Associative property**
  - Commutative property
- True or false:  $(A+B)x = (B+A)x$  for appropriate dimensions of A, B and x
  - **T**
  - F

# The Space of Matrices

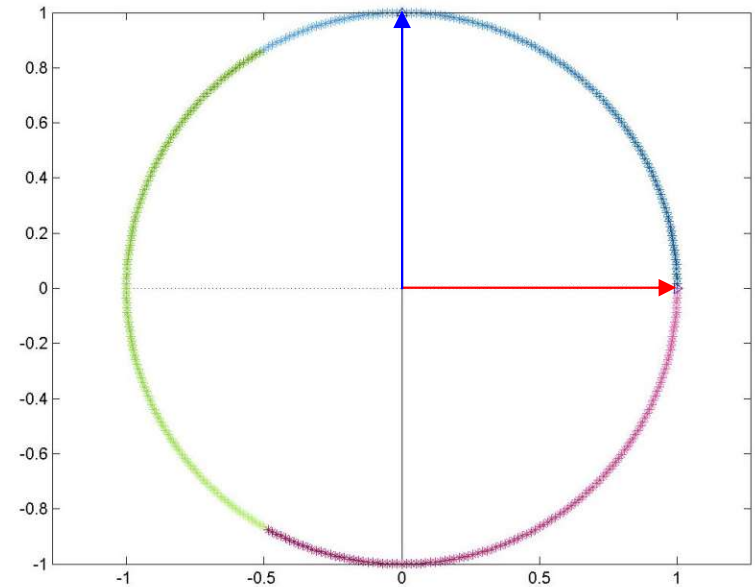
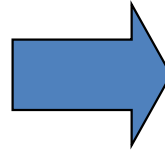
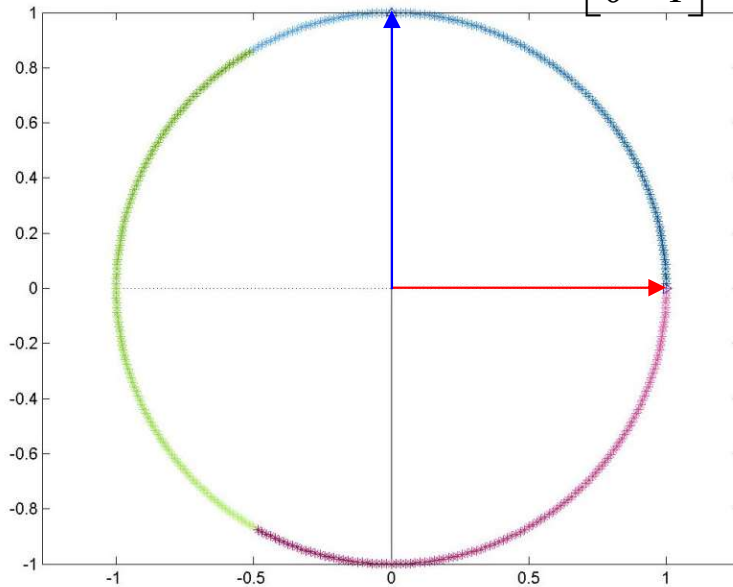
- The set of all matrices of a given size (e.g. all  $3 \times 4$  matrices) is a space!
  - Addition is closed
  - Scalar multiplication is closed
  - Zero matrix exists
  - Matrices have additive inverses
  - Associativity and commutativity rules apply!

# Overview

- Vectors and matrices
- Basic vector/matrix operations
- **Various matrix types**
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- Projections
- Eigen decomposition
- SVD

# The Identity Matrix

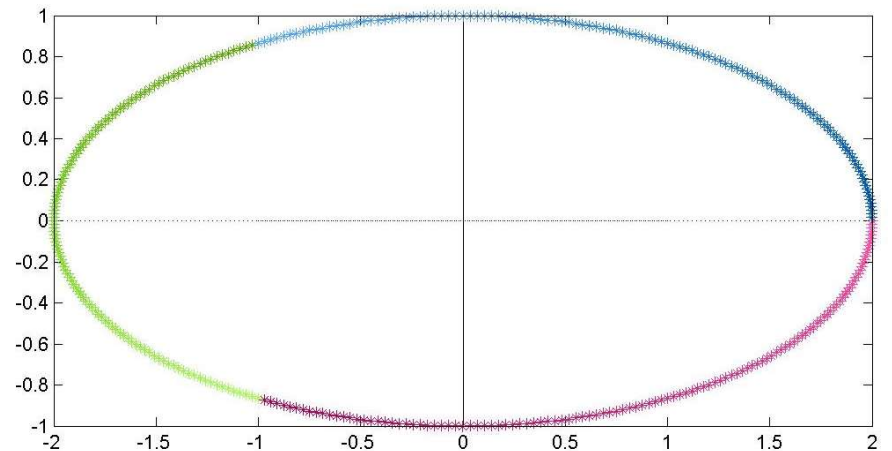
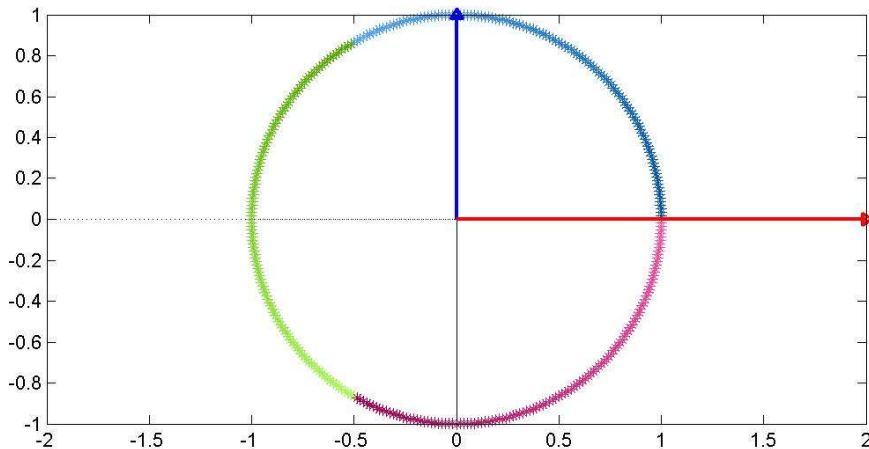
$$Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



- An identity matrix is a square matrix where
  - All diagonal elements are 1.0
  - All off-diagonal elements are 0.0
- Multiplication by an identity matrix does not change vectors

# Diagonal Matrix

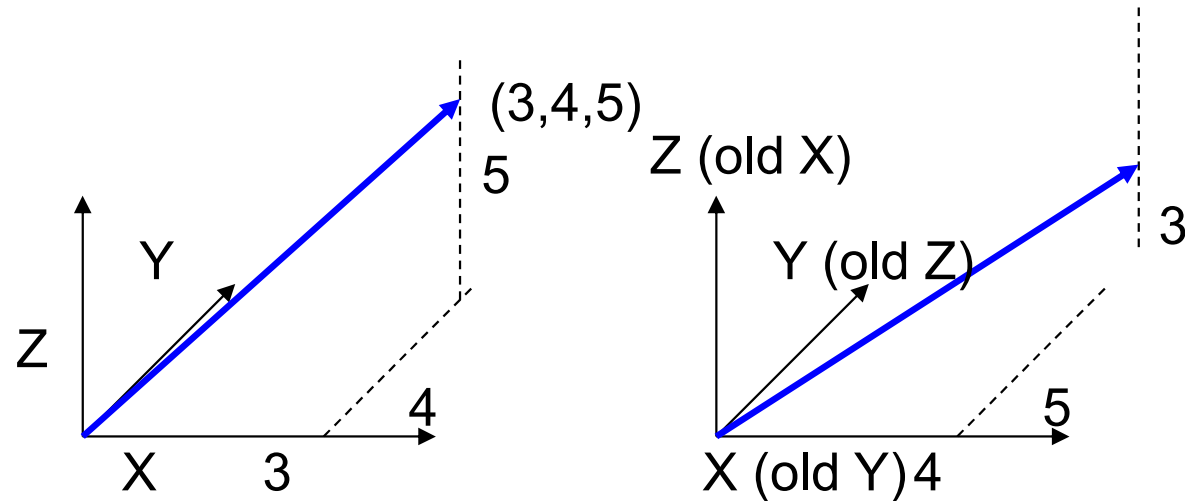
$$Y = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$



- All off-diagonal elements are zero
- Diagonal elements are non-zero
- Scales the axes
  - May flip axes

# Permutation Matrix

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



- A permutation matrix simply rearranges the axes
  - The row entries are axis vectors in a different order
  - The result is a combination of rotations and reflections
- The permutation matrix effectively *permutes* the arrangement of the elements in a vector

# Rotation Matrix

$$x' = x \cos \theta - y \sin \theta$$

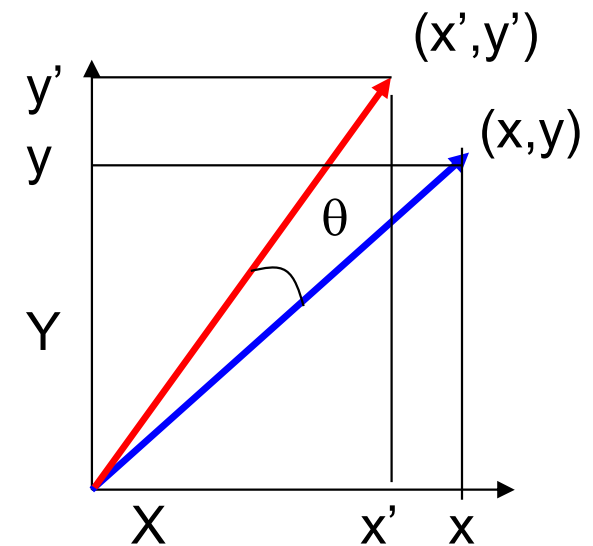
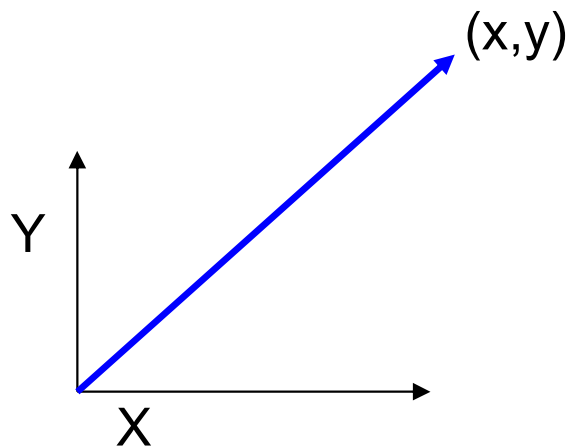
$$y' = x \sin \theta + y \cos \theta$$

$$\mathbf{R}_\theta = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

$$X = \begin{bmatrix} x \\ y \end{bmatrix}$$

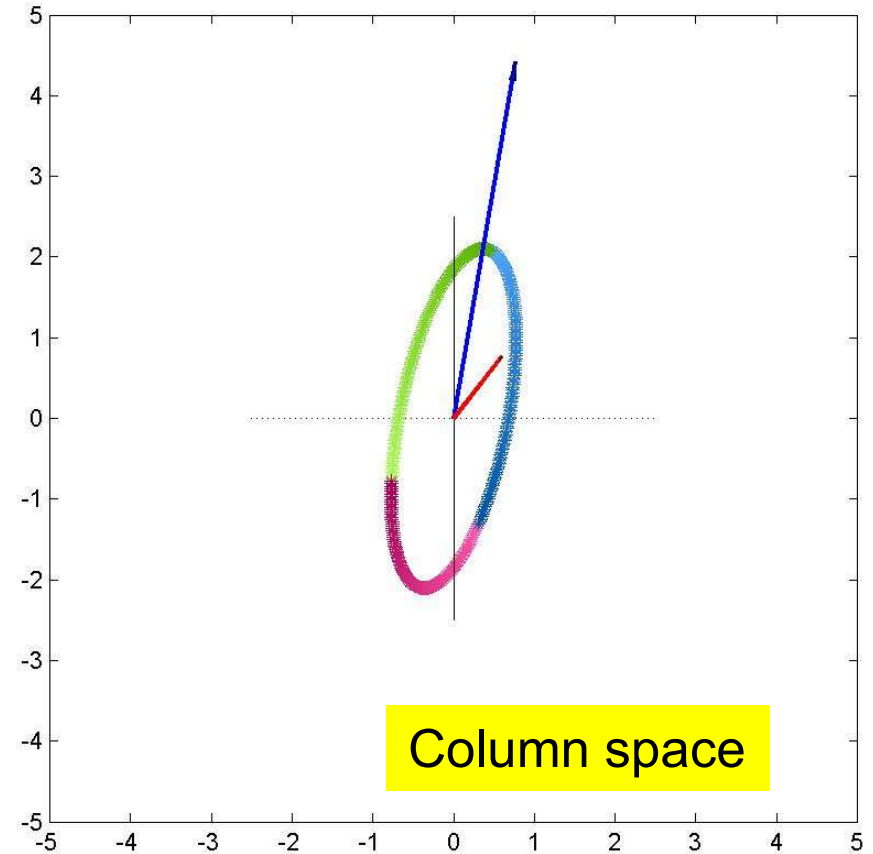
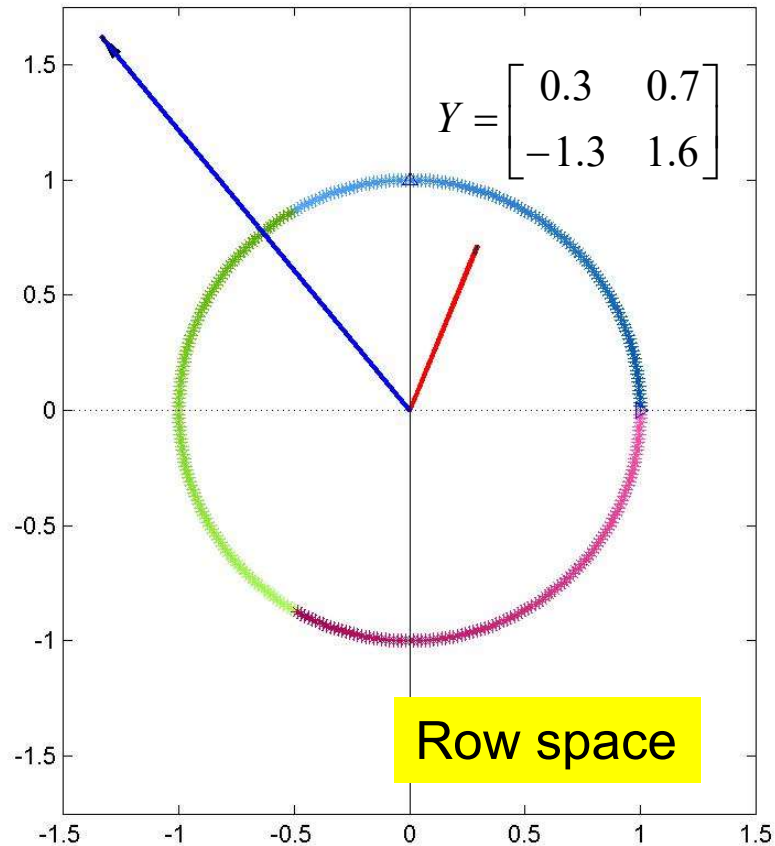
$$X_{new} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$R_\theta X = X_{new}$$



- A rotation matrix *rotates* the vector by some angle  $\theta$
- Alternately viewed, it rotates the axes
  - The new axes are at an angle  $\theta$  to the old one

# More generally



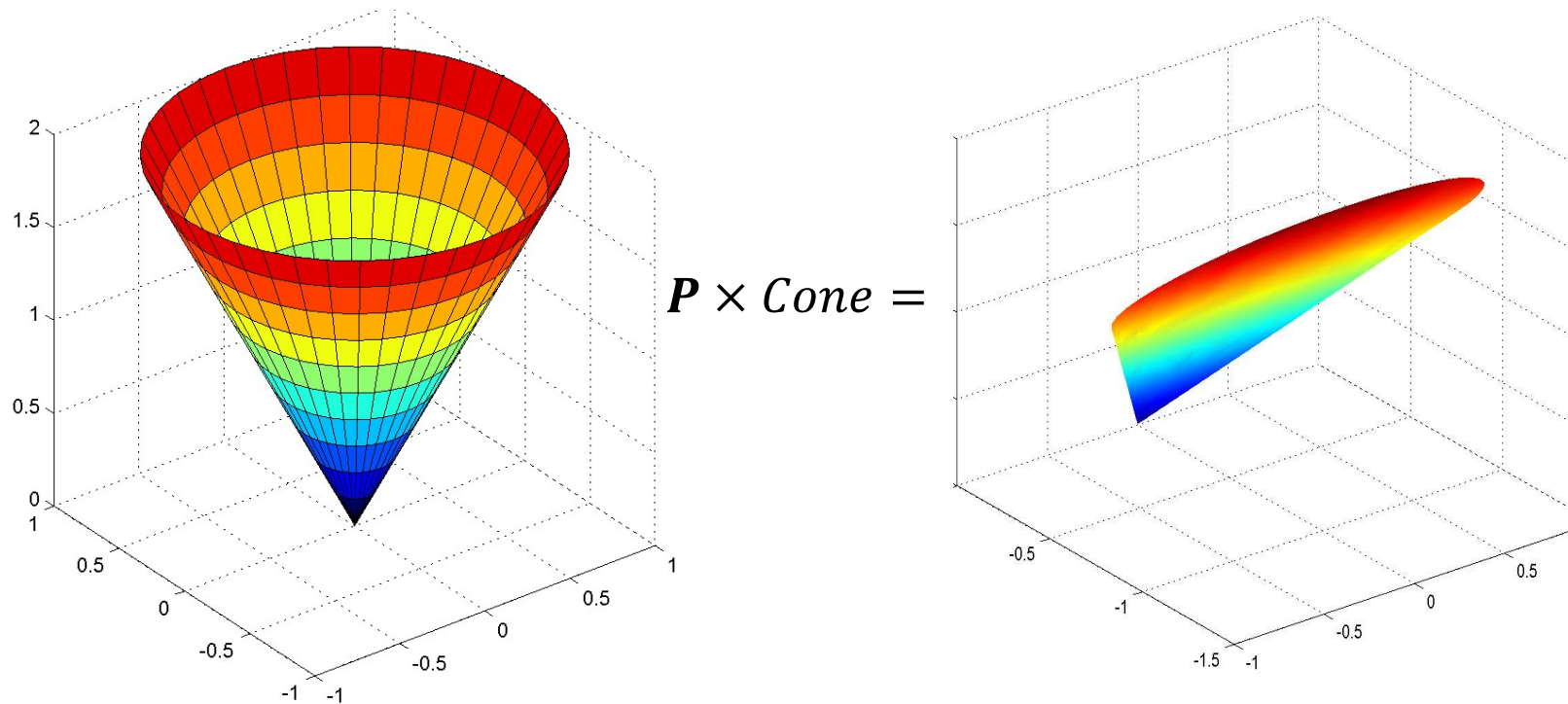
- Matrix operations are combinations of rotations, permutations and stretching



# Overview

- Vectors and matrices
- Basic vector/matrix operations
- Various matrix types
- **Matrix properties**
  - Rank
  - Determinant
  - Inverse
- Solving simultaneous equations
- Projections
- Eigen decomposition
- SVD

# Matrix Rank and Rank-Deficient Matrices

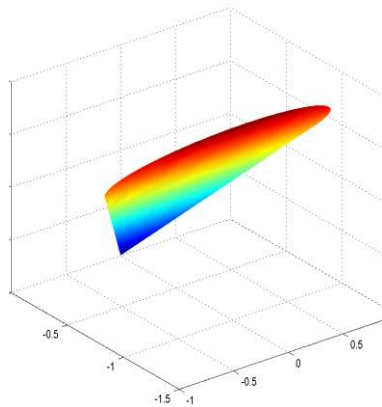


- Some matrices will eliminate one or more dimensions during transformation
  - These are *rank deficient* matrices
  - The **rank** of the matrix is the dimensionality of the transformed version of a **full-dimensional** object

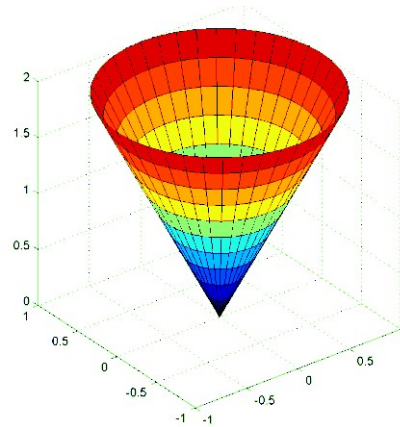
# Matrix Rank and Rank-Deficient Matrices

P =

```
1.0000    0    0
  0    0.2500  -0.4330
  0   -0.4330   0.7500
```

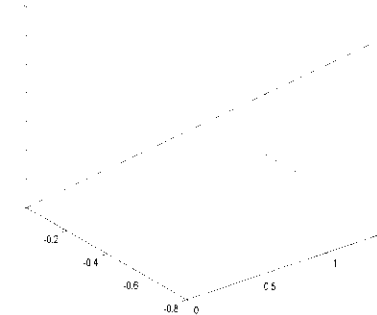


Rank = 2



P2 =

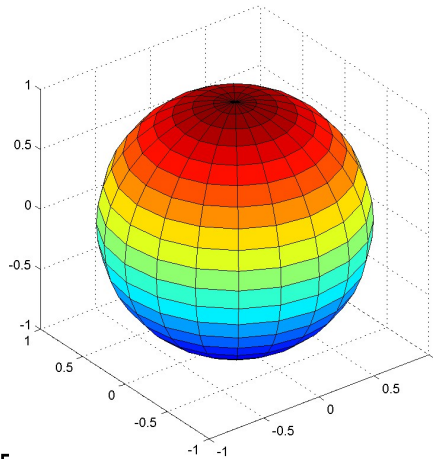
```
0.5000  -0.2500  0.4330
-0.2500   0.1250 -0.2165
 0.4330  -0.2165  0.3750
```



Rank = 1

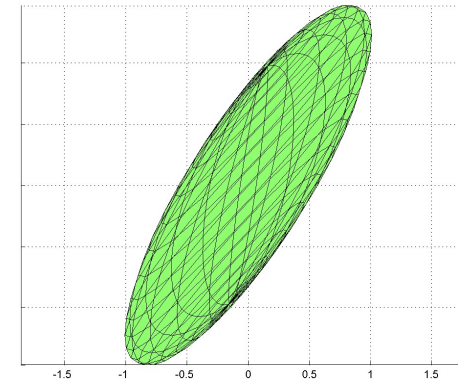
- Some matrices will eliminate one or more dimensions during transformation
  - These are *rank deficient* matrices
  - The rank of the matrix is the dimensionality of the transformed version of a full-dimensional object

# Non-square Matrices



$$\begin{bmatrix} x_1 & x_2 & \cdot & \cdot & x_N \\ y_1 & y_2 & \cdot & \cdot & y_N \\ z_1 & z_2 & \cdot & \cdot & z_N \end{bmatrix}$$

X = 3D data, rank 3



$$\begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \cdot & \cdot & \hat{x}_N \\ \hat{y}_1 & \hat{y}_2 & \cdot & \cdot & \hat{y}_N \end{bmatrix}$$

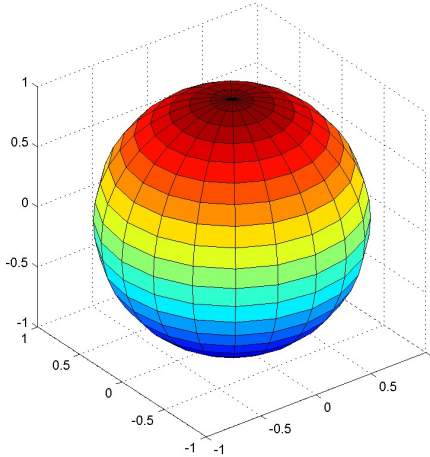
PX = 2D, rank 2

$$\begin{bmatrix} .3 & 1 & .2 \\ .5 & 1 & 1 \end{bmatrix}$$

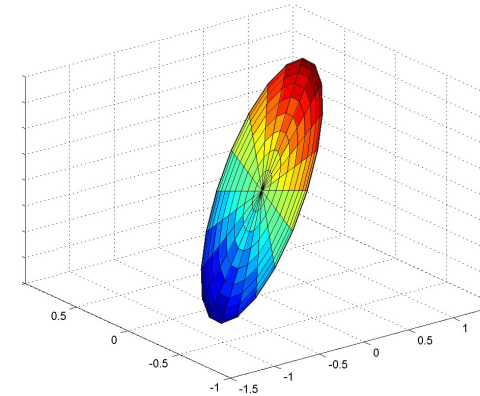
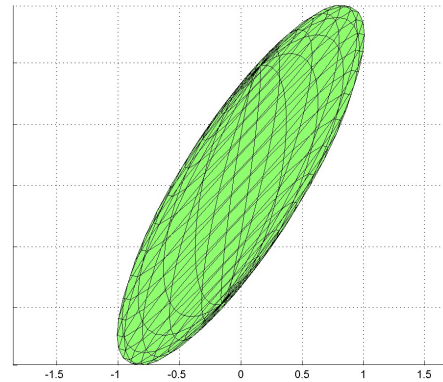
P = transform

- Non-square matrices add or subtract axes
  - More rows than columns  $\rightarrow$  add axes
    - But does not increase the dimensionality of the data
  - Fewer rows than columns  $\rightarrow$  reduce axes
    - May reduce dimensionality of the data

# The Rank of a Matrix



$$\begin{bmatrix} .3 & 1 & .2 \\ .5 & 1 & 1 \end{bmatrix}$$



$$\begin{bmatrix} .8 & .9 \\ .1 & .9 \\ .6 & 0 \end{bmatrix}$$

- The matrix rank is the dimensionality of the transformation of a full-dimensional object in the original space
- The matrix can never *increase* dimensions
  - Cannot convert a circle to a sphere or a line to a circle
- The rank of a matrix can never be greater than the lower of its two dimensions

# Rank – an alternate definition

- In terms of bases..
- Will get back to this shortly..

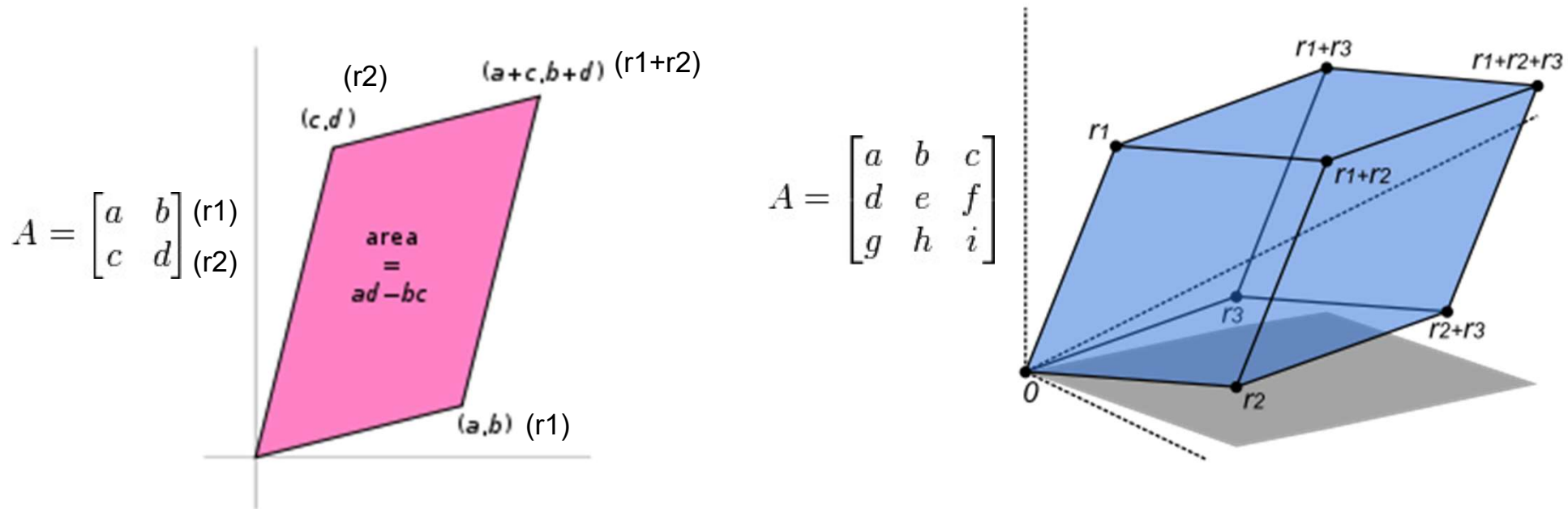
# Poll 3

# Poll 3

- True or false: Given a circle in  $R^3$ , there will always exist a matrix,  $M$ , such that the image of the circle under the transformation of  $M$  will be a sphere.
  - T
  - **F**
- True or false: A square matrix with  $N$  rows and  $N$  columns will always have rank  $N$ .
  - T
  - **F**



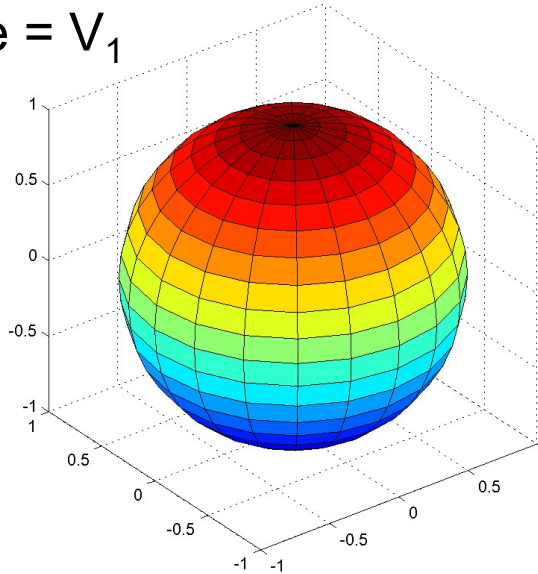
# Matrix Determinant



- The determinant is the “volume” of a matrix
- Actually, the volume of a parallelepiped formed from its row vectors
  - Also, the volume of the parallelepiped formed from its column vectors
- Standard formula for determinant in textbooks

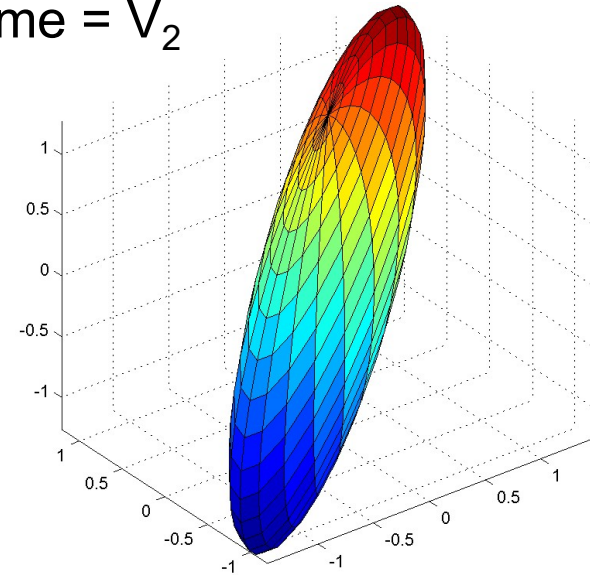
# Matrix Determinant: Another Perspective

Volume =  $V_1$



Volume =  $V_2$

$$\begin{bmatrix} 0.8 & 0 & 0.7 \\ 1.0 & 0.8 & 0.8 \\ 0.7 & 0.9 & 0.7 \end{bmatrix}$$



- The (magnitude of the) determinant is the ratio of N-volumes
  - If  $V_1$  is the volume of an N-dimensional sphere “O” in N-dimensional space
    - O is the complete set of points or vertices that specify the object
  - If  $V_2$  is the volume of the N-dimensional ellipsoid specified by  $A \cdot O$ , where A is a matrix that transforms the space
  - $|A| = V_2 / V_1$

# Matrix Determinants

- Matrix determinants are *only defined for square matrices*
  - They characterize volumes in linearly transformed space of the same dimensionality as the vectors
- Rank deficient matrices have determinant 0
  - Since they compress full-volumed N-dimensional objects into zero-volume N-dimensional objects
    - E.g. a 3-D sphere into a 2-D ellipse: The ellipse has 0 volume (although it does have area)
- Conversely, all matrices of determinant 0 are rank deficient
  - Since they compress full-volumed N-dimensional objects into zero-volume objects

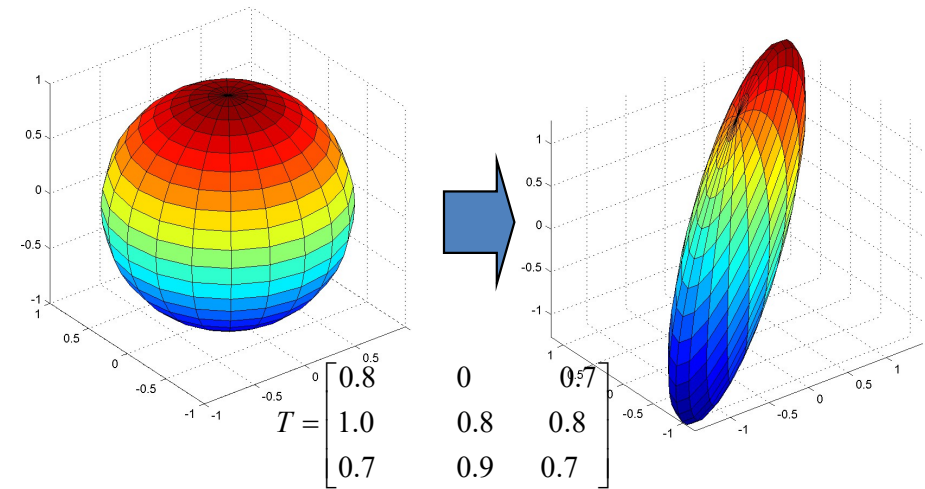
# Determinant properties

- Associative for square matrices  $|\mathbf{A} \cdot \mathbf{B} \cdot \mathbf{C}| = |\mathbf{A}| \cdot |\mathbf{B}| \cdot |\mathbf{C}|$ 
  - Scaling volume sequentially by several matrices is equal to scaling once by the product of the matrices
- Volume of sum  $\neq$  sum of Volumes  $|(\mathbf{B} + \mathbf{C})| \neq |\mathbf{B}| + |\mathbf{C}|$
- Commutative
  - The order in which you scale the volume of an object is irrelevant

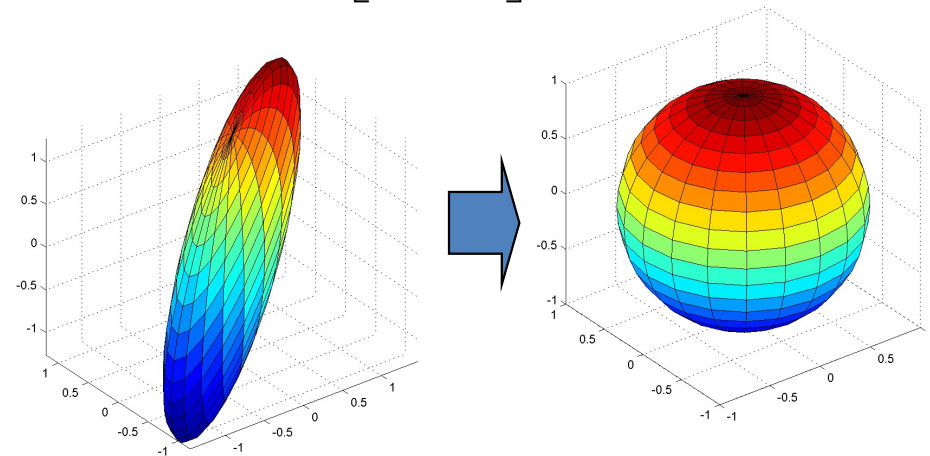
$$|\mathbf{A} \cdot \mathbf{B}| = |\mathbf{B} \cdot \mathbf{A}| = |\mathbf{A}| \cdot |\mathbf{B}|$$

# Matrix Inversion

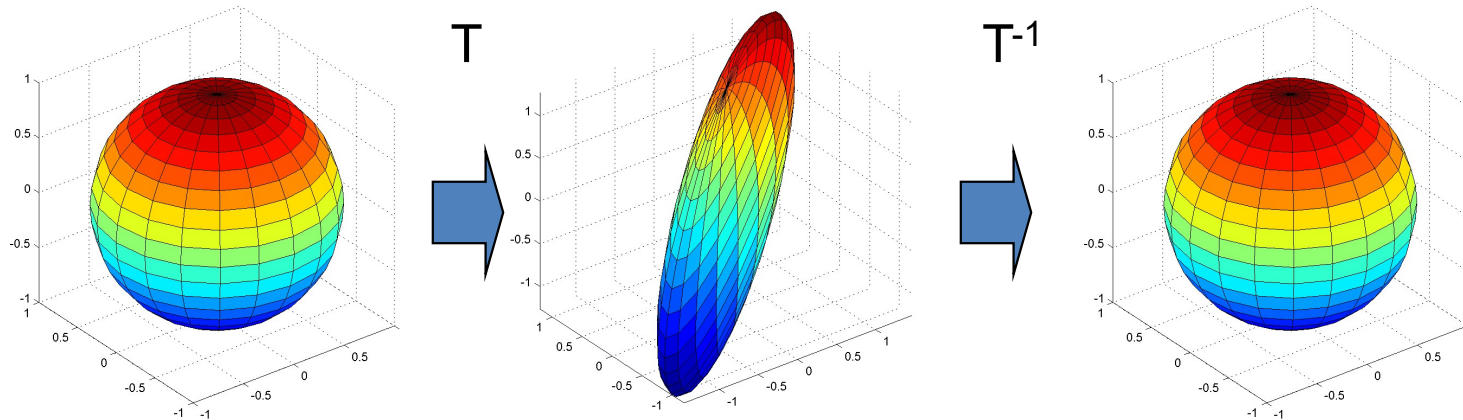
- A matrix transforms an N-dimensional object to a different N-dimensional object
- What transforms the new object back to the original?
  - The *inverse transformation*
- The inverse transformation is called the matrix inverse



$$Q = \begin{bmatrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{bmatrix} = T^{-1}$$



# Matrix Inversion

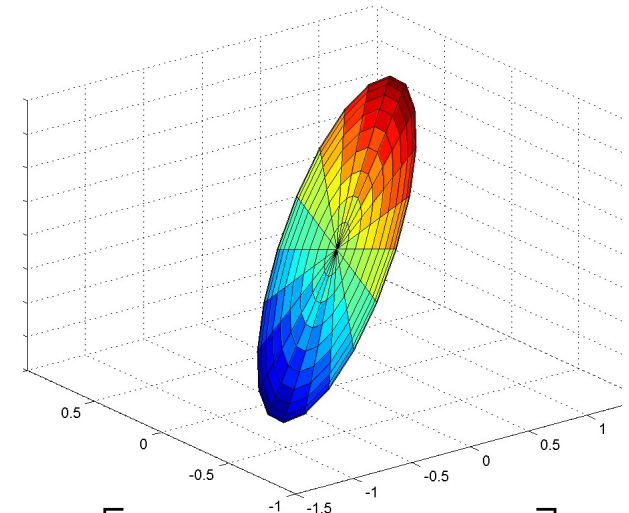
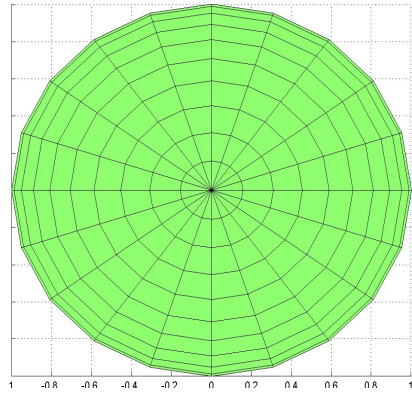


$$\mathbf{T}^{-1}\mathbf{T}\mathbf{D} = \mathbf{D} \Rightarrow \mathbf{T}^{-1}\mathbf{T} = \mathbf{I}$$

- The product of a matrix and its inverse is the identity matrix
  - Transforming an object, and then inverse transforming it gives us back the original object

$$\mathbf{T}\mathbf{T}^{-1}\mathbf{D} = \mathbf{D} \Rightarrow \mathbf{T}\mathbf{T}^{-1} = \mathbf{I}$$

# Non-square Matrices



$$\begin{bmatrix} x_1 & x_2 & \cdot & \cdot & x_N \\ y_1 & y_2 & \cdot & \cdot & y_N \end{bmatrix}$$

X = 2D data

$$\begin{bmatrix} .8 & .9 \\ .1 & .9 \\ .6 & 0 \end{bmatrix}$$

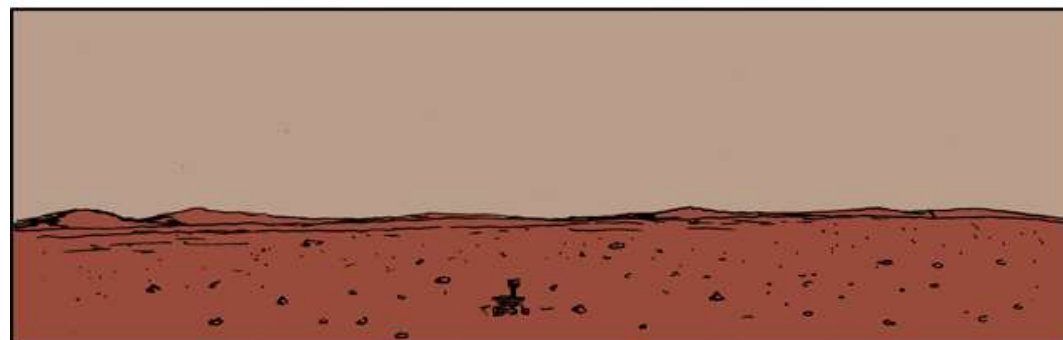
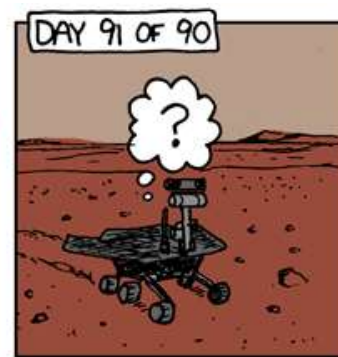
P = transform

$$\begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \cdot & \cdot & \hat{x}_N \\ \hat{y}_1 & \hat{y}_2 & \cdot & \cdot & \hat{y}_N \\ \hat{z}_1 & \hat{z}_2 & \cdot & \cdot & \hat{z}_N \end{bmatrix}$$

PX = 3D, rank 2

- Non-square matrices add or subtract axes
  - More rows than columns  $\rightarrow$  add axes
    - But does not increase the dimensionality of the data

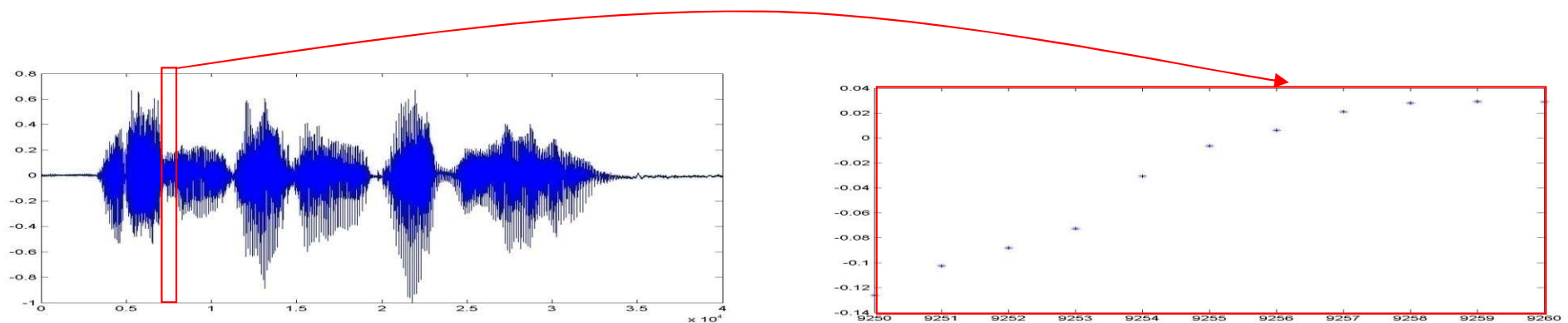






# Recap: Representing signals as vectors

- Signals are frequently represented as vectors for manipulation
- E.g. A segment of an audio signal

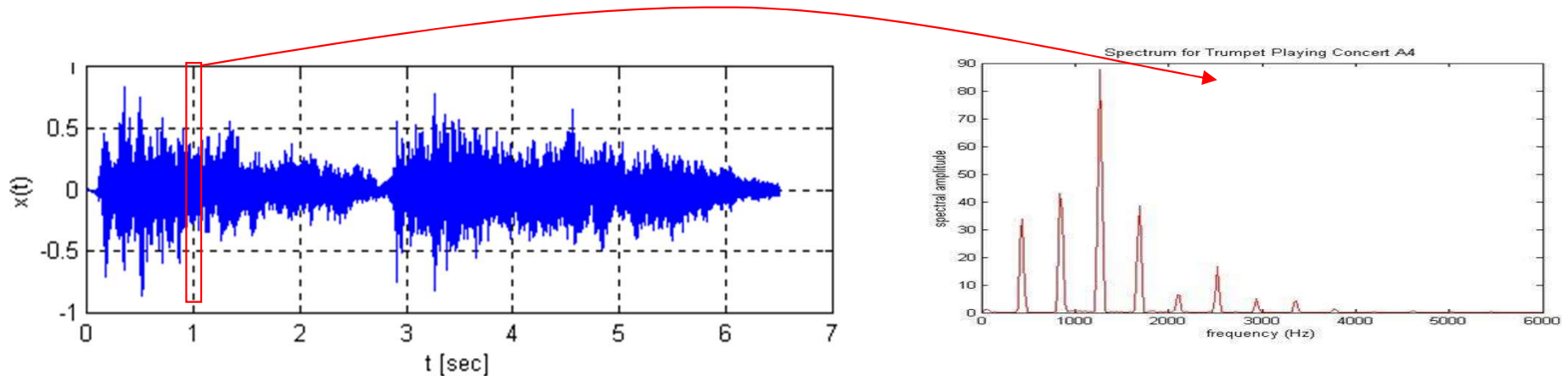


- Represented as a vector of sample values

$$[s_1 \ s_2 \ s_3 \ s_4 \ \dots \ s_N]$$

# Representing signals as vectors

- Signals are frequently represented as vectors for manipulation
- E.g. The *spectrum* segment of an audio signal



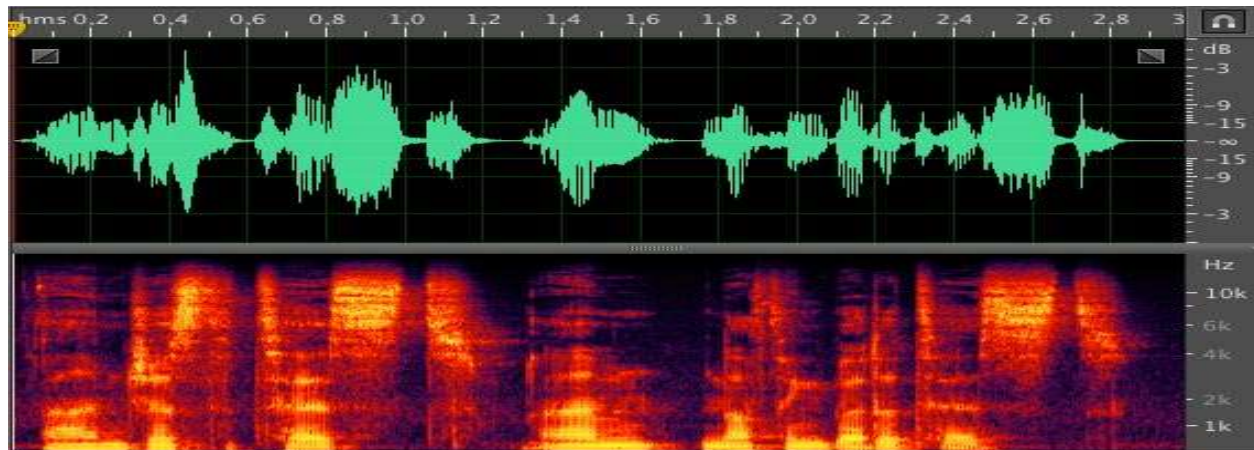
- Represented as a vector of sample values

$$[S_1 \ S_2 \ S_3 \ S_4 \ \dots \ S_M]$$

- Each component of the vector represents a frequency component of the spectrum

# Representing a signal as a matrix

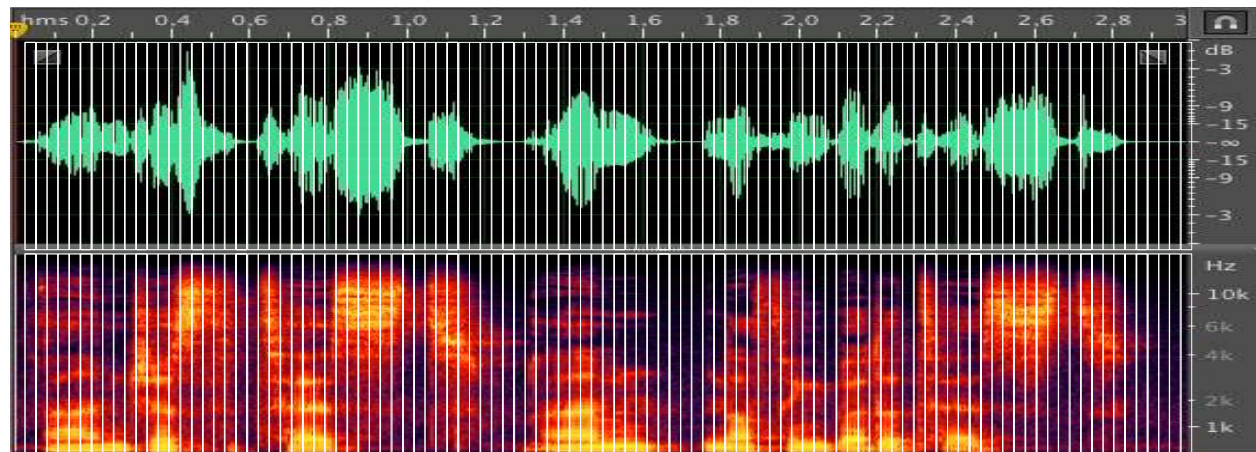
- Time series data like audio signals are often represented as spectrographic matrices



- Each column is the spectrum of a short segment of the audio signal

# Representing a signal as a matrix

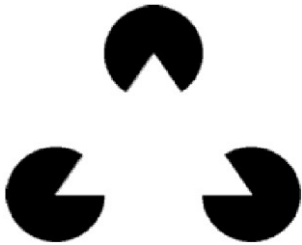
- Time series data like audio signals are often represented as spectrographic matrices



- Each column is the spectrum of a short segment of the audio signal

# Representing an image as a vector

- 3 pacmen
- A 321 x 399 grid of pixel values
  - Row and Column = position
- A 1 x 128079 vector
  - “Unraveling” the matrix

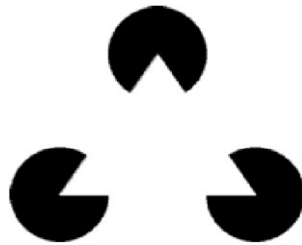


$$[1 \ 1 \ . \ 1 \ 1 \ . \ 0 \ 0 \ 0 \ . \ . \ 1]$$

- Note: This can be recast as the grid that forms the image

# Representing a signal as a matrix

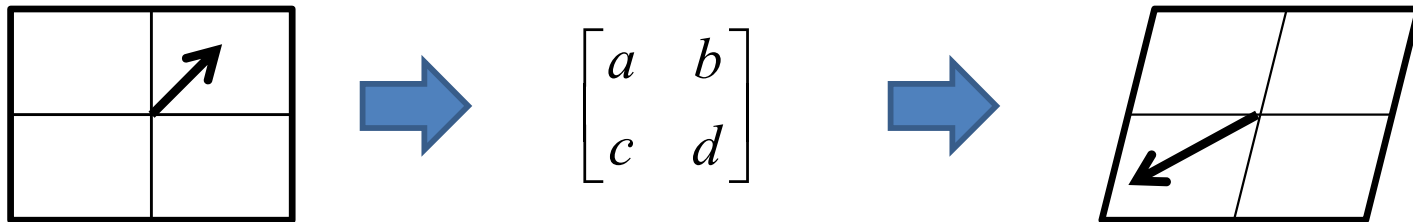
- Images are often just represented as matrices



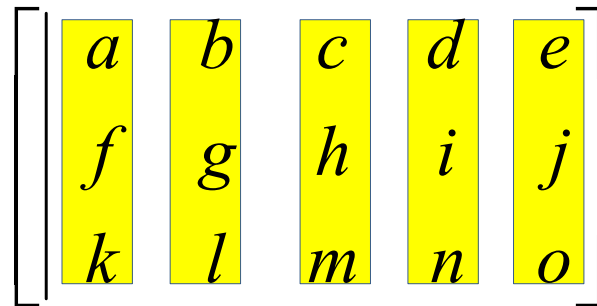
```
>> X(1:32:end,1:40:end)
ans =
  1  1  1  1  1  1  1  1  1  1
  1  1  1  1  0  0  0  1  1  1
  1  1  1  1  0  0  0  1  1  1
  1  1  1  1  0  1  0  1  1  1
  1  1  1  1  1  1  1  1  1  1
  1  1  1  1  1  1  1  1  1  1
  1  1  0  1  1  1  1  1  0  1
  1  0  0  1  1  1  1  1  0  0
  1  0  0  0  1  1  1  0  0  0
  1  0  0  0  1  1  1  0  0  0
  1  1  1  1  1  1  1  1  1  1
```

# Interpretations of a matrix

- As a **transform** that modifies vectors and vector spaces

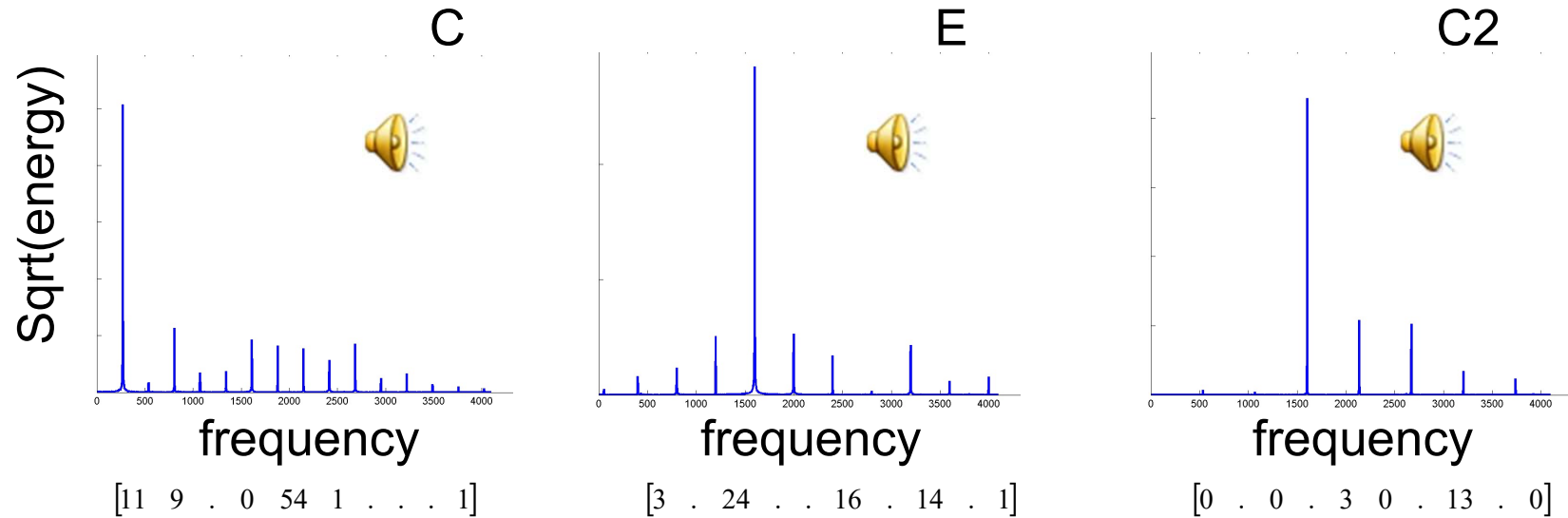


- As a **container** for data (vectors)



- As a generator of vector spaces..

# Revise.. Vector dot product



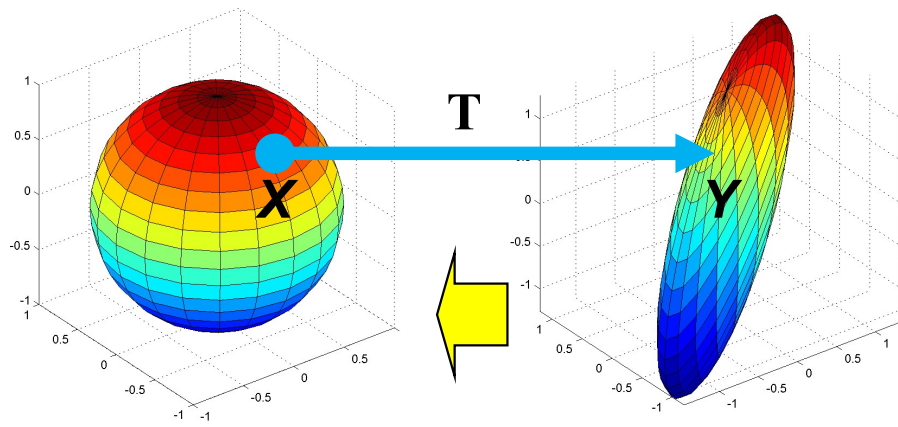
- How much of C is also in E
  - How much can you fake a C by playing an E
  - $C.E / |C| |E| = 0.1$
  - Not very much
- How much of C is in C2?
  - $C.C2 / |C| / |C2| = 0.5$
  - Not bad, you can fake it



# Overview

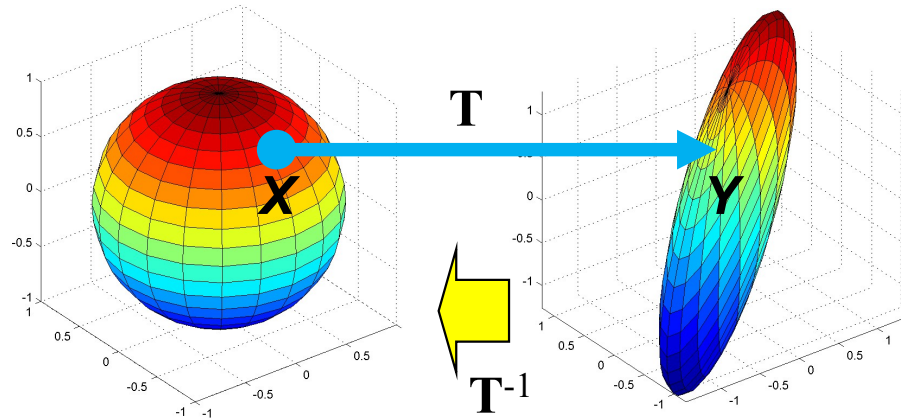
- Vectors and matrices
- Basic vector/matrix operations
- Various matrix types
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- **Solving simultaneous equations**
- Projections
- Eigen decomposition
- SVD

# The Inverse Transform and Simultaneous Equations



- Given the Transform  $T$  and transformed vector  $Y$ , how do we determine  $X$ ?

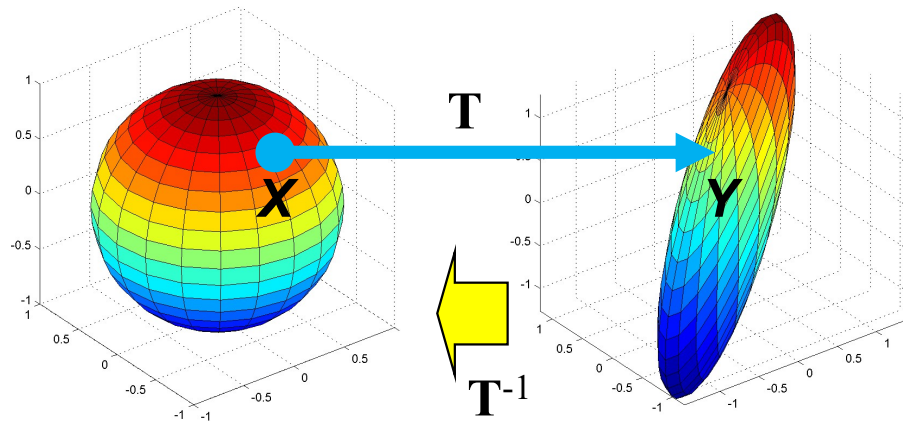
# Matrix inversion (division)



- The inverse of matrix multiplication
  - Not element-wise division!!
  - E.g.

$$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}^{-1} = \begin{bmatrix} 3/4 & -1/4 & -1/4 \\ -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & 3/4 \end{bmatrix}$$

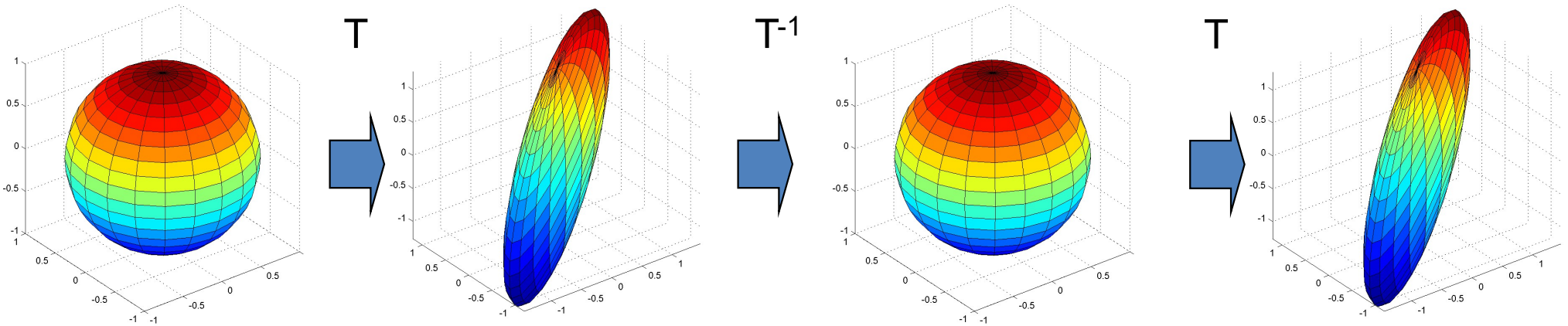
# Matrix inversion (division)



- Provides a way to “undo” a linear transform
- Undoing a transform must happen as soon as it is performed
- Effect on matrix inversion: Note order of multiplication

$$\mathbf{A} \cdot \mathbf{B} = \mathbf{C}, \quad \mathbf{A} = \mathbf{C} \cdot \mathbf{B}^{-1}, \quad \mathbf{B} = \mathbf{A}^{-1} \cdot \mathbf{C}$$

# Matrix Inversion

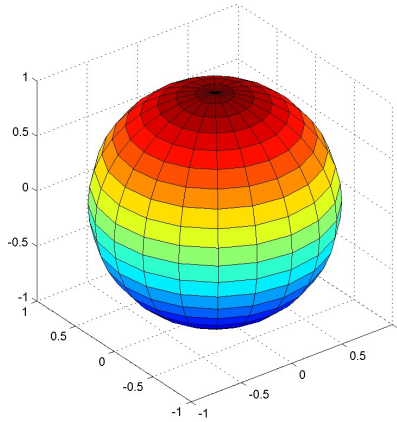



$$\mathbf{T}^{-1}\mathbf{T}\mathbf{D} = \mathbf{D} \Rightarrow \mathbf{T}^{-1}\mathbf{T} = \mathbf{I}$$

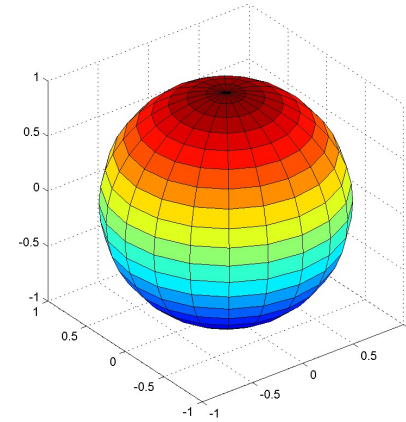
- The product of a matrix and its inverse is the identity matrix
  - Transforming an object, and then inverse transforming it gives us back the original object

$$\mathbf{T}\mathbf{T}^{-1}\mathbf{D} = \mathbf{D} \Rightarrow \mathbf{T}\mathbf{T}^{-1} = \mathbf{I}$$

# Matrix inversion (division)

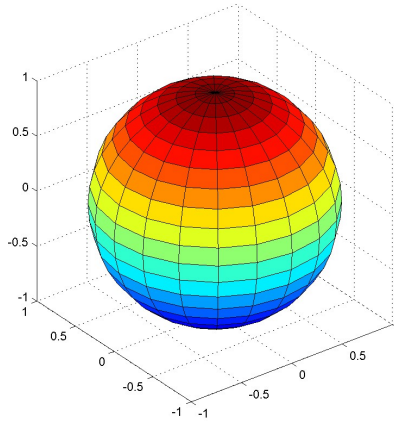


$$\mathbf{T}=\mathbf{I}$$

$$\mathbf{T}^{-1}=\mathbf{I}$$

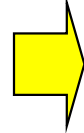


- Inverse of the unit matrix is itself

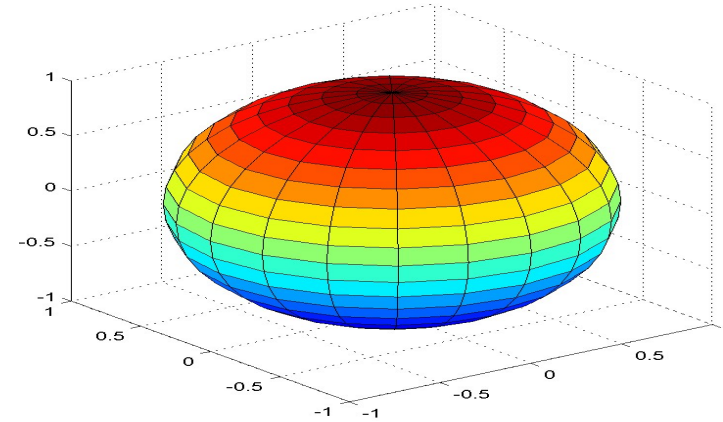
# Matrix inversion (division)



$$\mathbf{T} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

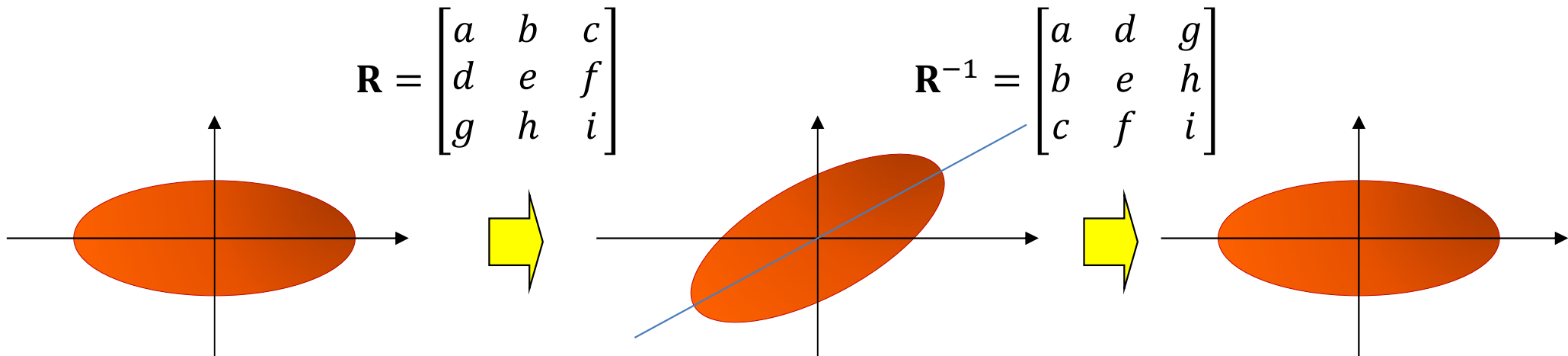


$$\mathbf{T}^{-1} = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- Inverse of the unit matrix is itself
- Inverse of a diagonal is diagonal

# Matrix inversion (division)



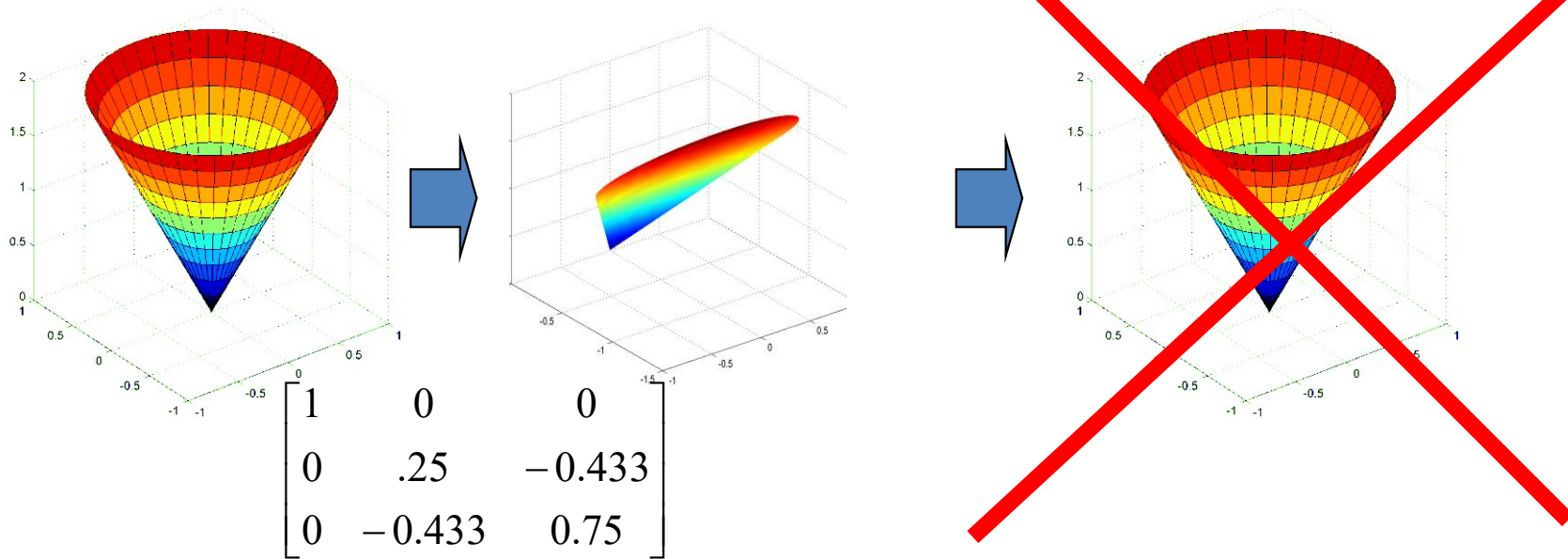
- Inverse of the unit matrix is itself
- Inverse of a diagonal is diagonal
- **Inverse of a rotation is a (counter)rotation (its transpose!)**
  - In 2D a forward rotation  $\theta$  by is cancelled by a backward rotation of  $-\theta$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \mathbf{R}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

- More generally, in any number of dimensions:  $\mathbf{R}^{-1} = \mathbf{R}^T$

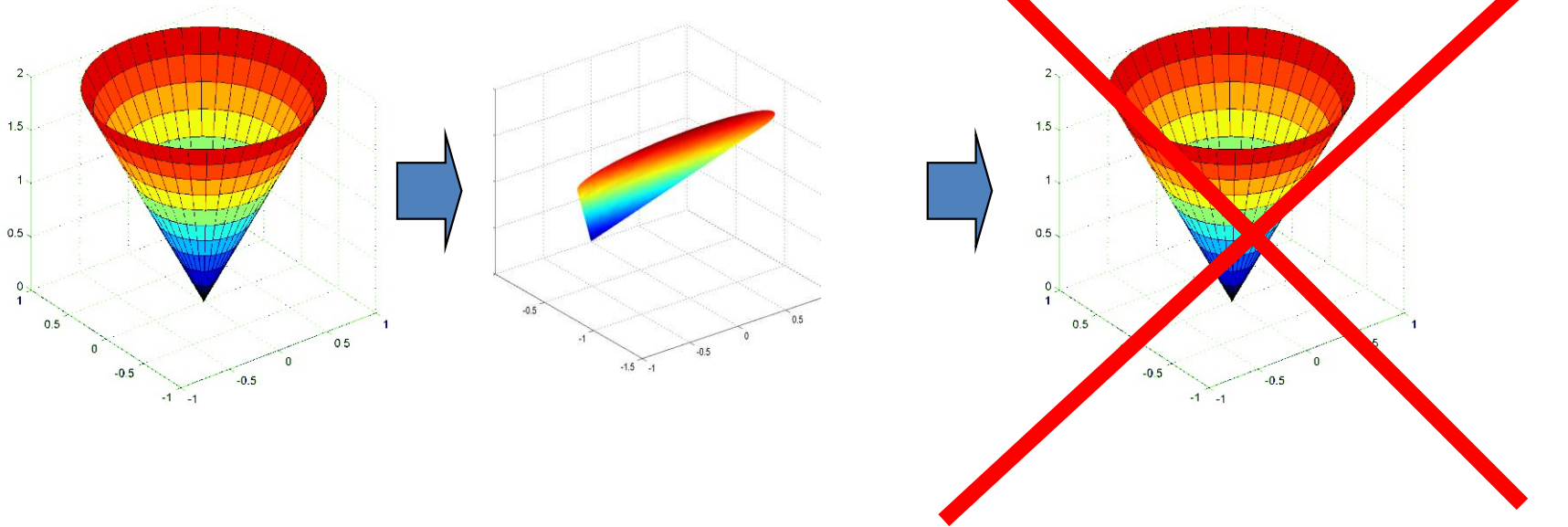


# Inverting rank-deficient matrices



- Rank deficient matrices “flatten” objects
  - In the process, multiple points in the original object get mapped to the same point in the transformed object
- It is not possible to go “back” from the flattened object to the original object
  - Because of the many-to-one forward mapping
- Rank deficient matrices have no inverse

# Matrix inversion (division)



- Inverse of the unit matrix is itself
- Inverse of a diagonal is diagonal
- Inverse of a rotation is a (counter)rotation (its transpose!)
- Inverse of a rank deficient matrix does not exist!

# Poll 4

# Poll 4

- True or false: Using the standard definition of matrix inversion, the inverse of an arbitrary matrix is always the matrix's transpose.
  - T
  - **F**
- True or false: Using the standard definition of matrix inversion, the inverse of a rank deficient matrix does NOT exist.
  - **T**
  - F

# Inverse Transform and Simultaneous Equation

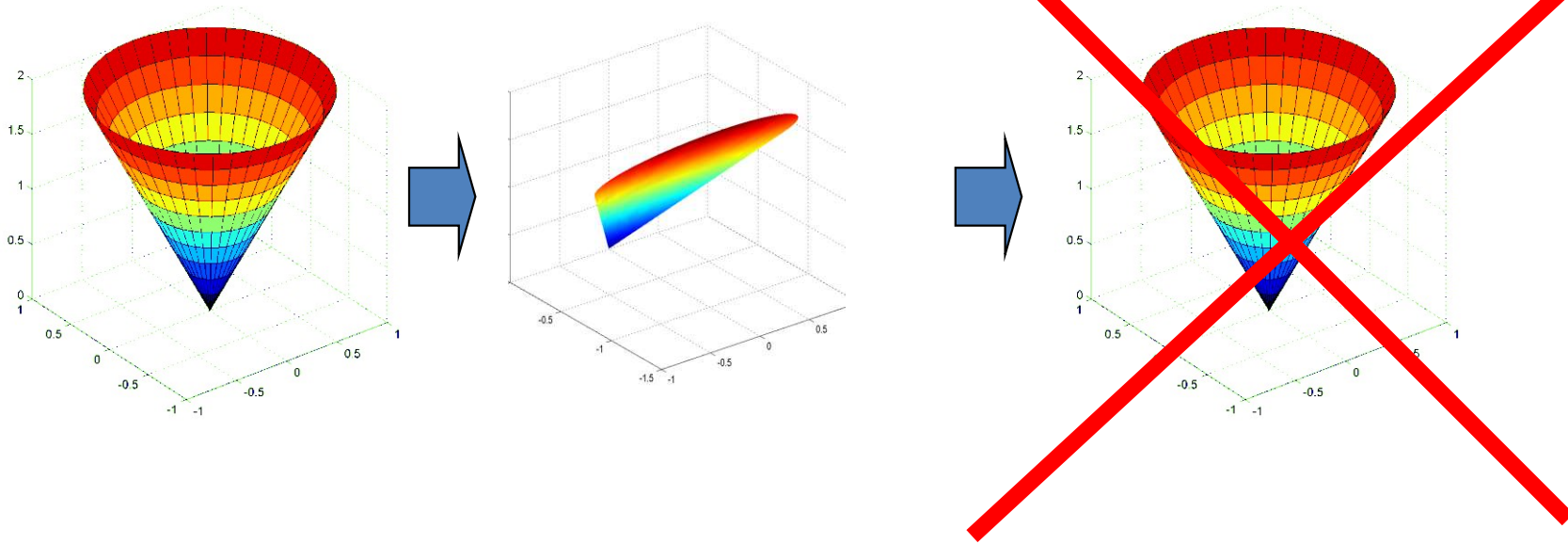
$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \longrightarrow \quad \begin{aligned} a &= T_{11}x + T_{12}y + T_{13}z \\ b &= T_{21}x + T_{22}y + T_{23}z \\ c &= T_{31}x + T_{32}y + T_{33}z \end{aligned}$$

Given  $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$  and  $\mathbf{T}$  find  $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

- Inverting the transform is identical to solving simultaneous equations

# Inverting rank-deficient matrices



- Rank deficient matrices have no inverse
  - In this example, there is no *unique* inverse

# Inverse Transform and Simultaneous Equation

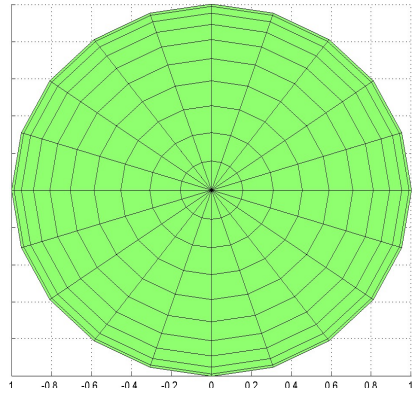
$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \longrightarrow \quad \begin{aligned} a &= T_{11}x + T_{12}y + T_{13}z \\ b &= T_{21}x + T_{22}y + T_{23}z \end{aligned}$$

Given  $\begin{bmatrix} a \\ b \end{bmatrix}$  and  $\mathbf{T}$  find  $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

- Inverting the transform is identical to solving simultaneous equations
- Rank-deficient transforms result in too-few ***independent*** equations
  - Cannot be inverted to obtain a *unique* solution

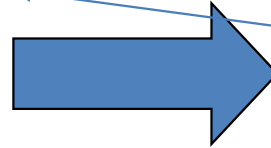
# Non-square Matrices



$$\begin{bmatrix} x_1 & x_2 & \cdot & \cdot & x_N \\ y_1 & y_2 & \cdot & \cdot & y_N \end{bmatrix}$$

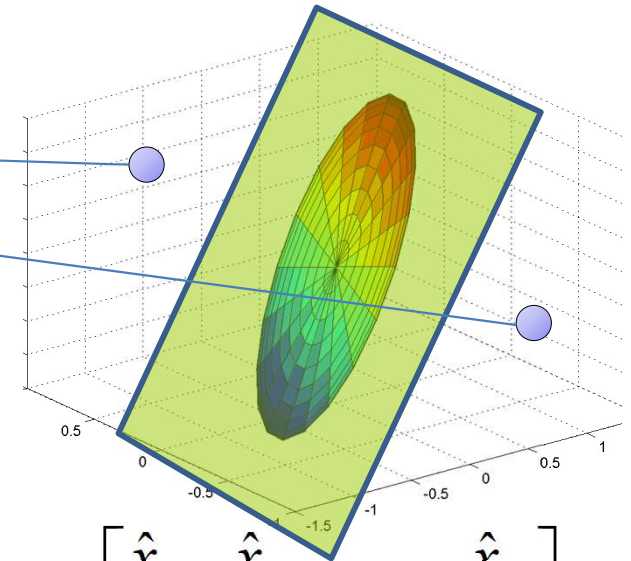
X = 2D data

?



$$\begin{bmatrix} .8 & .9 \\ .1 & .9 \\ .6 & 0 \end{bmatrix}$$

P = transform



$$\begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \cdot & \cdot & \hat{x}_N \\ \hat{y}_1 & \hat{y}_2 & \cdot & \cdot & \hat{y}_N \\ \hat{z}_1 & \hat{z}_2 & \cdot & \cdot & \hat{z}_N \end{bmatrix}$$

PX = 3D, rank 2

- When the transform *increases* the number of components most points in the new space will not have a corresponding preimage



# Inverse Transform and Simultaneous Equation

$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \\ T_{31} & T_{32} \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix} \quad \longrightarrow \quad \begin{aligned} a &= T_{11}x + T_{12}y \\ b &= T_{21}x + T_{22}y \\ c &= T_{31}x + T_{32}y \end{aligned}$$

Given  $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$  and  $\mathbf{T}$  find  $\begin{bmatrix} x \\ y \end{bmatrix}$

- Inverting the transform is identical to solving simultaneous equations
- Rank-deficient transforms result in too few independent equations
  - Cannot be inverted to obtain a unique solution
- Or too *many* equations
  - Cannot be inverted to obtain an exact solution

# The Pseudo Inverse (PINV)

$$V \approx T \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} x \\ y \\ z \end{bmatrix} \approx \text{Pinv}(T)V$$

- When you can't *really* invert  $T$ , you perform the *pseudo* inverse

# Generalization to matrices

- **Unique exact solution exists**
- **T must be square**

$$\mathbf{X} = \mathbf{T}\mathbf{Y} \Rightarrow \mathbf{Y} = \mathbf{T}^{-1}\mathbf{X}$$

Left multiplication

$$\mathbf{X} = \mathbf{Y}\mathbf{T} \Rightarrow \mathbf{Y} = \mathbf{X}\mathbf{T}^{-1}$$

Right multiplication

- **No unique exact solution exists**
  - At least one (if not both) of the forward and backward equations may be inexact
- **T may or may not be square**

$$\mathbf{X} = \mathbf{T}\mathbf{Y} \Rightarrow \mathbf{Y} = \text{Pinv}(\mathbf{T})\mathbf{X}$$

Left multiplication

$$\mathbf{X} = \mathbf{Y}\mathbf{T} \Rightarrow \mathbf{Y} = \mathbf{X}\text{Pinv}(\mathbf{T})$$

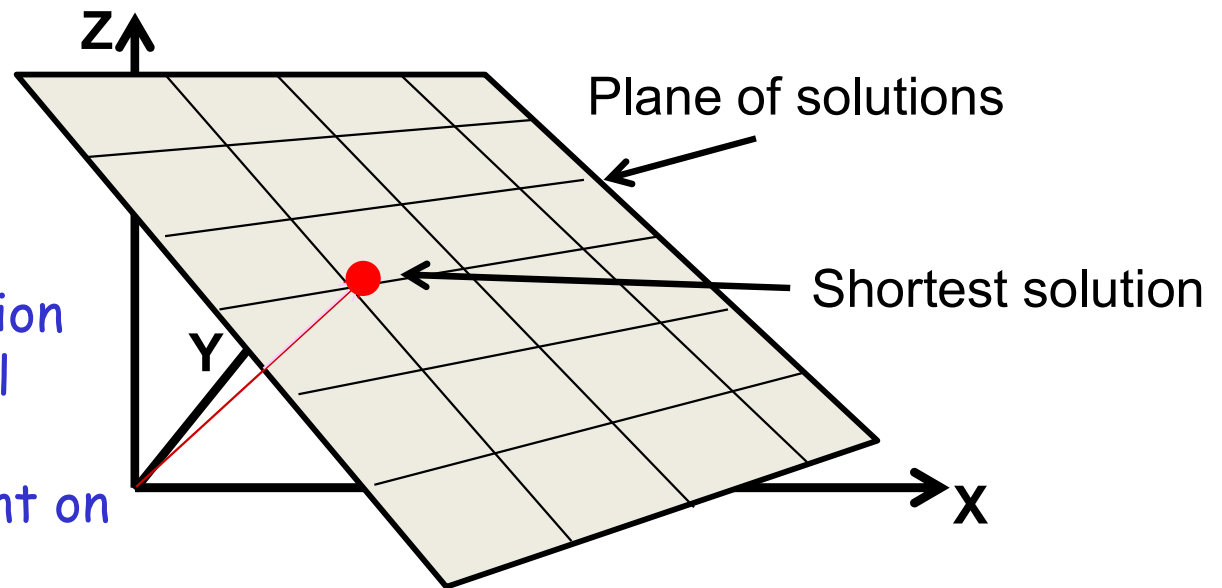
Right multiplication

# Underdetermined Pseudo Inverse

$$\begin{bmatrix} a \\ b \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \longrightarrow \quad \begin{aligned} a &= T_{11}x + T_{12}y + T_{13}z \\ b &= T_{21}x + T_{22}y + T_{23}z \end{aligned}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \text{Pinv}(\mathbf{T}) \begin{bmatrix} a \\ b \end{bmatrix}$$

Figure only meant for illustration for the above equations, actual set of solutions is a line, not a plane.  $\text{Pinv}(\mathbf{T})\mathbf{A}$  will be the point on the line closest to origin



- **Case 1: Too many solutions**
- $\text{Pinv}(\mathbf{T})\mathbf{A}$  picks the *shortest* solution

# The Pseudo Inverse for the underdetermined case

$$\begin{bmatrix} a \\ b \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \longrightarrow \quad \begin{aligned} a &= T_{11}x + T_{12}y + T_{13}z \\ b &= T_{21}x + T_{22}y + T_{23}z \end{aligned}$$

$$V \approx T \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \text{Pinv}(T)V$$

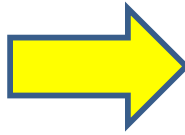
$$\text{Pinv}(T) = T^T (TT^T)^{-1}$$

$$T \begin{bmatrix} x \\ y \\ z \end{bmatrix} = T \text{Pinv}(T)V = TT^T (TT^T)^{-1}V = V$$

# The Pseudo Inverse

$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \\ T_{31} & T_{32} \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix}$$



$$a = T_{11}x + T_{12}y$$

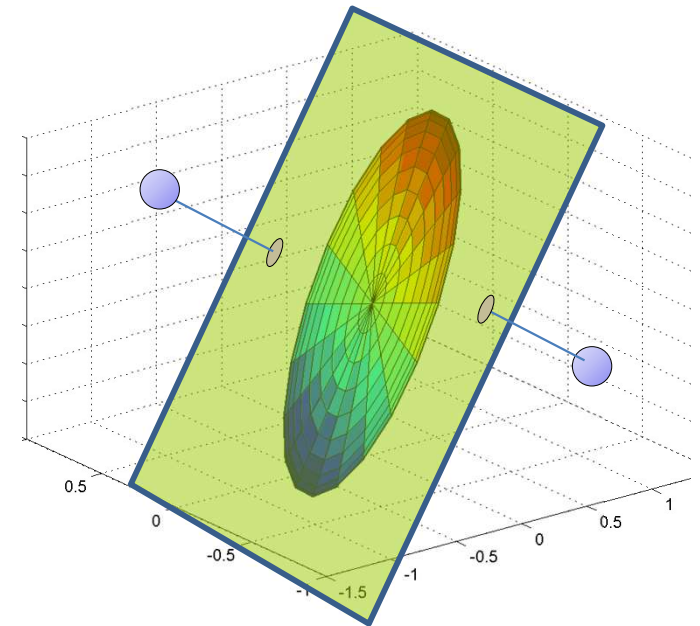
$$b = T_{21}x + T_{22}y$$

$$c = T_{31}x + T_{32}y$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \text{Pinv}(\mathbf{T}) \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\|\mathbf{A} - \mathbf{TX}\|^2$$

Figure only meant for illustration for the above equations,  $\text{Pinv}(\mathbf{T})$  will actually have 6 components. The error is a quadratic in 6 dimensions



- **Case 2: No exact solution**
- $\text{Pinv}(\mathbf{T})\mathbf{A}$  picks the solution that results in the lowest error

# The Pseudo Inverse for the overdetermined case

$$E = \|TX - A\|^2 = (TX - A)^T (TX - A)$$

$$E = X^T T^T T X - 2X^T T^T A + A^T A$$

Differentiating and equating to 0 we get:

$$X = (T^T T)^{-1} T^T A = \text{Pinv}(T)A$$

$$\text{Pinv}(T) = (T^T T)^{-1} T^T$$

# Shortcut: overdetermined case

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \begin{aligned} a &= T_{11}x + T_{12}y \\ b &= T_{21}x + T_{22}y \\ c &= T_{31}x + T_{32}y \end{aligned}$$

$$\mathbf{V} \approx \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \mathbf{T}^T \mathbf{V} \approx \mathbf{T}^T \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \begin{bmatrix} x \\ y \end{bmatrix} = (\mathbf{T}^T \mathbf{T})^{-1} \mathbf{T}^T \mathbf{V}$$

$$\mathit{Pinv}(\mathbf{T}) = (\mathbf{T}^T \mathbf{T})^{-1} \mathbf{T}^T$$

Note that in this case:

$$\mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{T} \mathit{Pinv}(\mathbf{T}) \mathbf{V} = \mathbf{T} (\mathbf{T}^T \mathbf{T})^{-1} \mathbf{T}^T \mathbf{V} \neq \mathbf{V}$$

Why?



# Overdetermined vs Underdetermined

- Underdetermined case: Exact solution exists. We find *one* of the exact solutions. Hence..

$$T \begin{bmatrix} x \\ y \\ z \end{bmatrix} = T \text{Pinv}(T)V = TT^T (TT^T)^{-1}V = V$$

- Overdetermined case: Solution generally does not exist. Solution is only an approximation..

$$T \begin{bmatrix} x \\ y \end{bmatrix} = T \text{Pinv}(T)V = T(T^T T)^{-1}T^T V \neq V$$

# Properties of the Pseudoinverse

- For the underdetermined case:

$$TPinv(T) = \mathbf{I}$$

- For the overdetermined case

$$TPinv(T) = ?$$

- We return to this question shortly

# Matrix inversion (division)

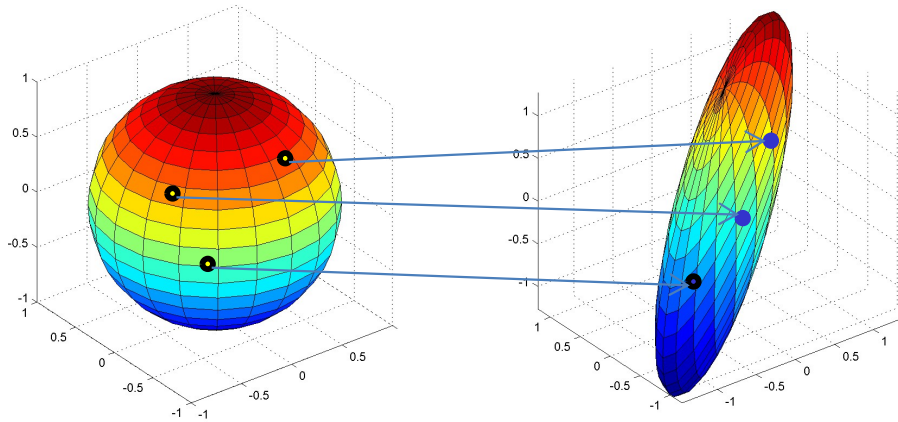
- The inverse of matrix multiplication
  - Not element-wise division!!
- Provides a way to “undo” a linear transformation
- For square matrices: Pay attention to multiplication side!

$$\mathbf{A} \cdot \mathbf{B} = \mathbf{C}, \quad \mathbf{A} = \mathbf{C} \cdot \mathbf{B}^{-1}, \quad \mathbf{B} = \mathbf{A}^{-1} \cdot \mathbf{C}$$

- If matrix is not square use a matrix pseudoinverse:

$$\mathbf{A} \cdot \mathbf{B} \approx \mathbf{C}, \quad \mathbf{A} = \mathbf{C} \cdot \mathbf{B}^+, \quad \mathbf{B} = \mathbf{A}^+ \cdot \mathbf{C}$$

# Finding the Transform

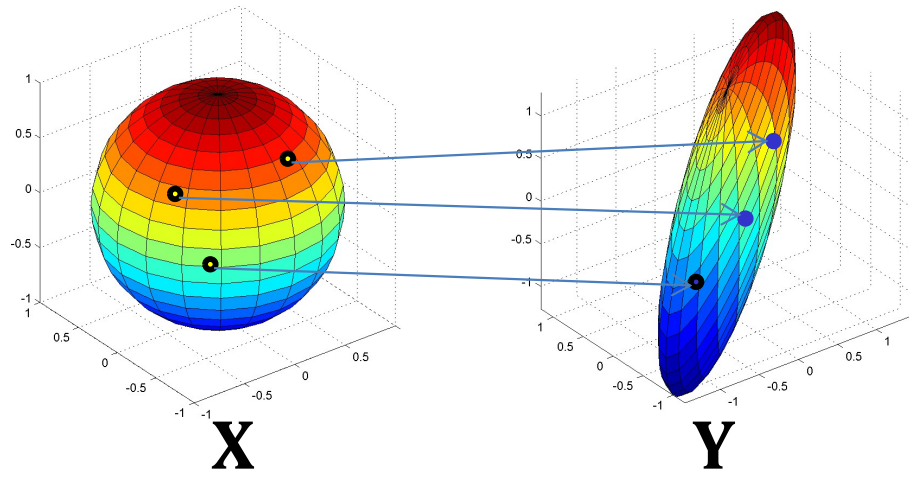


- Given examples
  - $\mathbf{T} \cdot \mathbf{X}_1 = Y_1$
  - $\mathbf{T} \cdot \mathbf{X}_2 = Y_2$
  - ..
  - $\mathbf{T} \cdot \mathbf{X}_N = Y_N$
- Find  $\mathbf{T}$

# Finding the Transform

$$\mathbf{X} = \begin{bmatrix} \uparrow & \vdots & \uparrow \\ X_1 & \ddots & X_N \\ \downarrow & \vdots & \downarrow \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} \uparrow & \vdots & \uparrow \\ Y_1 & \ddots & Y_N \\ \downarrow & \vdots & \downarrow \end{bmatrix}$$



$$\mathbf{Y} = \mathbf{TX}$$

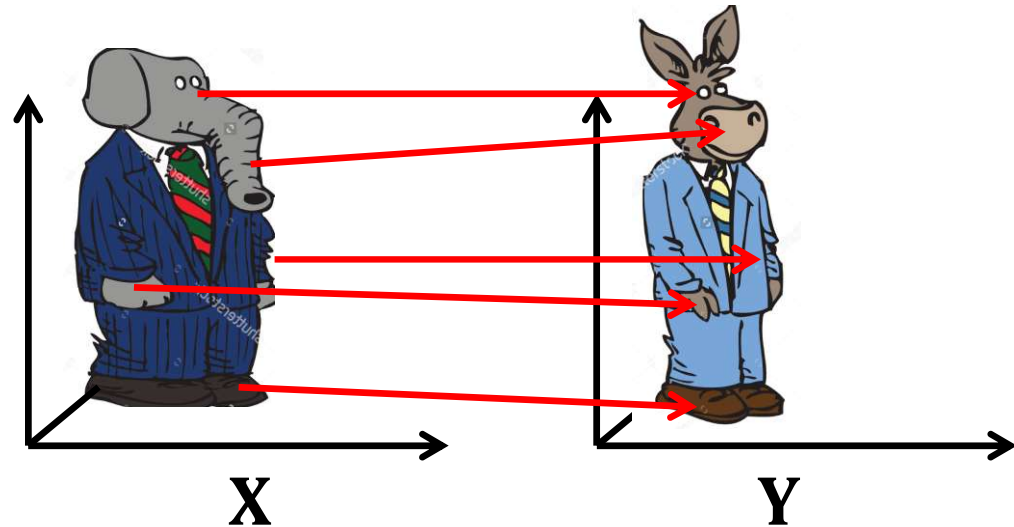
$$\mathbf{T} = \mathbf{Y}\text{Pinv}(\mathbf{X})$$

- Pinv works here too

# Finding the Transform: Inexact

$$\mathbf{X} = \begin{bmatrix} \uparrow & \vdots & \uparrow \\ X_1 & \ddots & X_N \\ \downarrow & \vdots & \downarrow \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} \uparrow & \vdots & \uparrow \\ Y_1 & \ddots & Y_N \\ \downarrow & \vdots & \downarrow \end{bmatrix}$$



$$\mathbf{Y} \approx \mathbf{TX} \Rightarrow \mathbf{T} = \mathbf{Y} \mathbf{P} \mathbf{inv}(\mathbf{X})$$

$$\text{minimizes } \sum_i \|Y_i - \mathbf{T}X_i\|^2$$

- Even works for inexact solutions
- We *desire* to find a linear transform  $\mathbf{T}$  that maps  $\mathbf{X}$  to  $\mathbf{Y}$ 
  - But such a linear transform doesn't really exist
- *Pinv* will give us the “best guess” for  $\mathbf{T}$  that minimizes the total squared error between  $\mathbf{Y}$  and  $\mathbf{TX}$

# Overview

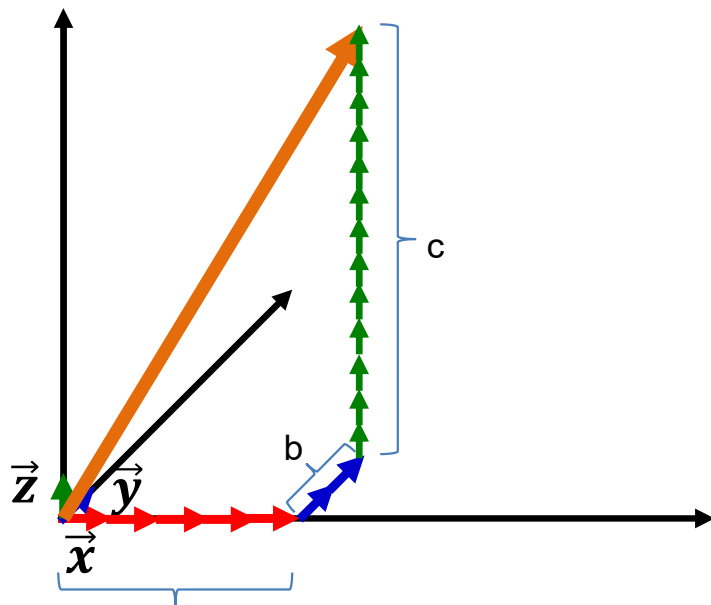
- Vectors and matrices
- Basic vector/matrix operations
- Various matrix types
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- Solving simultaneous equations
- **Projections**
- Eigen decomposition
- SVD

# Flashback: The *true* representation of a vector

$$v = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ using } \vec{x}, \vec{y}, \vec{z}$$

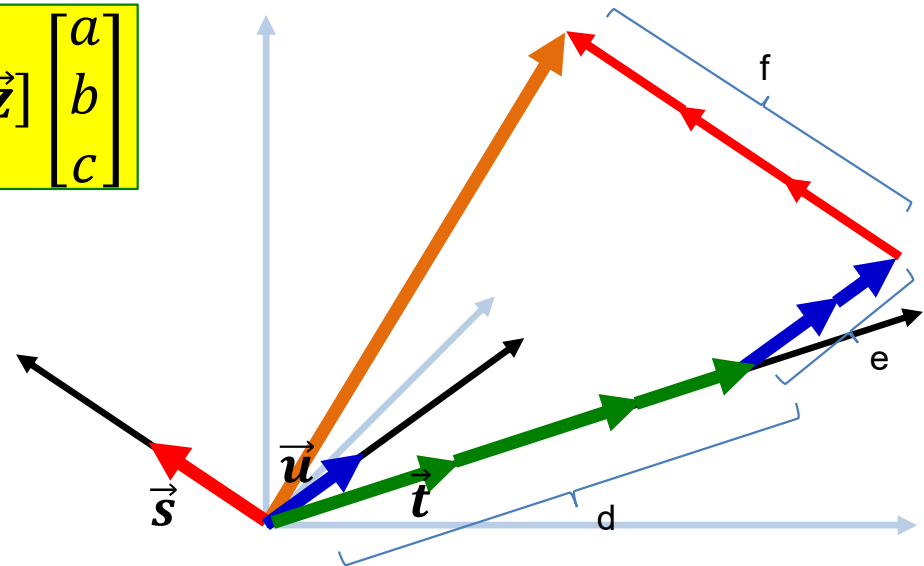
$$v = a\vec{x} + b\vec{y} + c\vec{z}$$

$$v = [\vec{x} \quad \vec{y} \quad \vec{z}] \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$



$$v = d\vec{u} + e\vec{v} + f\vec{w}$$

$$v = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$



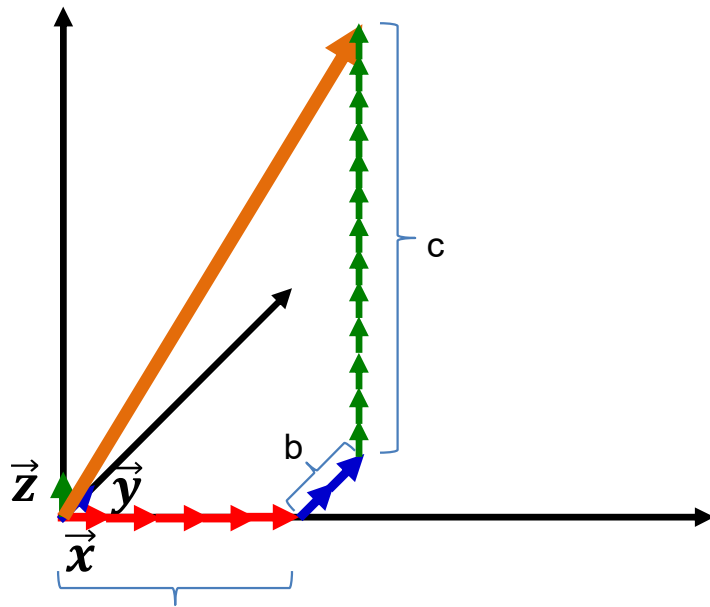
$$v = [\vec{s} \quad \vec{t} \quad \vec{u}] \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$

- What the column (or row) of numbers really means
  - The “basis matrix” is implicit

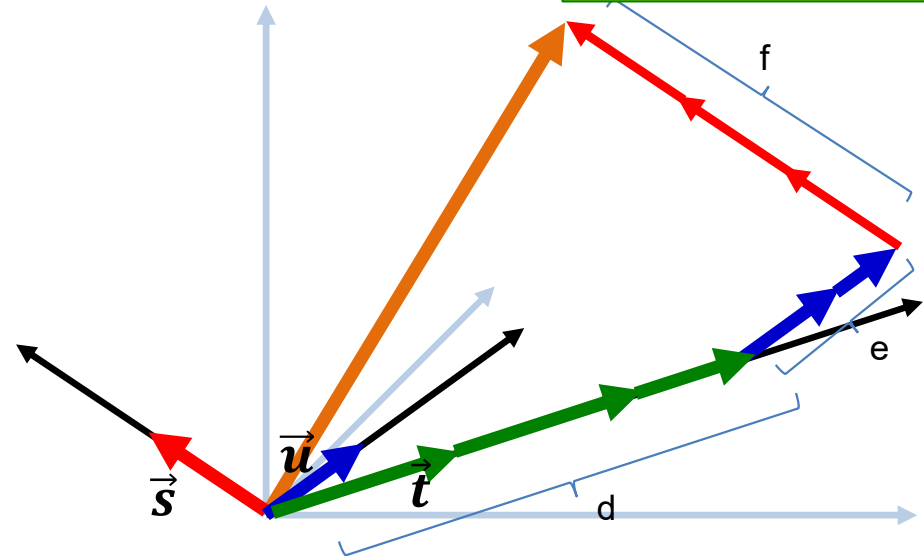


# Flashforward: Changing bases

$$v = [\vec{x} \quad \vec{y} \quad \vec{z}] \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$



$$v = [\vec{s} \quad \vec{t} \quad \vec{u}] \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$



- Given representation  $[a, b, c]$  and bases  $\vec{x} \quad \vec{y} \quad \vec{z}$ , how do we derive the representation  $[d \ e \ f]$  in terms of a different set of bases  $\vec{s} \quad \vec{t} \quad \vec{u}$ ?

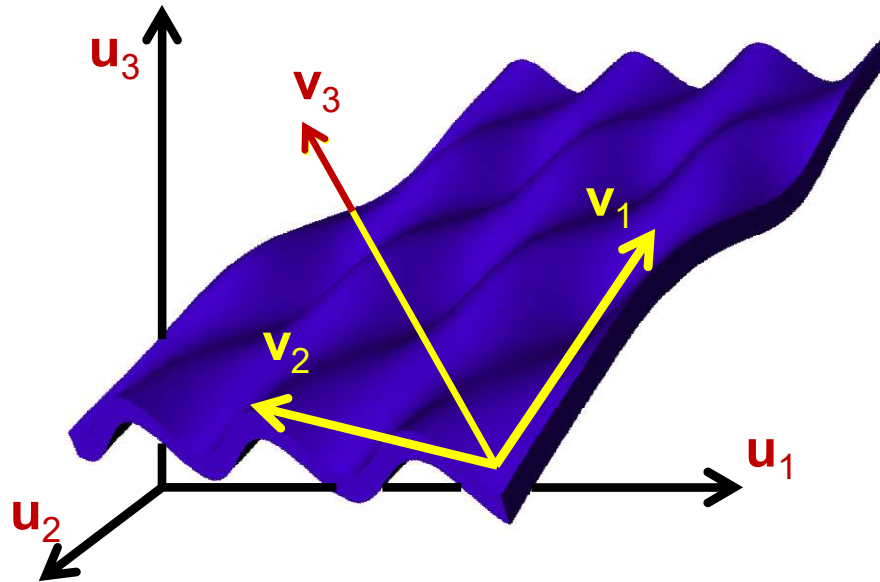
# Matrix as a Basis transform

$$\mathbf{X} = a\mathbf{v}_1 + b\mathbf{v}_2 + c\mathbf{v}_3, \quad \leftarrow \quad \mathbf{X} = x\mathbf{u}_1 + y\mathbf{u}_2 + z\mathbf{u}_3$$

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

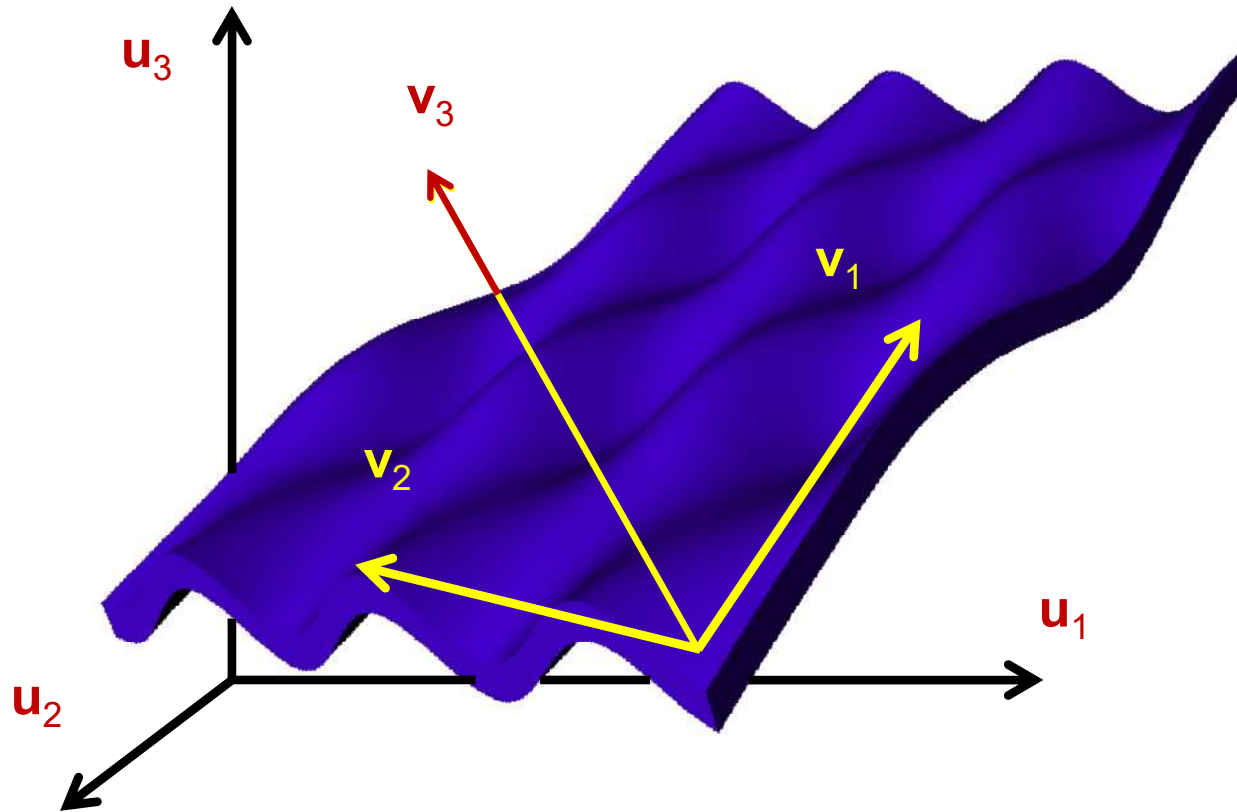
- A matrix transforms a representation in terms of a standard basis  $\mathbf{u}_1 \mathbf{u}_2 \mathbf{u}_3$  to a representation in terms of a different bases  $\mathbf{v}_1 \mathbf{v}_2 \mathbf{v}_3$
- Finding best bases: Find matrix that transforms standard representation to these bases

# Basis based representation



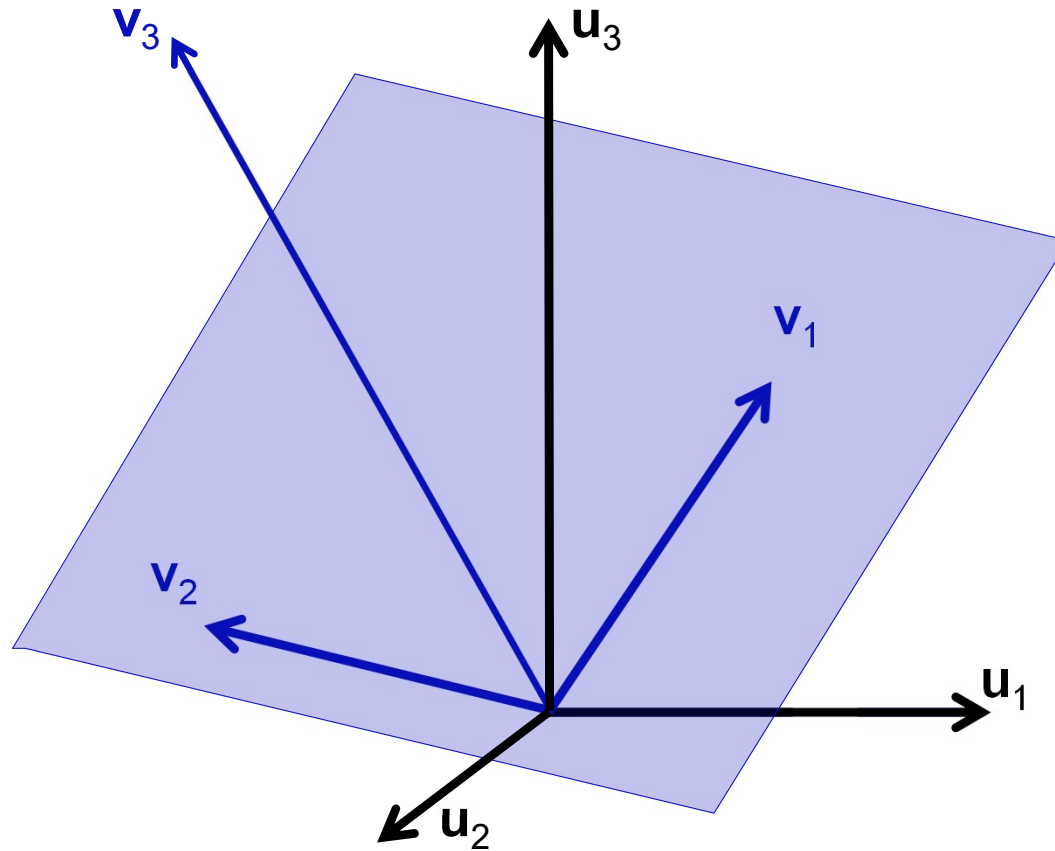
- A “good” basis captures *data* structure
- Here  $\mathbf{u}_1$ ,  $\mathbf{u}_2$  and  $\mathbf{u}_3$  all take large values for data in the set
- But in the  $(\mathbf{v}_1 \mathbf{v}_2 \mathbf{v}_3)$  set, coordinate values along  $\mathbf{v}_3$  are always small for data on the blue sheet
  - $\mathbf{v}_3$  likely represents a “noise subspace” for these data

# Basis based representation



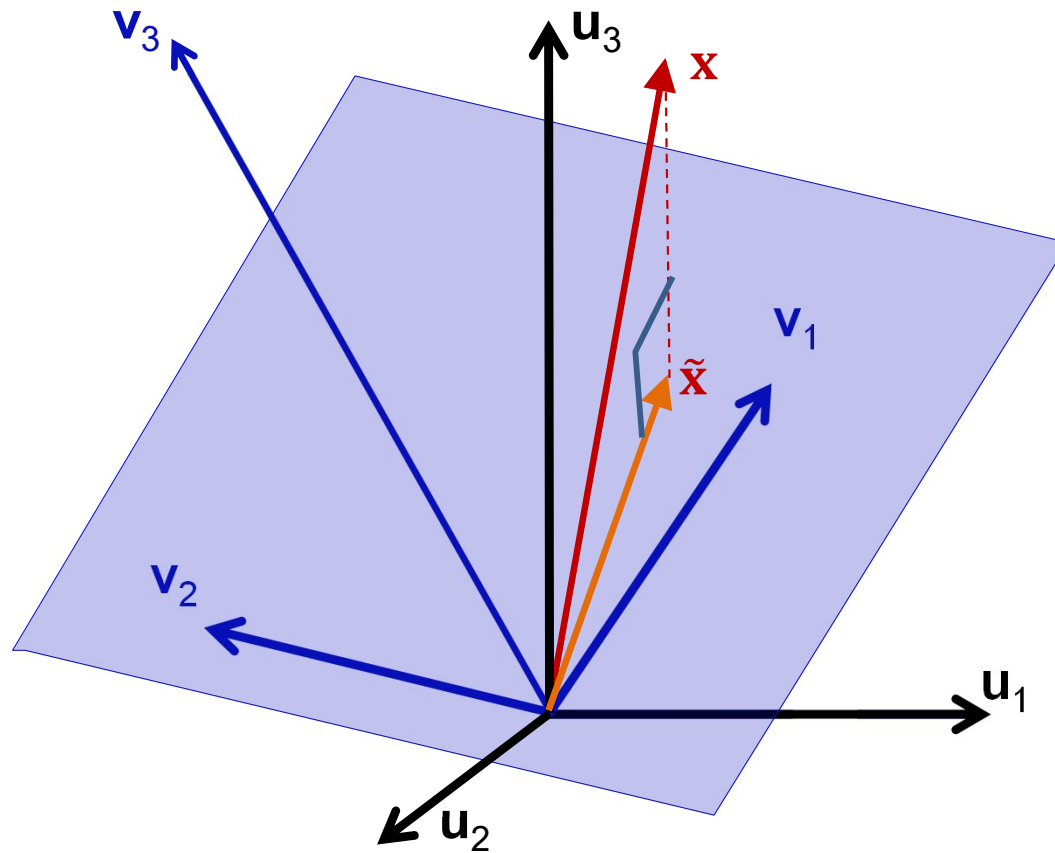
- *The most important challenge* in ML: Find the best set of bases for a given data set

# Basis based representation



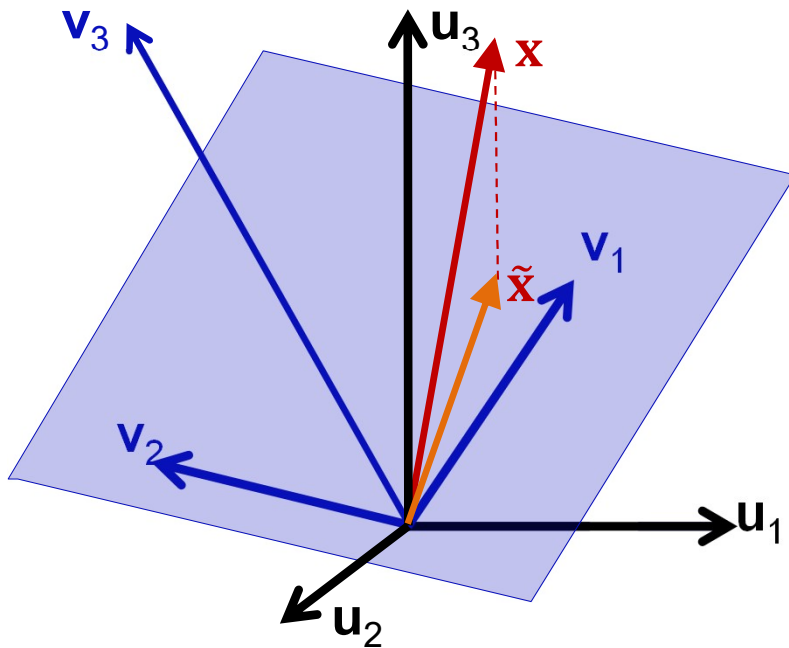
- **Modified problem:** Given the new bases  $v_1, v_2, v_3$ 
  - Find best representation of every data point on  $v_1$ - $v_2$  plane
    - Put it on the main sheet and disregard the  $v_3$  component

# Basis based representation



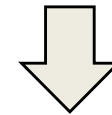
- **Modified problem:**
  - For any vector  $\mathbf{x}$
  - Find the closest approximation  $\tilde{\mathbf{x}} = a\mathbf{v}_1 + b\mathbf{v}_2$ 
    - Which lies entirely in the  $\mathbf{v}_1$ - $\mathbf{v}_2$  plane

# Basis based representation

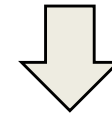


$$\mathbf{V} = [\mathbf{v}_1 \mathbf{v}_2] \quad \mathbf{a} = \begin{bmatrix} a \\ b \end{bmatrix}$$

$$\mathbf{x} \approx \mathbf{V}\mathbf{a}$$



$$\mathbf{a} = \mathbf{V}^+ \mathbf{x}$$

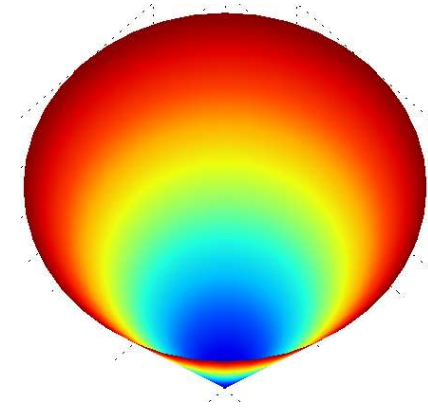
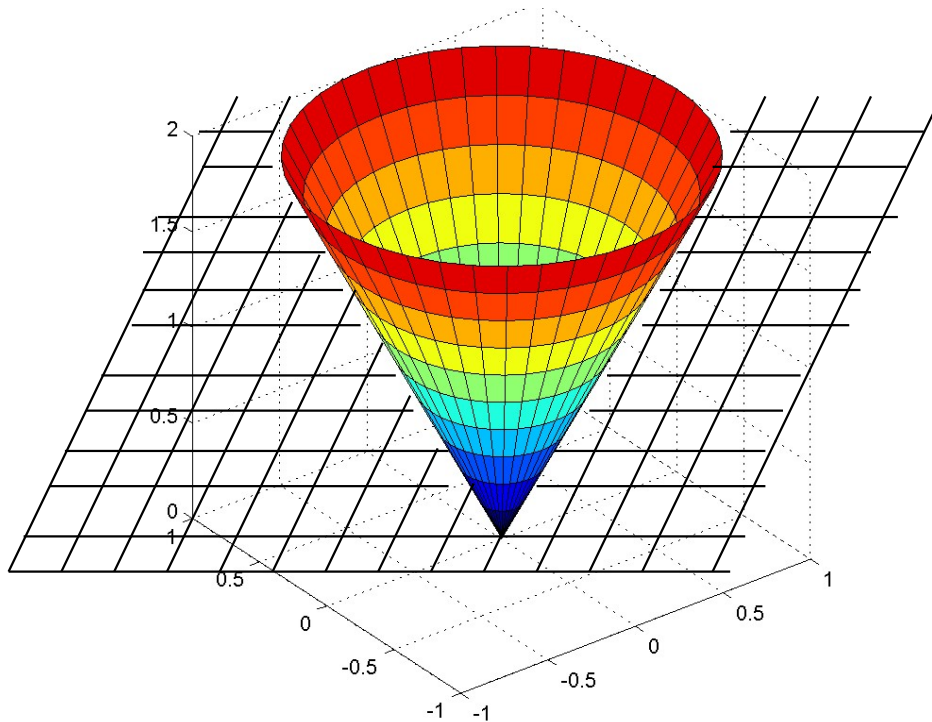


$$\tilde{\mathbf{x}} = \mathbf{V}\mathbf{V}^+ \mathbf{x}$$

- $\mathbf{P} = \mathbf{V}\mathbf{V}^+$  is the “projection” matrix that “projects” any vector  $\mathbf{x}$  down to its “shadow”  $\tilde{\mathbf{x}}$  on the  $\mathbf{v}_1$ - $\mathbf{v}_2$  plane

- Expanding:  $\mathbf{P} = \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T$

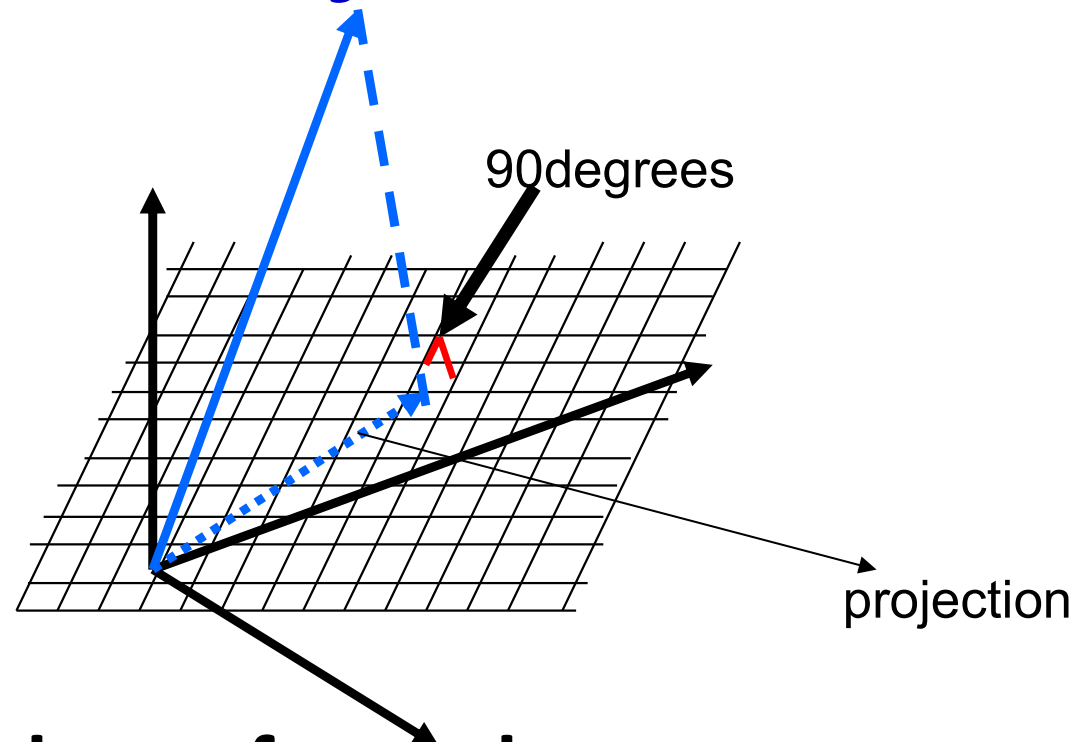
# Projections onto a plane



- What would we see if the cone to the left were transparent if we looked at it from above the plane shown by the grid?
  - Normal to the plane
  - Answer: the figure to the right
- How do we get this? Projection

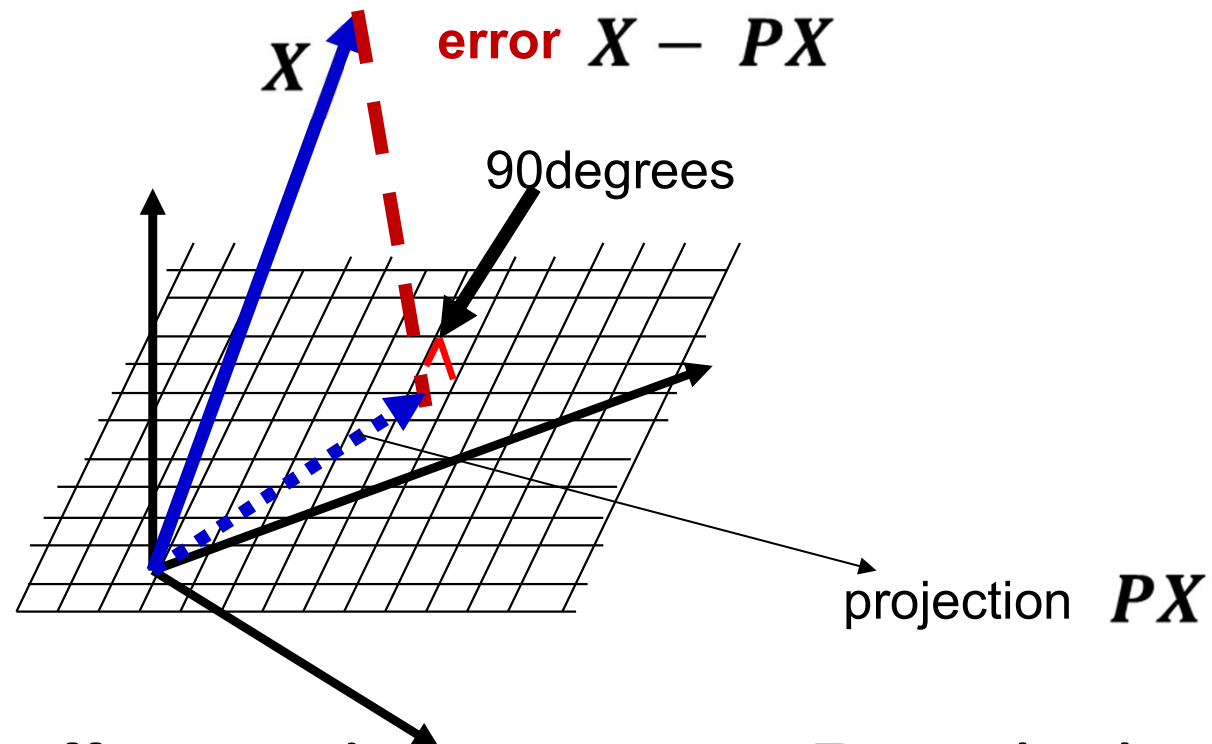


# Projections



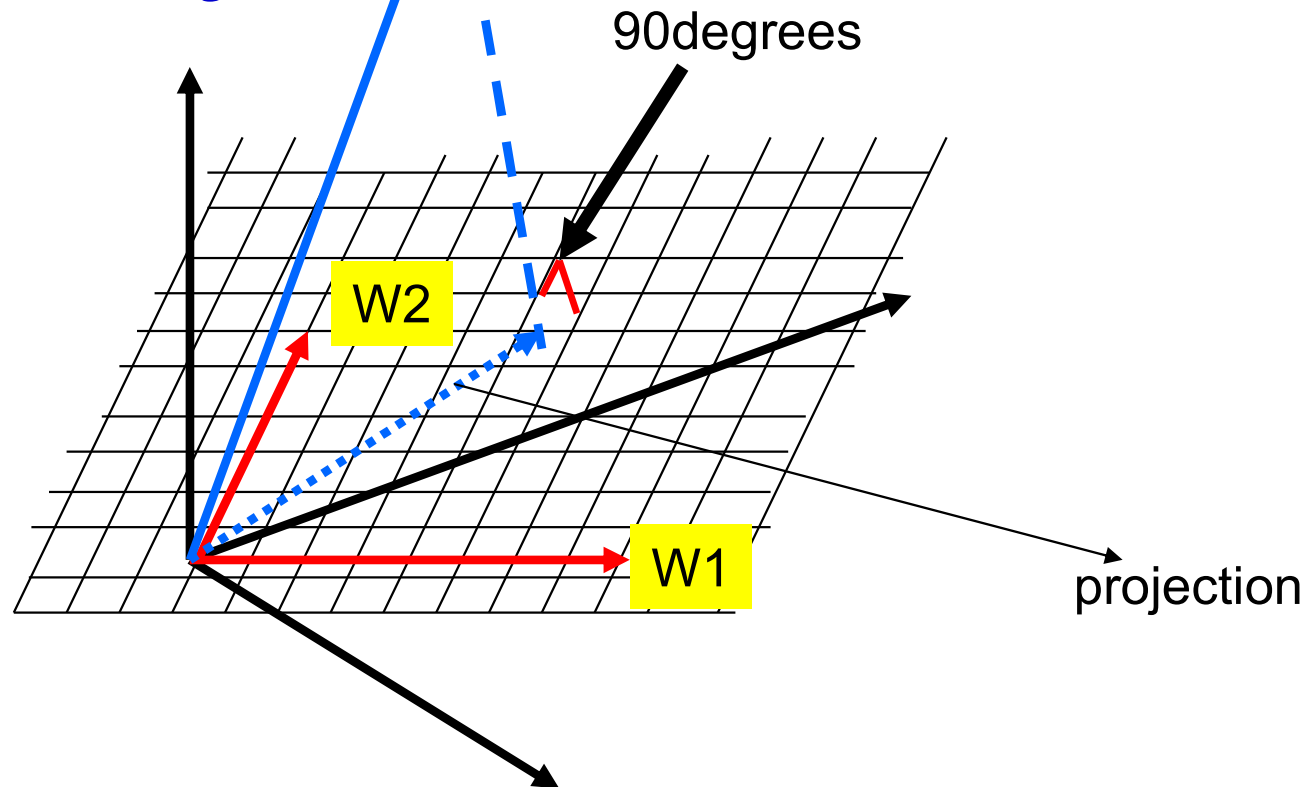
- **Actual problem: for each vector**
  - What is the corresponding vector on the plane that is “closest approximation” to it?
  - What is the *transform* that converts the vector to its approximation on the plane?

# Projections



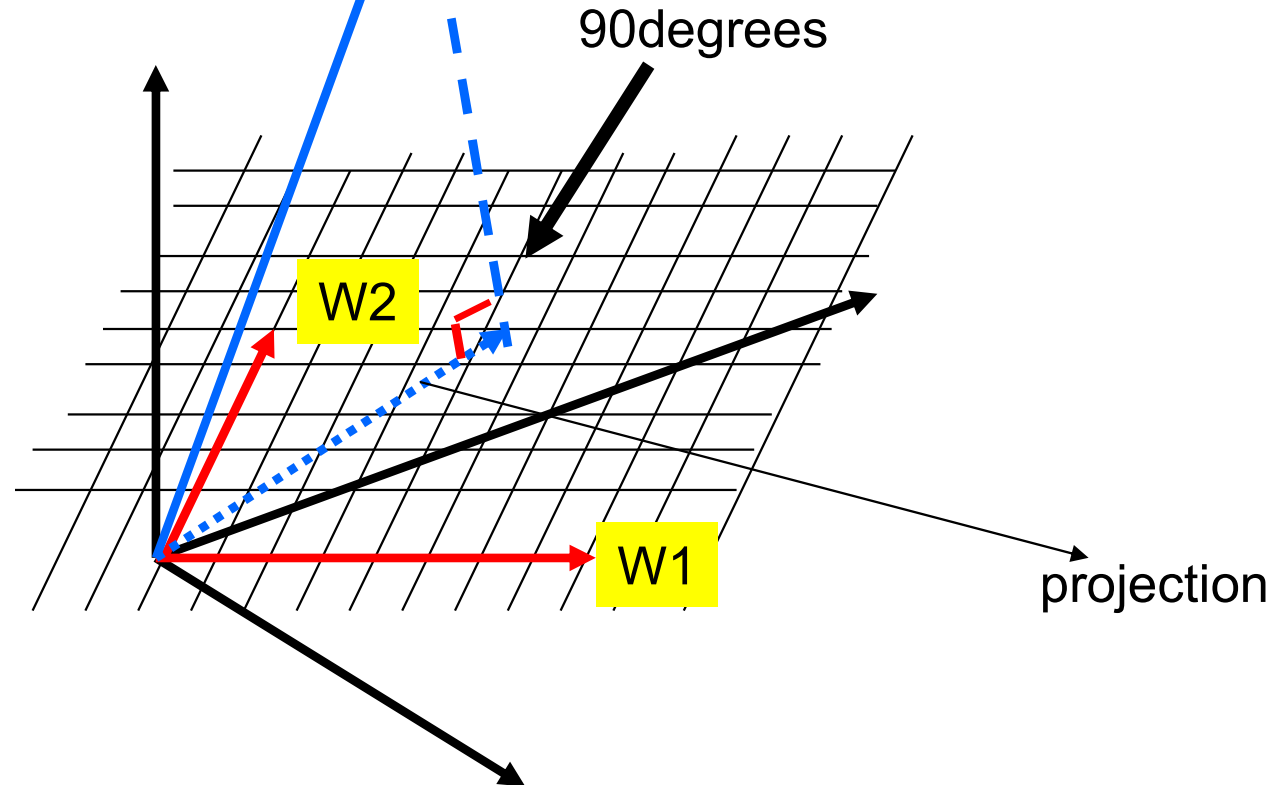
- **Arithmetically:** Find the matrix  $P$  such that
  - For **every** vector  $X$ ,  $PX$  lies on the plane
    - The plane is the column space of  $P$
  - $\|X - PX\|^2$  is the smallest possible

# Projection Matrix



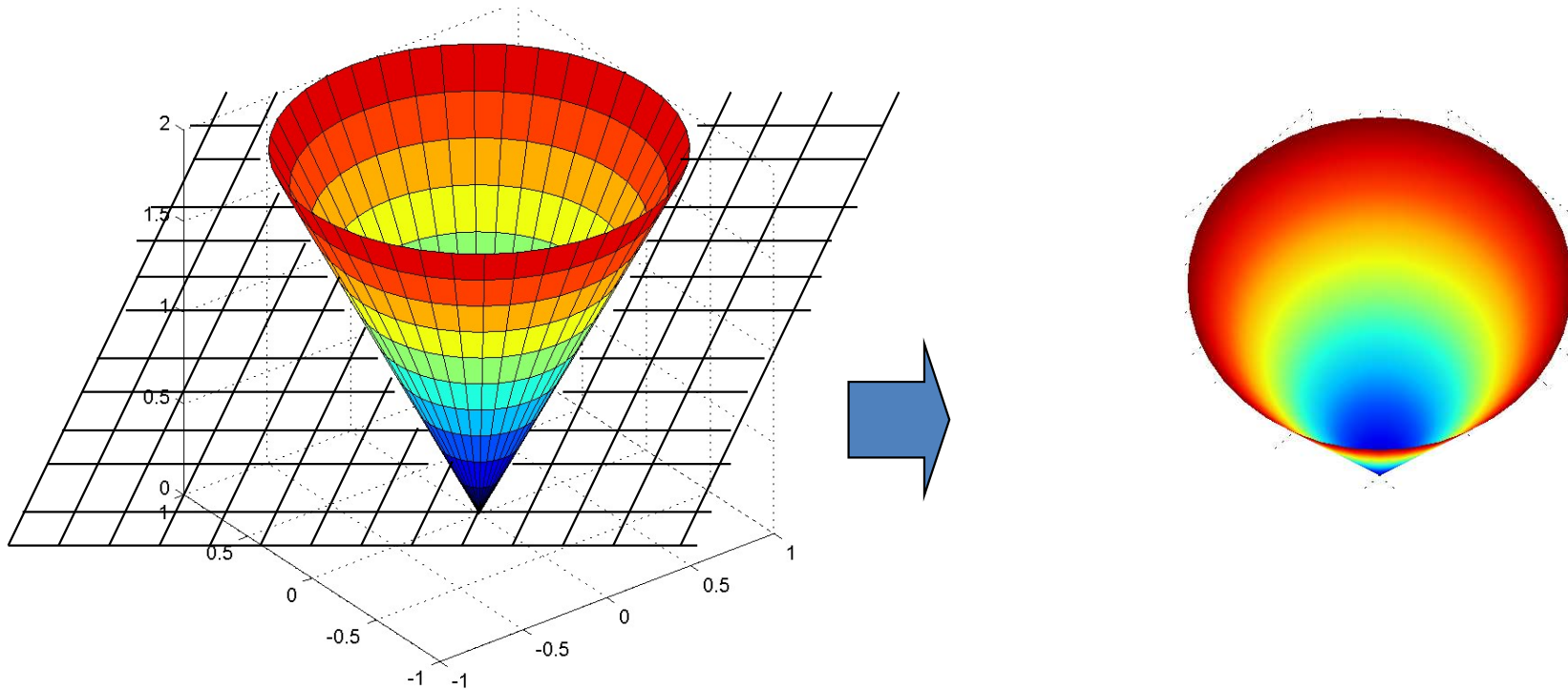
- Consider any set of *independent* vectors (bases)  $\mathbf{W}_1, \mathbf{W}_2, \dots$  on the plane
  - Arranged as a matrix  $[\mathbf{W}_1, \mathbf{W}_2, \dots]$
  - The plane is the *column space* of the matrix
- Find the projection matrix  $\mathbf{P}$  that projects on to the plane formed from  $[\mathbf{W}_1, \mathbf{W}_2, \dots]$

# Projection Matrix



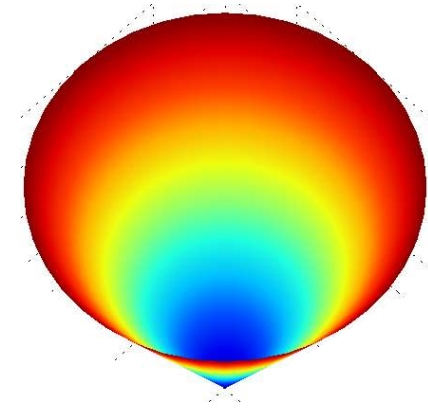
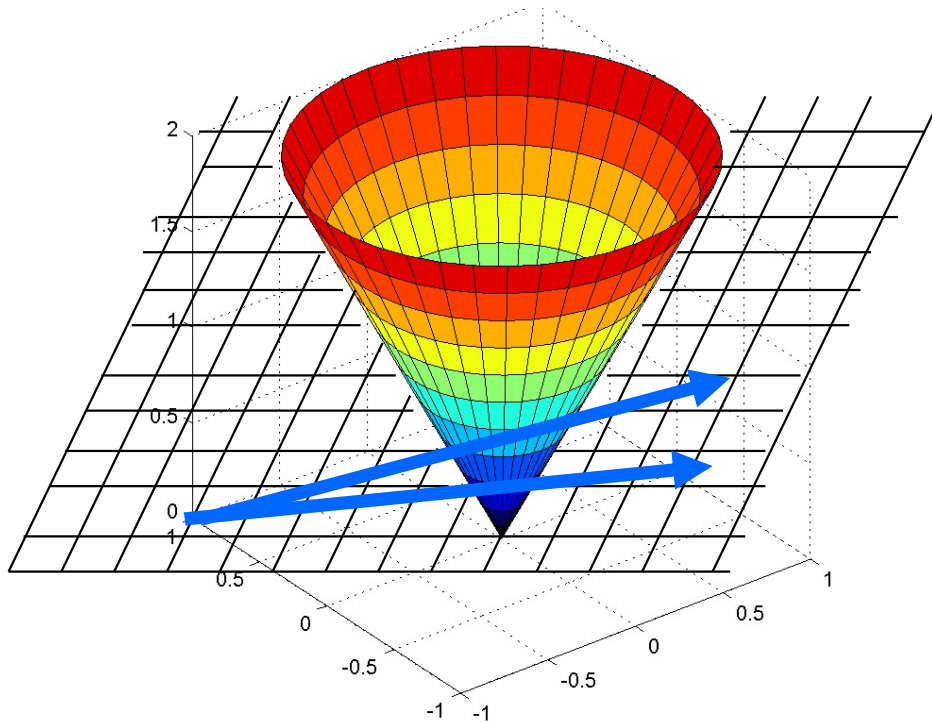
- Given a set of vectors  $W_1, W_2, \dots$  which form a matrix  $W = [W_1, W_2, \dots]$
- The projection matrix to transform a vector  $X$  to its projection on the plane is
  - $P = W(W^T W)^{-1} W^T$

# Projections



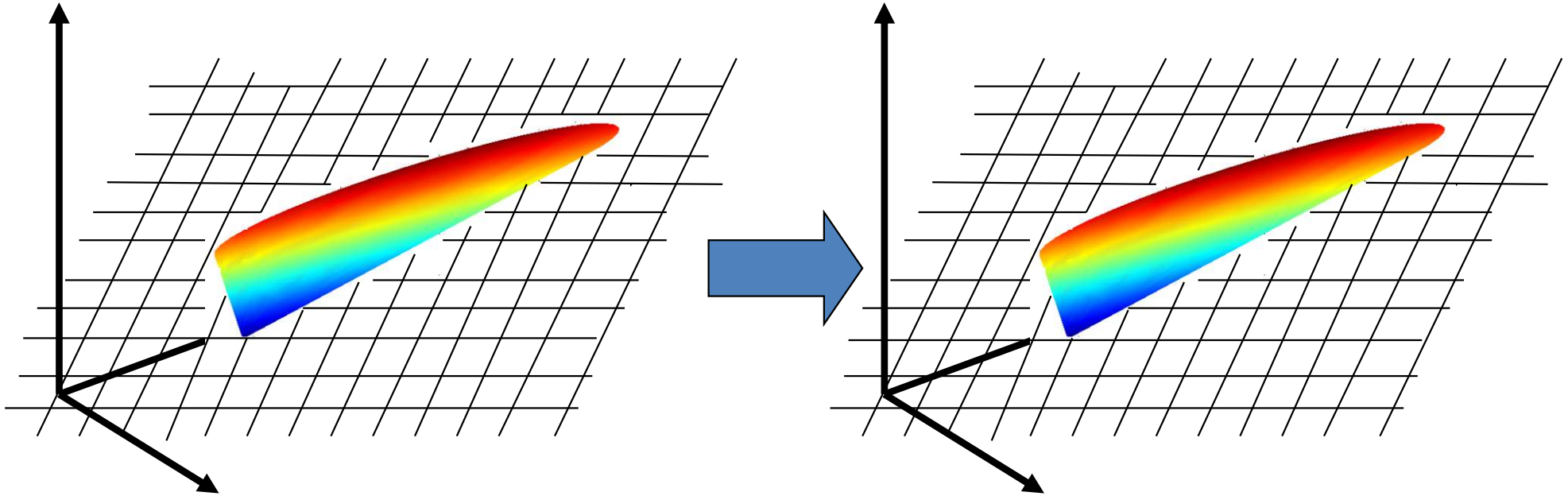
- HOW?

# Projections



- Draw any two vectors  $\mathbf{W}_1$  and  $\mathbf{W}_2$  that lie on the plane
  - **ANY two so long as they have different angles**
- Compose a matrix  $\mathbf{W} = [\mathbf{W}_1 \ \mathbf{W}_2 \dots]$
- Compose the projection matrix  $\mathbf{P} = \mathbf{W} (\mathbf{W}^T \mathbf{W})^{-1} \mathbf{W}^T$
- Multiply every point on the cone by  $\mathbf{P}$  to get its projection

# Projection matrix properties



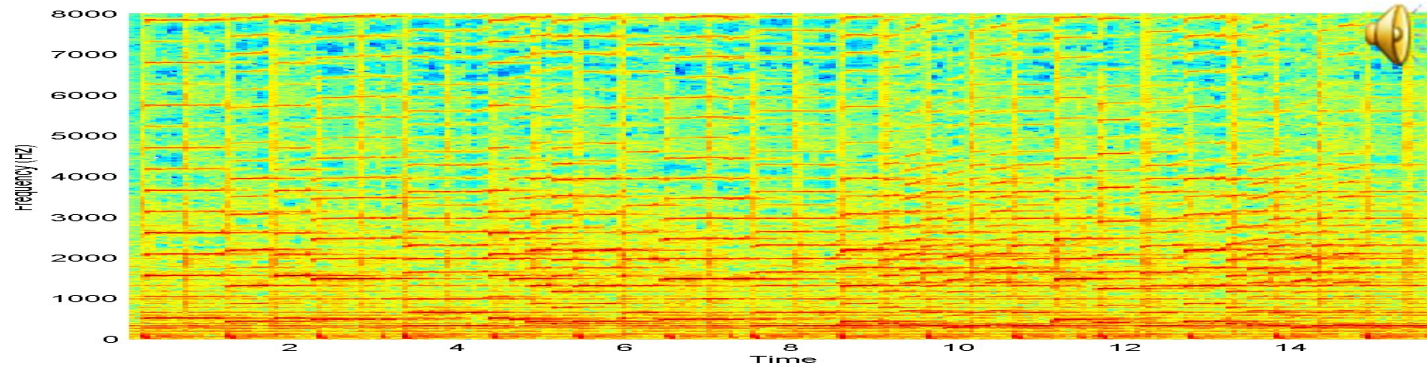
- The projection of any vector that is already on the plane is the vector itself
  - $\mathbf{PX} = \mathbf{X}$  if  $\mathbf{X}$  is on the plane
  - If the object is already on the plane, there is no further projection to be performed
- The projection of a projection is the projection
  - $\mathbf{P}(\mathbf{PX}) = \mathbf{PX}$
- Projection matrices are *idempotent*
  - $\mathbf{P}^2 = \mathbf{P}$

# Projections: A more physical meaning

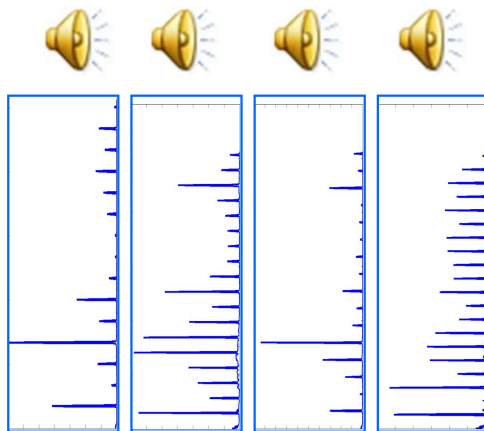
- Let  $\mathbf{W}_1, \mathbf{W}_2 \dots \mathbf{W}_k$  be “bases”
- We want to explain our data in terms of these “bases”
  - We often cannot do so
  - But we can explain a significant portion of it
- The portion of the data that can be expressed in terms of our vectors  $\mathbf{W}_1, \mathbf{W}_2, \dots \mathbf{W}_k$ , is the projection of the data on the  $\mathbf{W}_1 \dots \mathbf{W}_k$  (hyper) plane
  - In our previous example, the “data” were all the points on a cone, and the bases were vectors on the plane



# Projection : an example with sounds



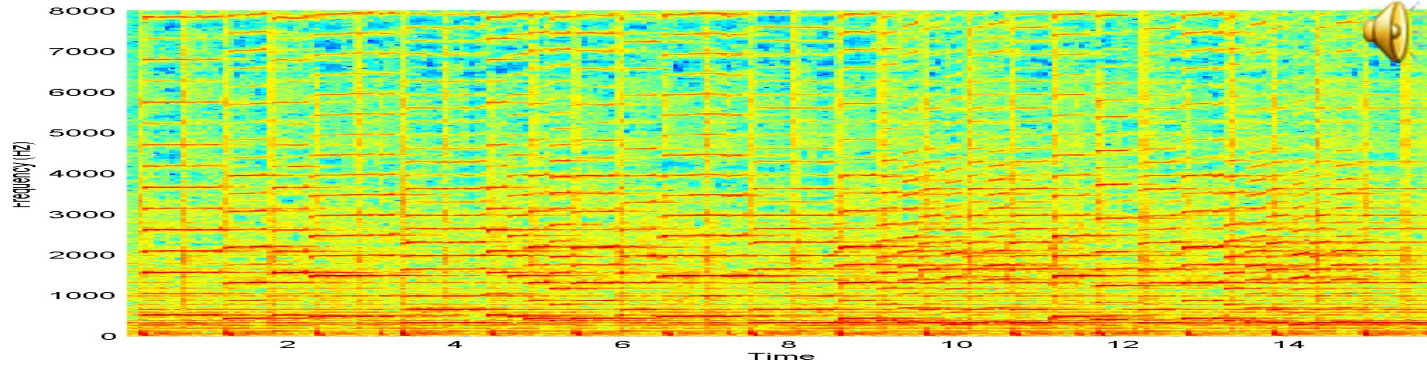
- The spectrogram (matrix) of a piece of music



- How much of the above music was composed of the above notes
  - I.e. how much can it be explained by the notes

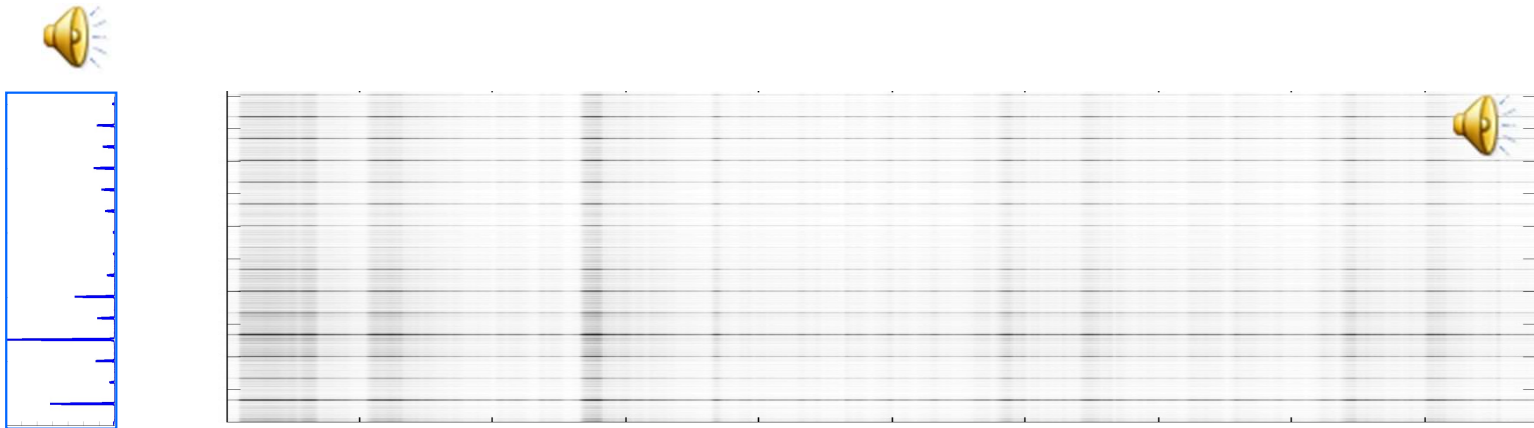
# Projection: one note

M =



- The spectrogram (matrix) of a piece of music

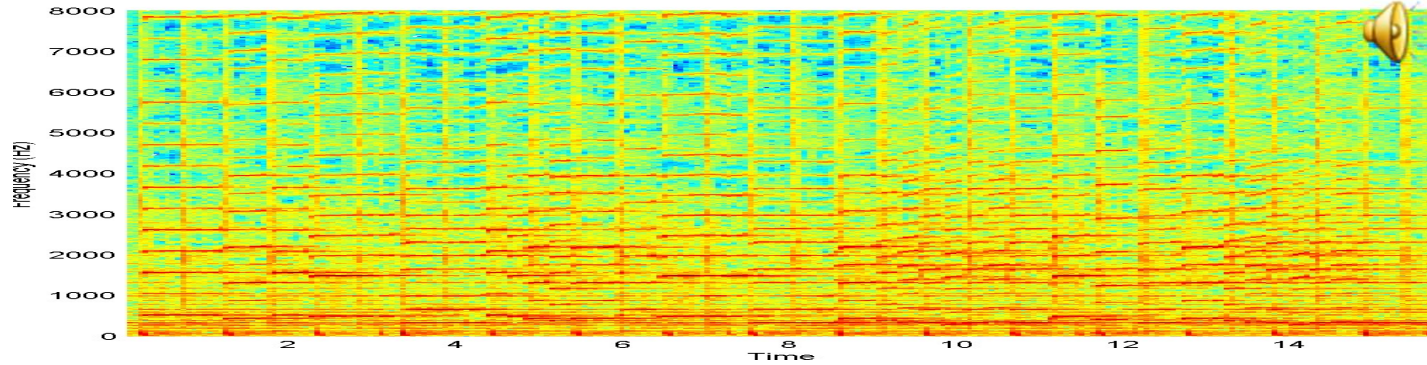
W =



- $M = \text{spectrogram}; W = \text{note}$
- $P = W(W^T W)^{-1} W^T$
- Projected Spectrogram =  $PM$

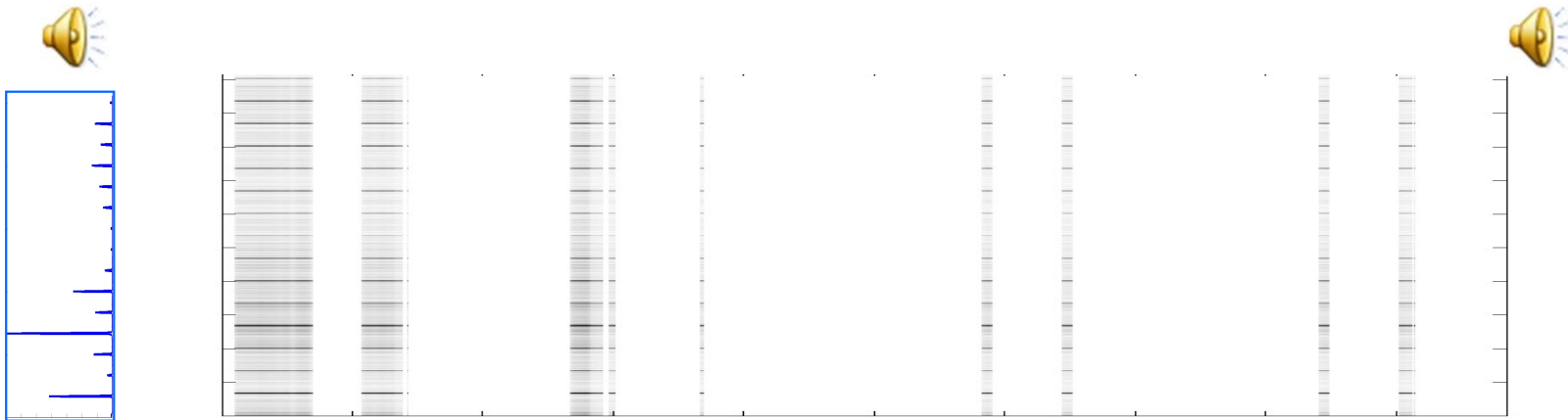
# Projection: one note – cleaned up

M =



- The spectrogram (matrix) of a piece of music

W =

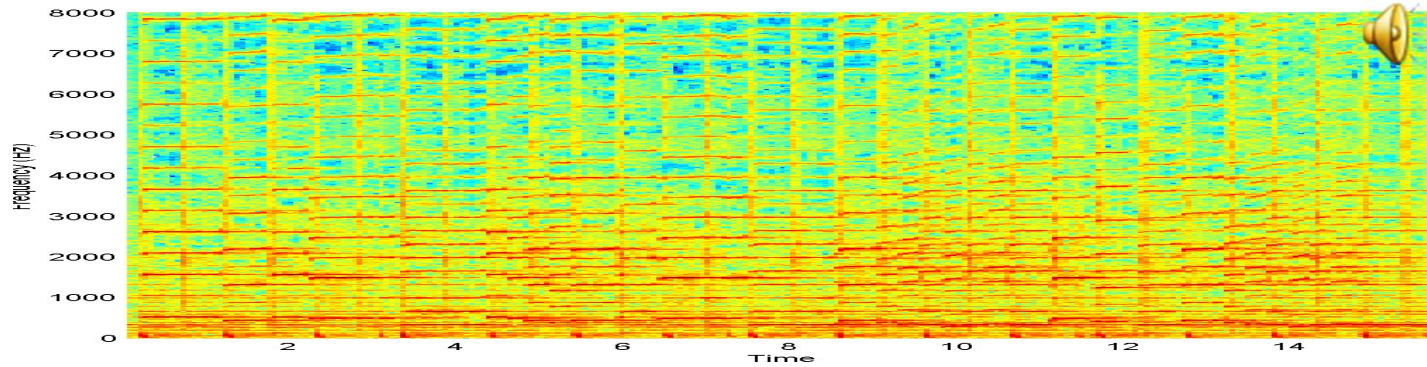


- Floored all matrix values below a threshold to zero



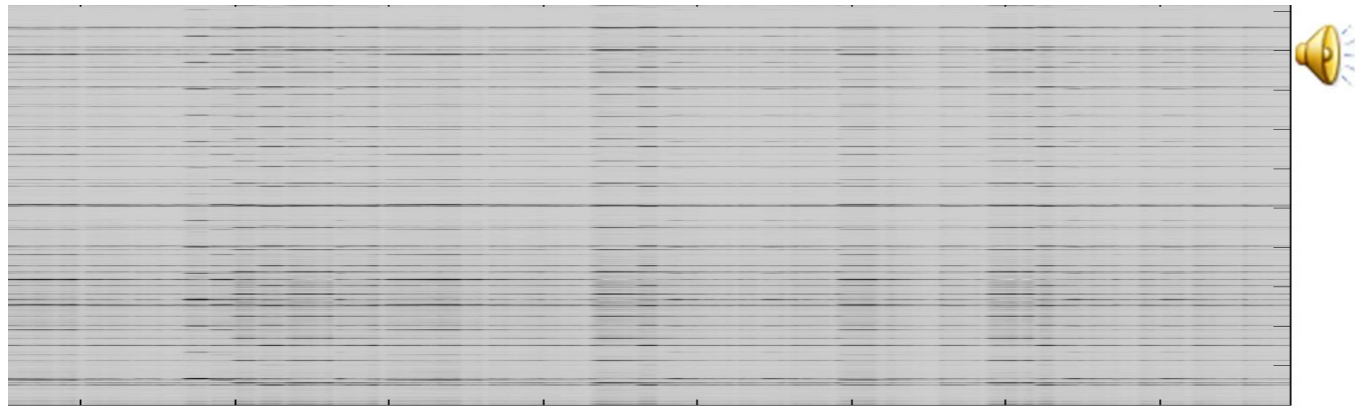
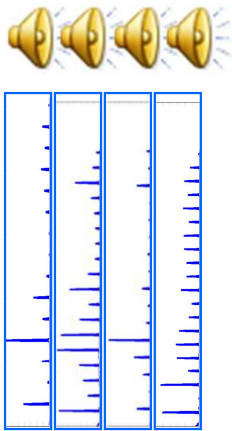
# Projection: multiple notes

M =



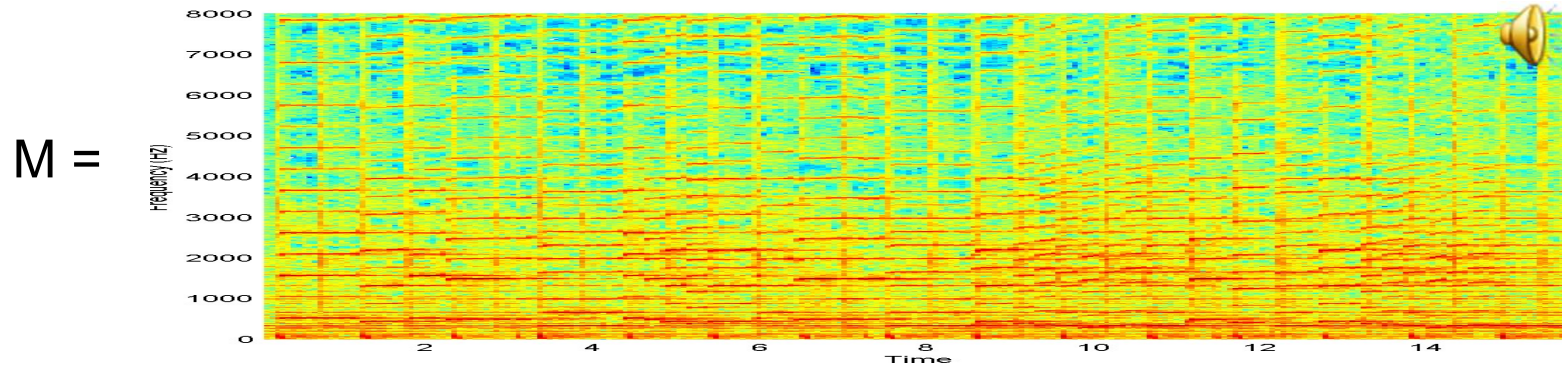
- The spectrogram (matrix) of a piece of music

W =



- $P = W (W^T W)^{-1} W^T$
- Projected Spectrogram =  $P * M$

# Projection: multiple notes, cleaned up



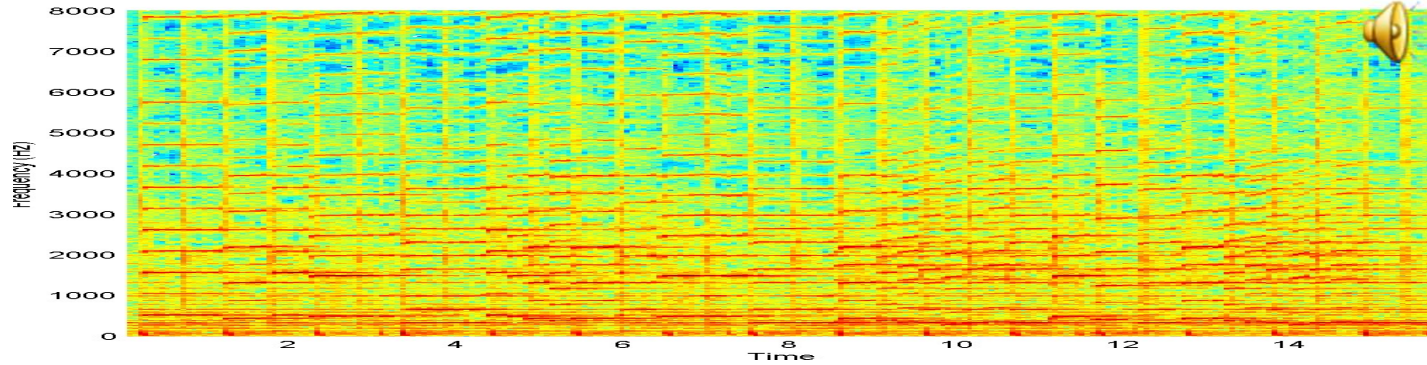
- The spectrogram (matrix) of a piece of music



- $P = W(W^T W)^{-1} W^T$
- Projected Spectrogram =  $PM$

# Projection: one note

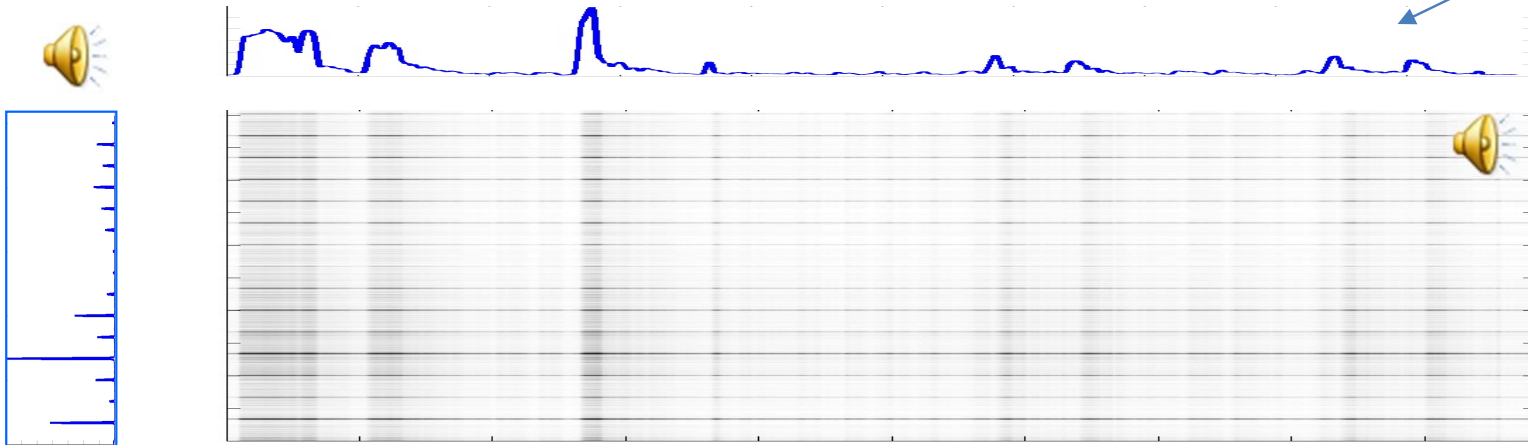
M =



- The spectrogram (matrix) of a piece of music

$$T = W^+ M$$

W =



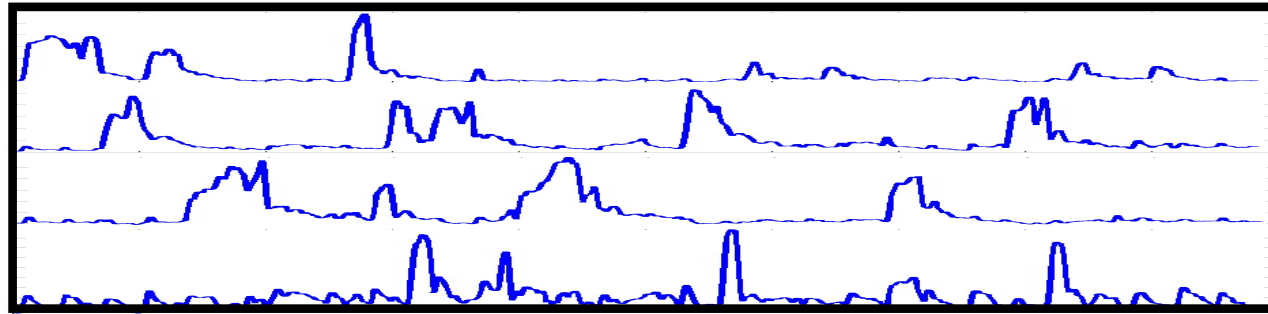
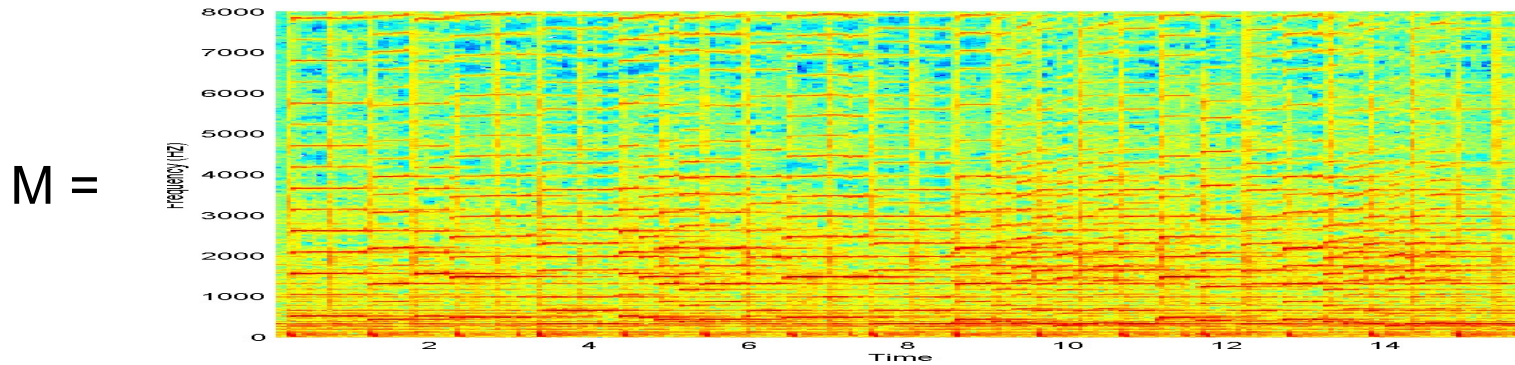
- The “transcription” of the note is

$$T = W^+ M = (W^T W)^{-1} W^T M$$

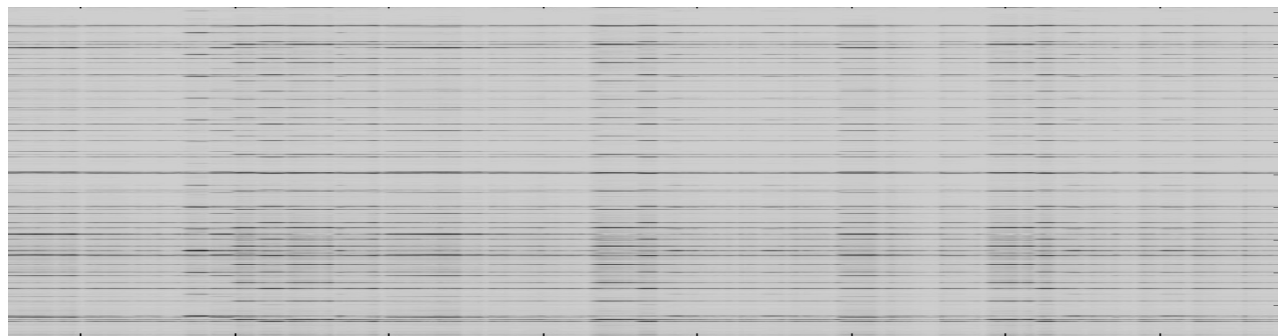
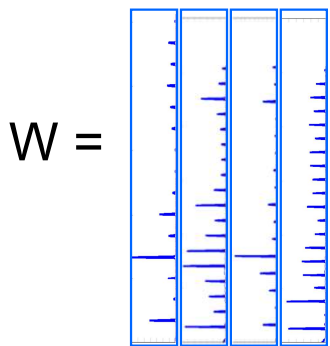
- Projected Spectrogram =  $WT = PM$



# Explanation with multiple notes

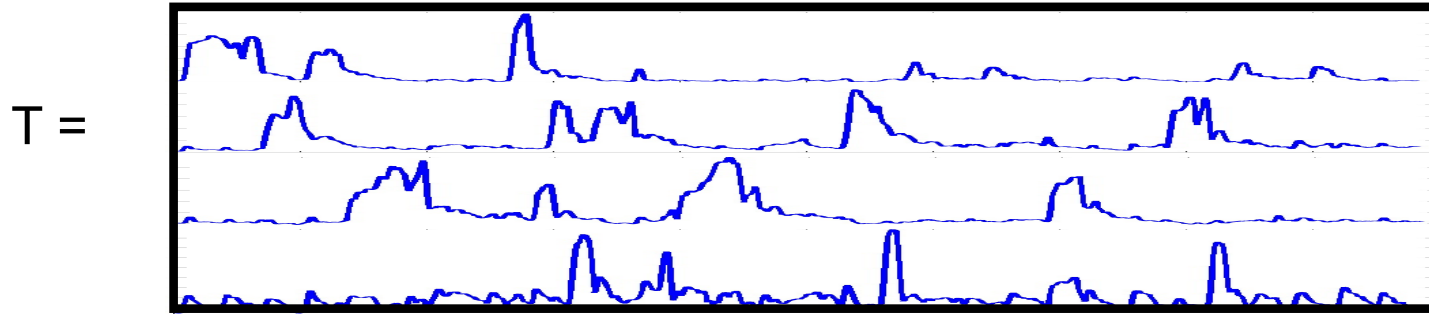
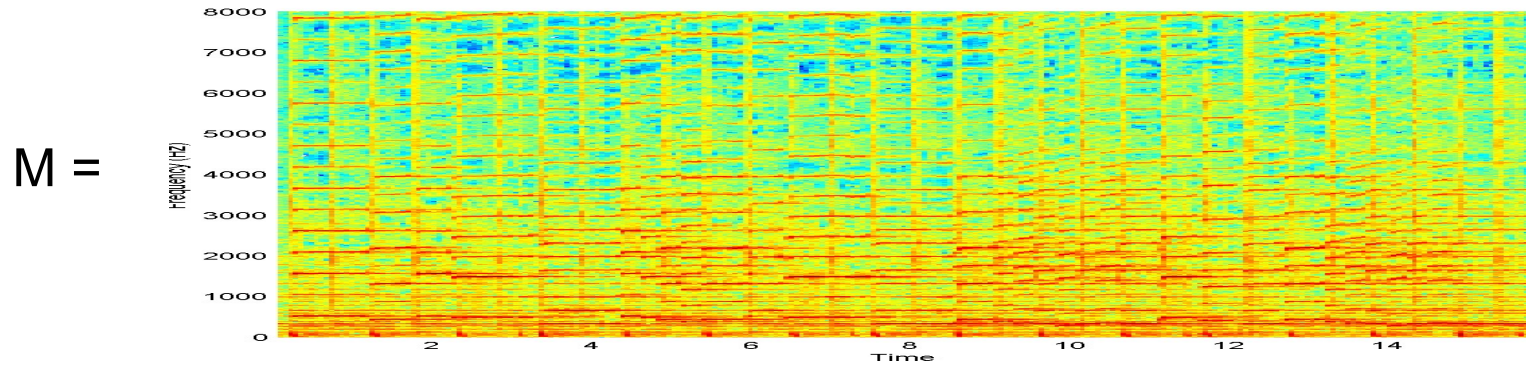


$$T = W^+ M$$



- The “transcription” of the set of notes is
 
$$T = W^+ M = (W^T W)^{-1} W^T M$$
- Projected Spectrogram =  $WT = PM$

# How about the other way?



W = ?

U = ?

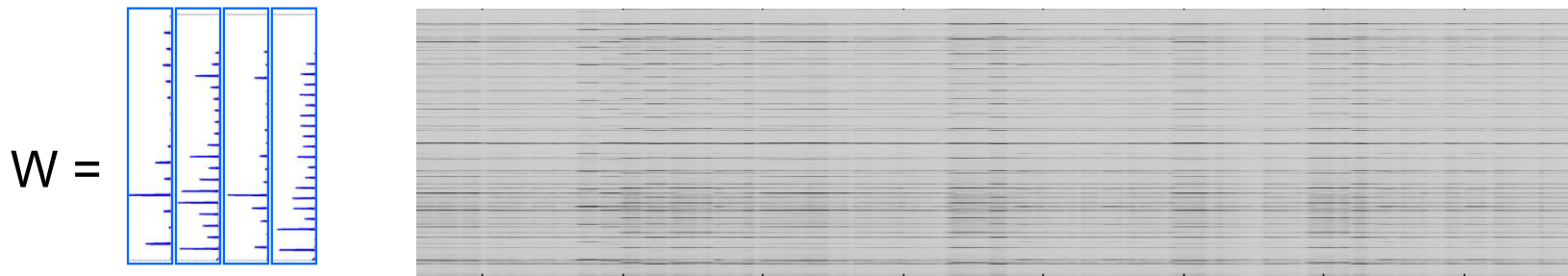
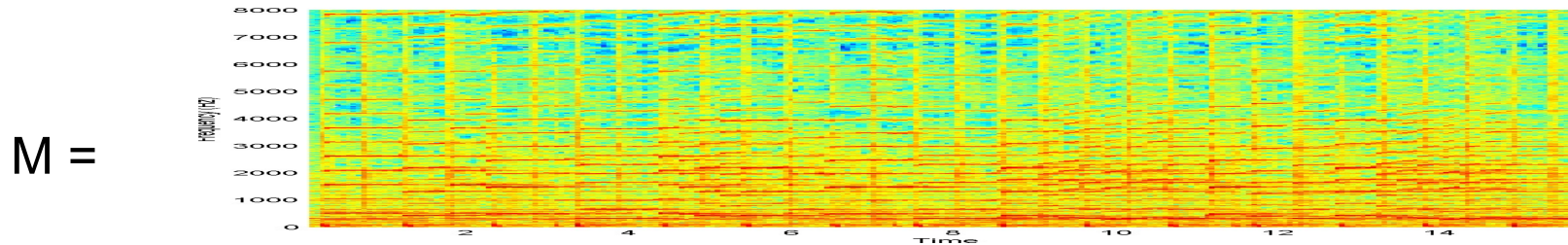
■  $WT \approx M$

$W = M P_{\text{inv}}(T)$

$U = WT$

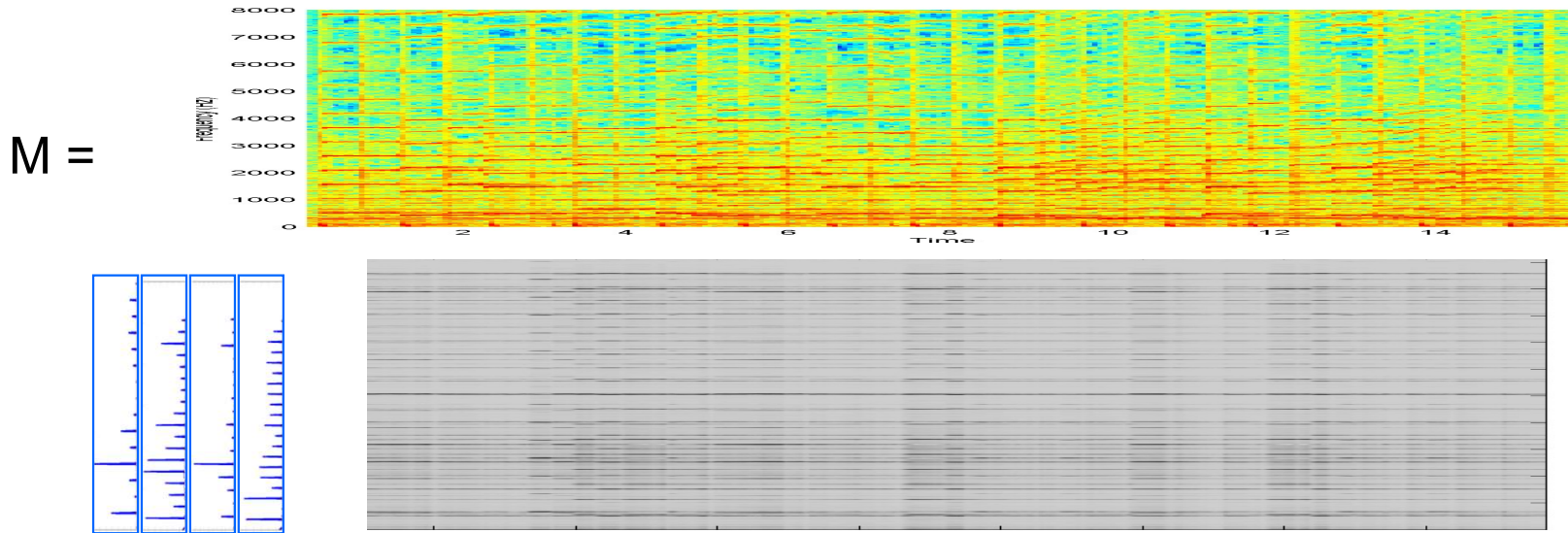


# Projections are often examples of rank-deficient transforms



- $P = W(W^T W)^{-1} W^T$ ; Projected Spectrogram :  $M_{proj} = PM$
- The original spectrogram can never be recovered
  - $P$  is rank deficient
- $P$  explains all vectors in the new spectrogram as a mixture of only the 4 vectors in  $W$ 
  - There are only a maximum of 4 **linearly independent** bases
  - Rank of  $P$  is 4

# The Rank of Matrix



- Projected Spectrogram =  $P M$ 
  - Every vector in it is a combination of only 4 bases
- The rank of the matrix is the *smallest* no. of bases required to describe the output
  - E.g. if note no. 4 in  $P$  could be expressed as a combination of notes 1,2 and 3, it provides no additional information
  - Eliminating note no. 4 would give us the same projection
  - The rank of  $P$  would be 3!



# Pseudo-inverse (PINV)

- $Pinv()$  applies to non-square matrices and non-invertible square matrices
- $Pinv(Pinv(\mathbf{A})) = \mathbf{A}$
- $\mathbf{A}Pinv(\mathbf{A}) =$  projection matrix!
  - Projection onto the columns of  $\mathbf{A}$
- If  $\mathbf{A}$  is a  $K \times N$  matrix and  $K > N$ ,  $\mathbf{A}$  projects  $N$ -dimensional vectors into a higher-dimensional  $K$ -dimensional space
  - $Pinv(\mathbf{A})$  is a  $N \times K$  matrix
  - $Pinv(\mathbf{A})\mathbf{A} = \mathbf{I}$  in this case
- Otherwise  $\mathbf{A}Pinv(\mathbf{A}) = \mathbf{I}$

# Overview

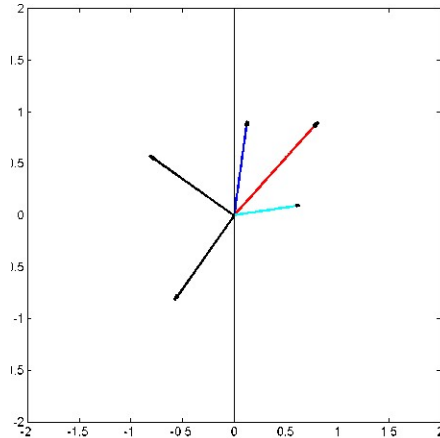
- Vectors and matrices
- Basic vector/matrix operations
- Various matrix types
- Matrix properties
  - Determinant
  - Inverse
  - Rank
- Solving simultaneous equations
- Projections
- **Eigen decomposition**
- **SVD**

# Eigenanalysis

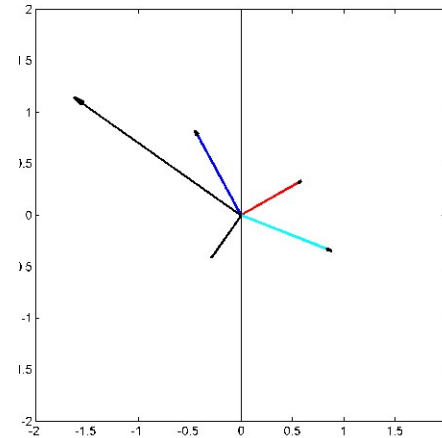
- If something can go through a process mostly unscathed in character it is an *eigen*-something
  - Sound example:  
- A vector that can undergo a matrix multiplication and keep pointing the same way is an *eigenvector*
  - Its length can change though
- How much its length changes is expressed by its corresponding *eigenvalue*
  - Each eigenvector of a matrix has its eigenvalue
- Finding these “eigenthings” is called eigenanalysis

# EigenVectors and EigenValues

Black  
vectors  
are  
eigen  
vectors



$$M = \begin{bmatrix} 1.5 & -0.7 \\ -0.7 & 1.0 \end{bmatrix}$$

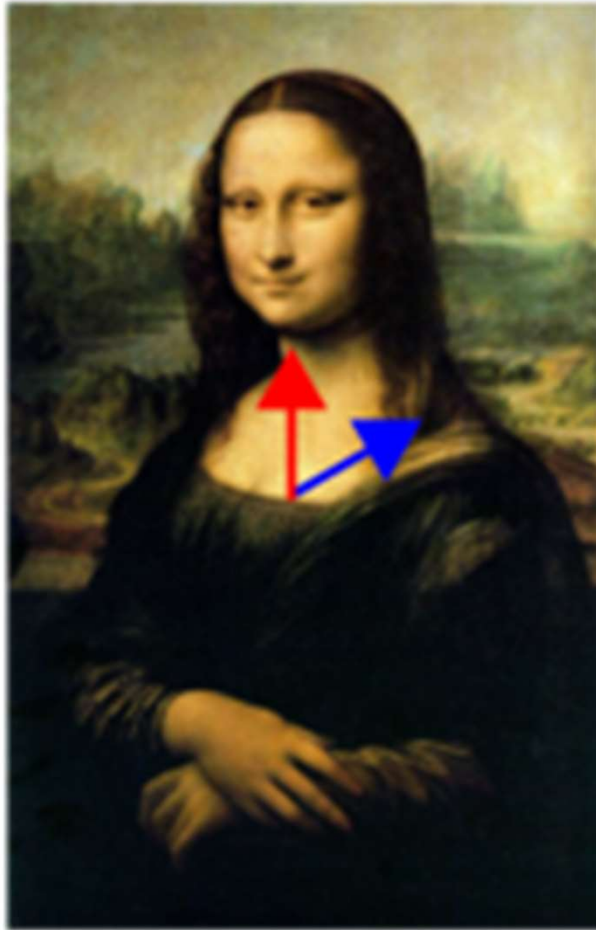


- Vectors that do not change angle upon transformation
  - They may change length

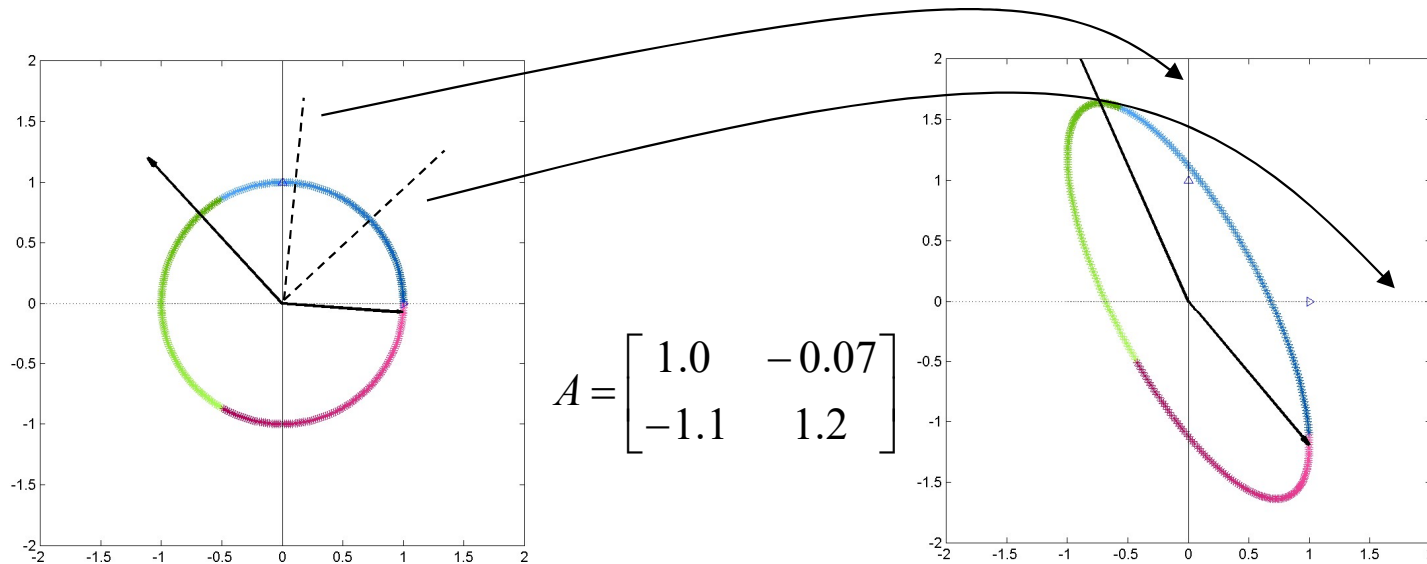
$$MV = \lambda V$$

- $V$  = eigen vector
- $\lambda$  = eigen value

# Eigen vector example



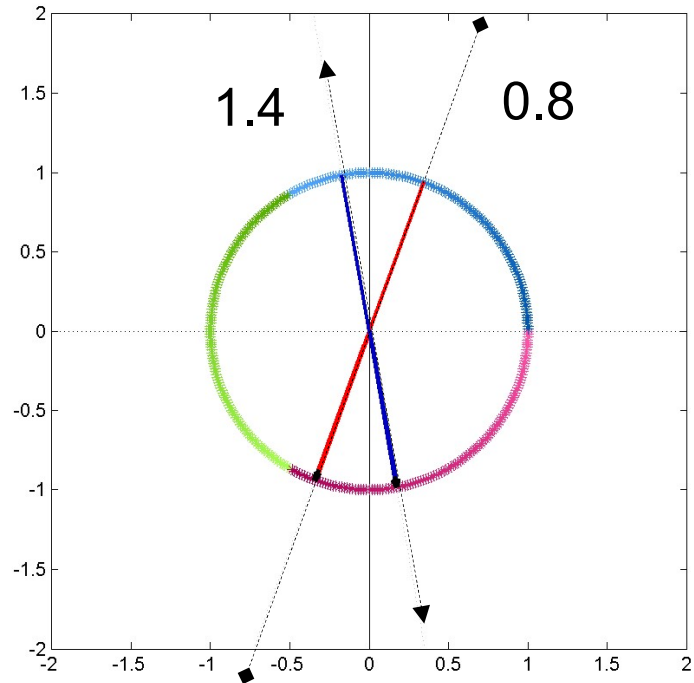
# Matrix multiplication revisited



- Matrix transformation “transforms” the space
  - Warps the paper so that the normals to the two vectors now lie along the axes

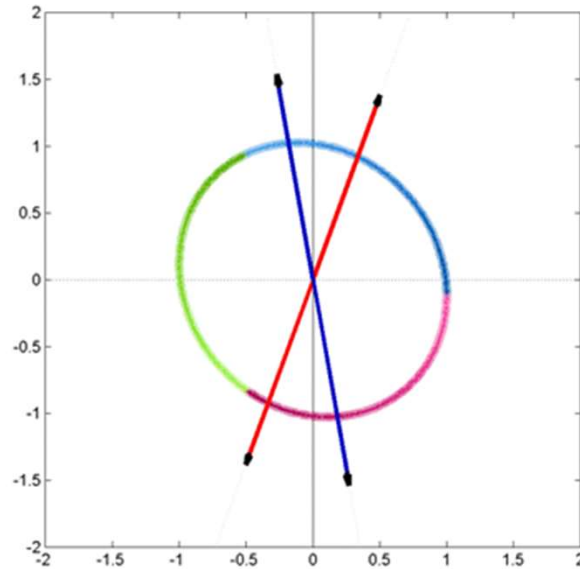


# A stretching operation



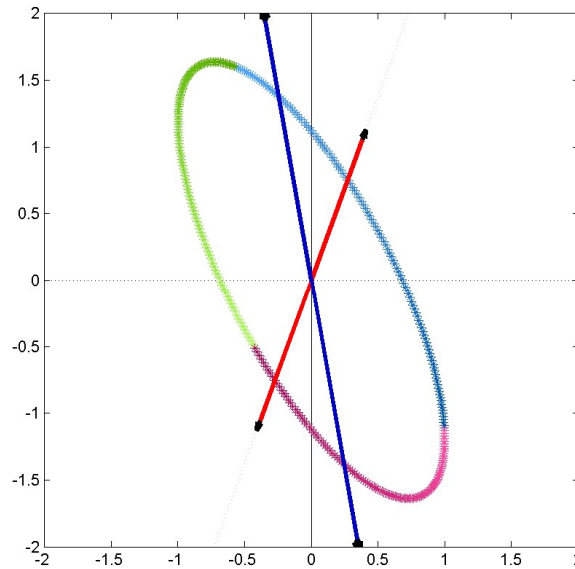
- Draw two lines
- Stretch / shrink the paper along these lines by factors  $\lambda_1$  and  $\lambda_2$ 
  - The factors could be negative – implies flipping the paper
- The result is a transformation of the space

# A stretching operation



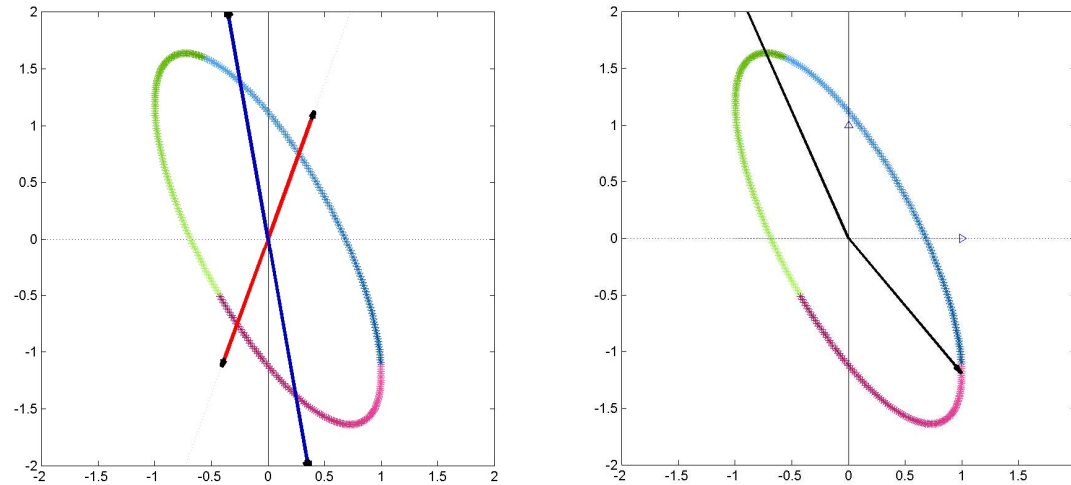
- Draw two lines
- Stretch / shrink the paper along these lines by factors  $\lambda_1$  and  $\lambda_2$ 
  - The factors could be negative – implies flipping the paper
- The result is a transformation of the space

# A stretching operation



- Draw two lines
- Stretch / shrink the paper along these lines by factors  $\lambda_1$  and  $\lambda_2$ 
  - The factors could be negative – implies flipping the paper
- The result is a transformation of the space

# Physical interpretation of eigen vector



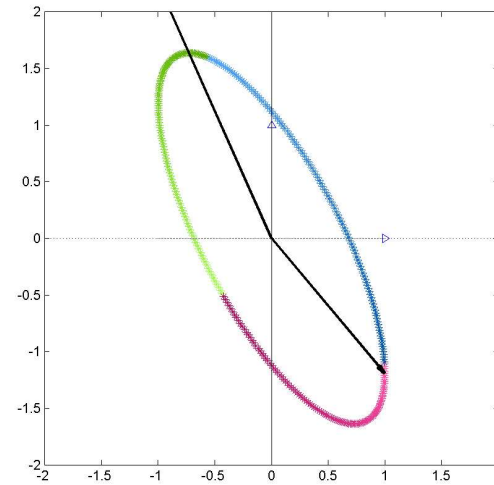
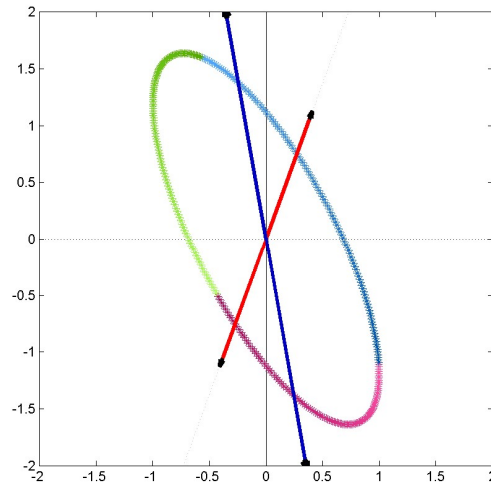
- The result of the stretching is exactly the same as transformation by a matrix
- The axes of stretching/shrinking are the eigenvectors
  - The degree of stretching/shrinking are the corresponding eigenvalues
- The EigenVectors and EigenValues convey all the information about the matrix

# Physical interpretation of eigen vector

$$V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

$$M = V\Lambda V^{-1}$$



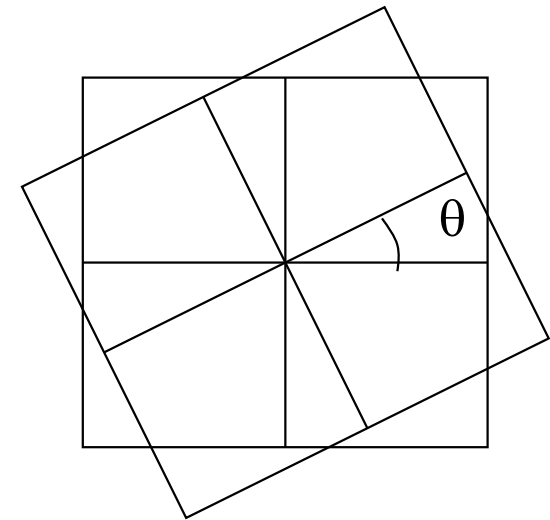
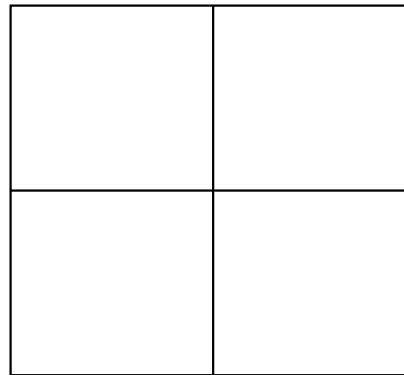
- The result of the stretching is exactly the same as transformation by a matrix
- The axes of stretching/shrinking are the eigenvectors
  - The degree of stretching/shrinking are the corresponding eigenvalues
- The EigenVectors and EigenValues convey all the information about the matrix
- The determinant of the matrix is the product of the eigenvalues

$$|M| = |V||\Lambda||V^{-1}| = C \cdot \prod_i \lambda_i \cdot C^{-1} = \prod_i \lambda_i$$

# Eigen Analysis

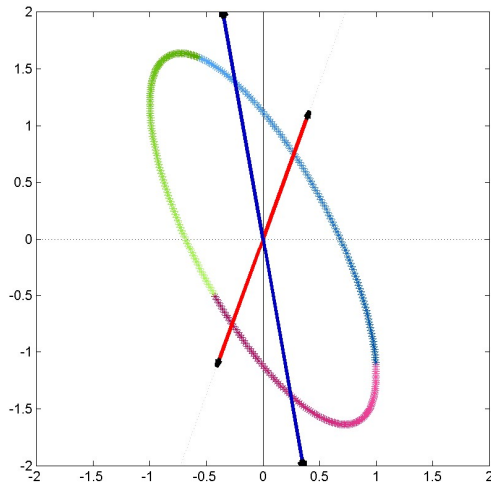
- Not all square matrices have nice eigen values and vectors
  - E.g. consider a rotation matrix

$$\mathbf{R}_\theta = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$
$$X = \begin{bmatrix} x \\ y \end{bmatrix}$$
$$X_{new} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

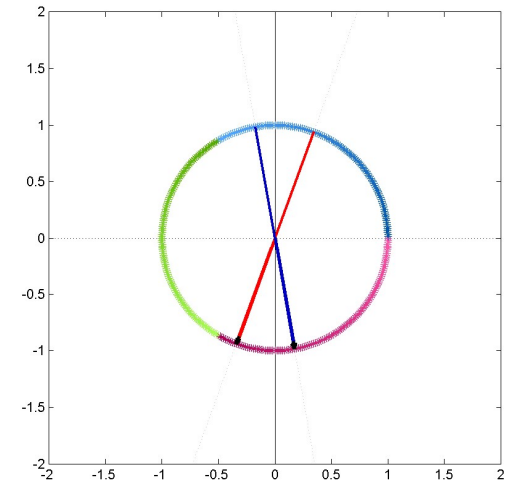
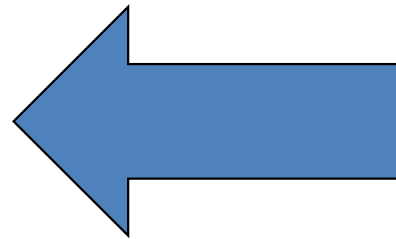


- This rotates every vector in the plane
  - No vector that remains unchanged
- In these cases the Eigen vectors and values are complex
  - Actually complex conjugate pairs

# Singular Value Decomposition

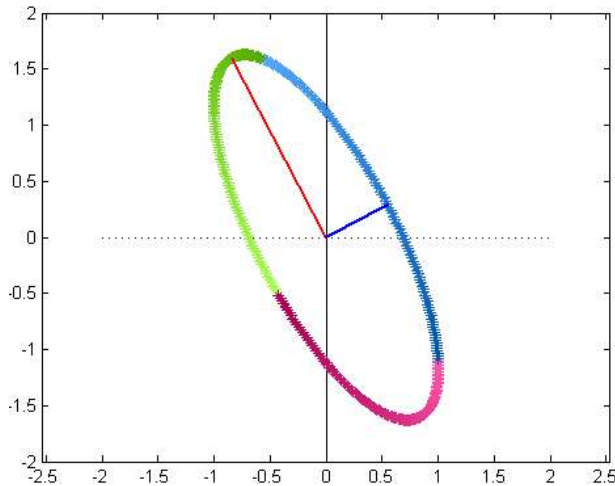


$$A = \begin{bmatrix} 1.0 & -0.07 \\ -1.1 & 1.2 \end{bmatrix}$$

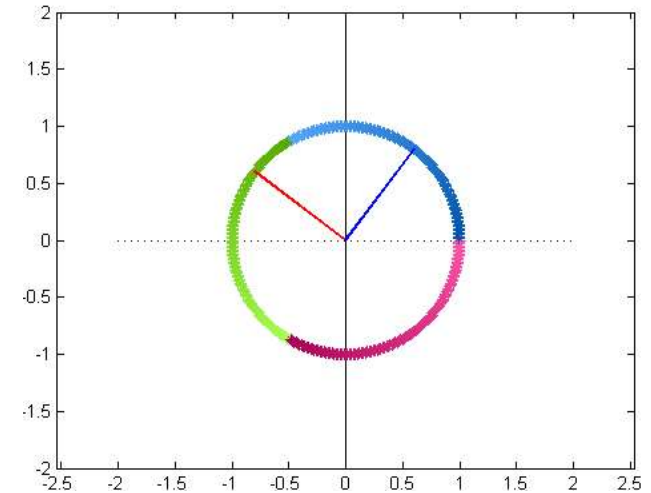
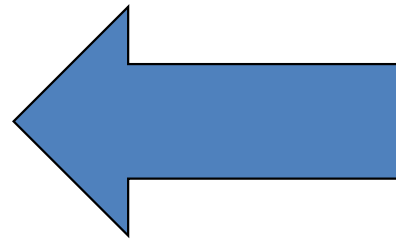


- Matrix transformations convert circles to ellipses
- Eigen vectors are vectors that do not change direction in the process
- There is another key feature of the ellipse to the left that carries information about the transform
  - Can you identify it?

# Singular Value Decomposition



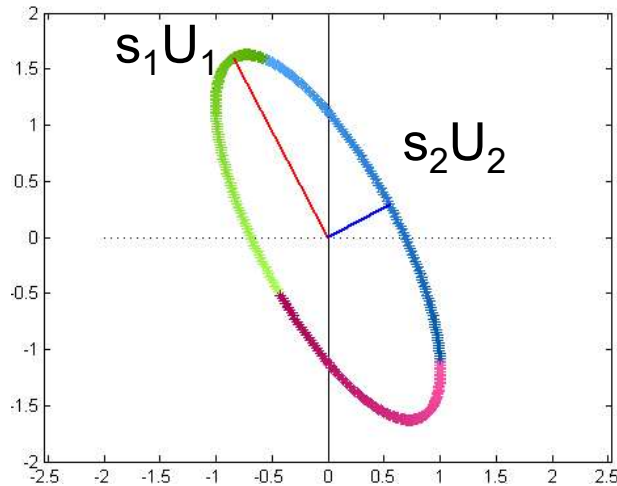
$$A = \begin{bmatrix} 1.0 & -0.07 \\ -1.1 & 1.2 \end{bmatrix}$$



- The major and minor axes of the transformed ellipse define the ellipse
  - They are at right angles
- These are transformations of right-angled vectors on the original circle!



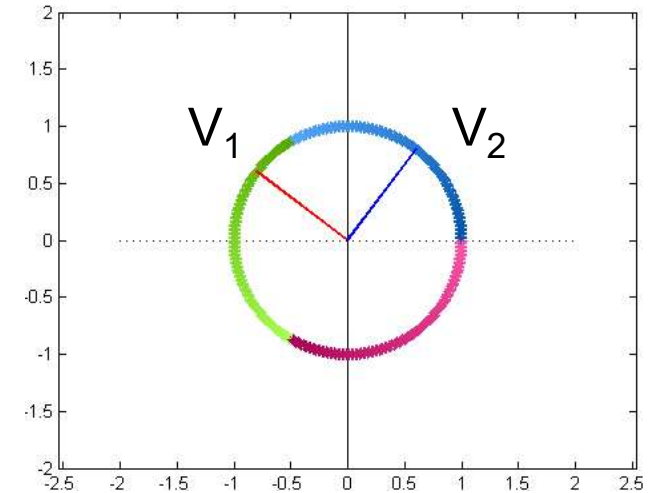
# Singular Value Decomposition



$$A = \begin{bmatrix} 1.0 & -0.07 \\ -1.1 & 1.2 \end{bmatrix}$$

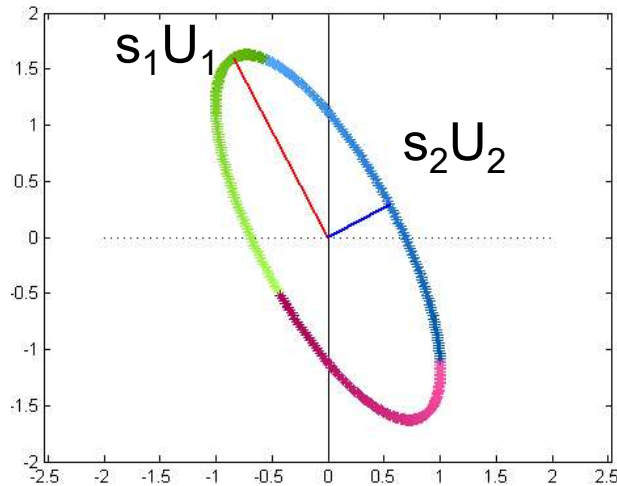
$$A = U S V^T$$

matlab:  
`[U,S,V] = svd(A)`



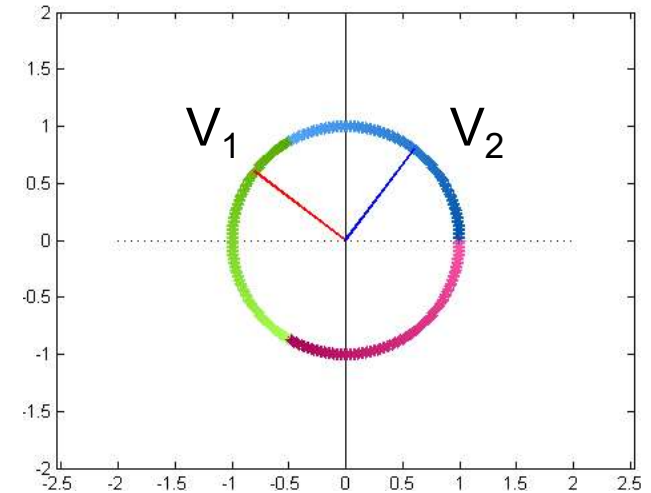
- $U$  and  $V$  are orthonormal matrices
  - Columns are orthonormal vectors
- $S$  is a diagonal matrix
- The *right singular vectors* in  $V$  are transformed to the *left singular vectors* in  $U$ 
  - And scaled by the *singular values* that are the diagonal entries of  $S$

# Singular Value Decomposition



$$A = U S V^T$$

$$A^T = V S U^T$$

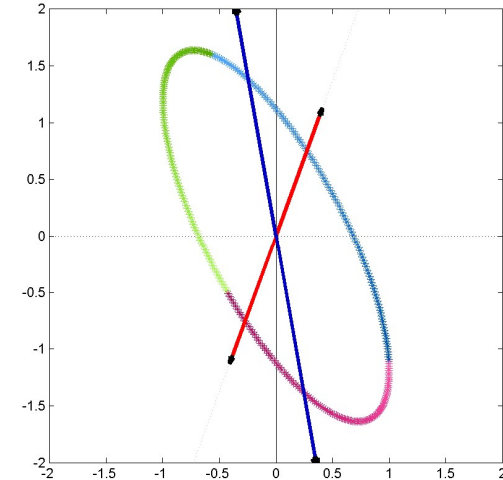
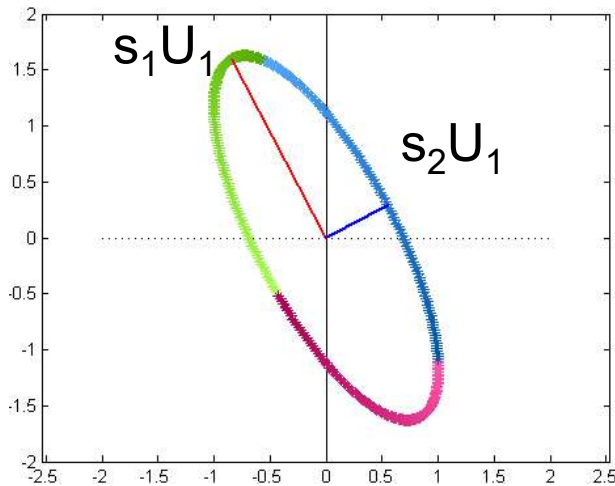


- A matrix  $A$  converts *right* singular vectors  $V$  to *left* singular vectors  $U$
- $A^T$  converts  $U$  to  $V$

# Singular Value Decomposition

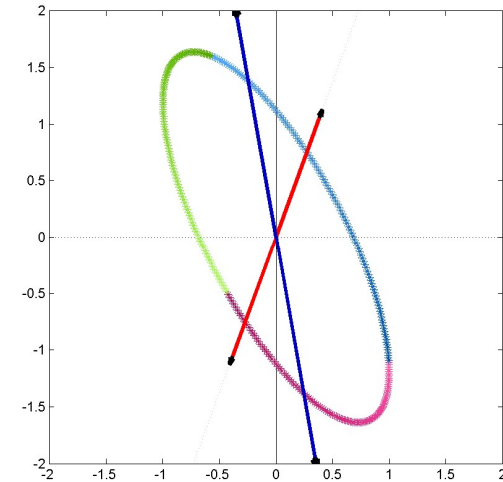
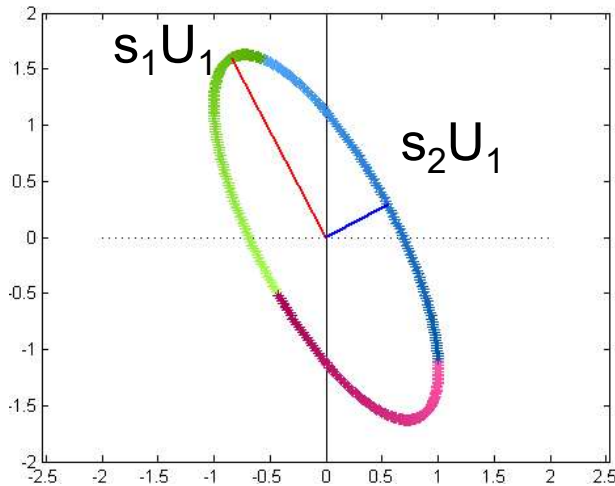
- The left and right singular vectors are not the same
  - If  $A$  is not a square matrix, the left and right singular vectors will be of different dimensions
- The singular values are always real
- The largest singular value is the largest amount by which a vector is scaled by  $A$ 
  - $\text{Max} (|Ax| / |x|) = s_{\text{max}}$
- The smallest singular value is the smallest amount by which a vector is scaled by  $A$ 
  - $\text{Min} (|Ax| / |x|) = s_{\text{min}}$
  - This can be 0 (for low-rank or non-square matrices)

# The Singular Values



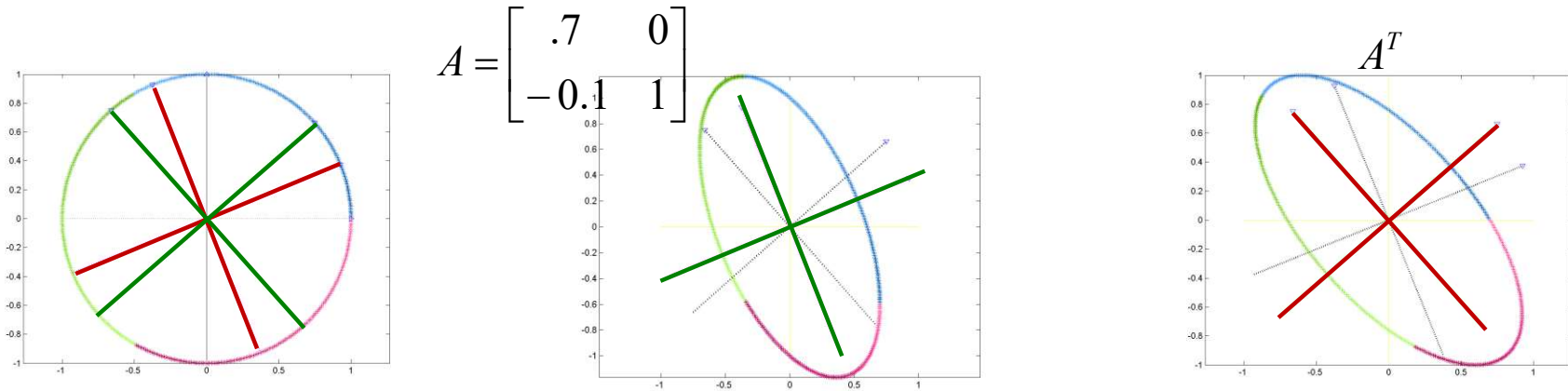
- Square matrices: product of singular values = determinant of the matrix
  - This is also the product of the *eigen* values
  - I.e. there are two different sets of axes whose products give you the area of an ellipse
- For any “broad” rectangular matrix  $A$ , the largest singular value of any square submatrix  $B$  cannot be larger than the largest singular value of  $A$ 
  - An analogous rule applies to the smallest singular value
  - This property is utilized in various problems

# SVD vs. Eigen Analysis



- Eigen analysis of a matrix **A**:
  - Find vectors such that their absolute directions are not changed by the transform
- SVD of a matrix **A**:
  - Find orthogonal set of vectors such that the *angle* between them is not changed by the transform
- For one class of matrices, these two operations are the same

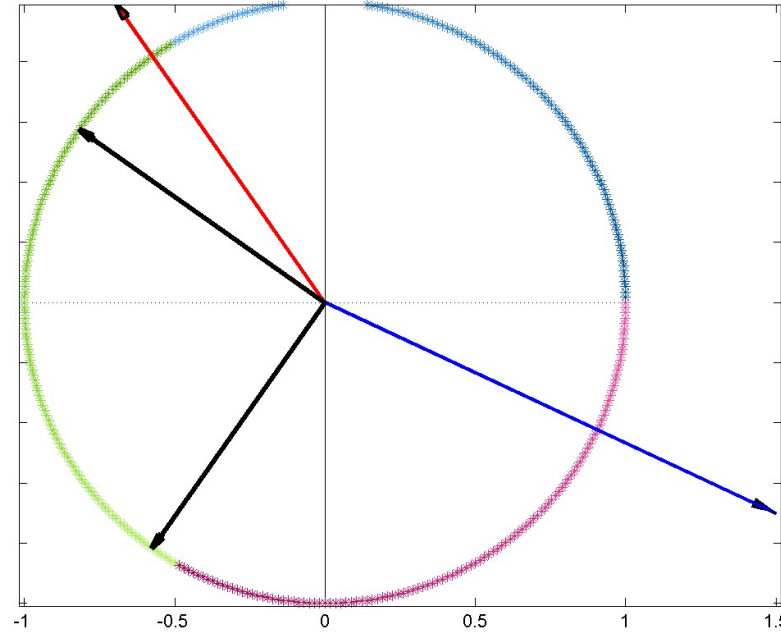
# A matrix vs. its transpose



- Multiplication by matrix  $A$ :
  - Transforms right singular vectors in  $V$  to left singular vectors  $U$
- Multiplication by its transpose  $A^T$ :
  - Transforms *left* singular vectors  $U$  to right singular vector  $V$
- $A A^T$  : Converts  $V$  to  $U$ , then brings it back to  $V$ 
  - Result: Only scaling

# Symmetric Matrices

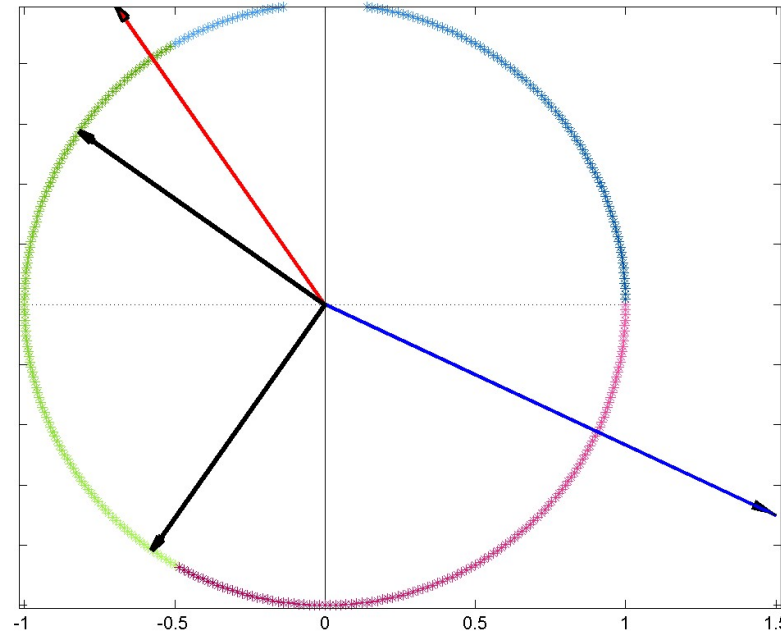
$$\begin{bmatrix} 1.5 & -0.7 \\ -0.7 & 1 \end{bmatrix}$$



- Matrices that do not change on transposition
  - Row and column vectors are identical
- The left and right singular vectors are identical
  - $U = V$
  - $A = U S U^T$
- They are identical to the *Eigen vectors* of the matrix
- **Symmetric matrices do not rotate the space**
  - Only scaling and, if Eigen values are negative, reflection

# Symmetric Matrices

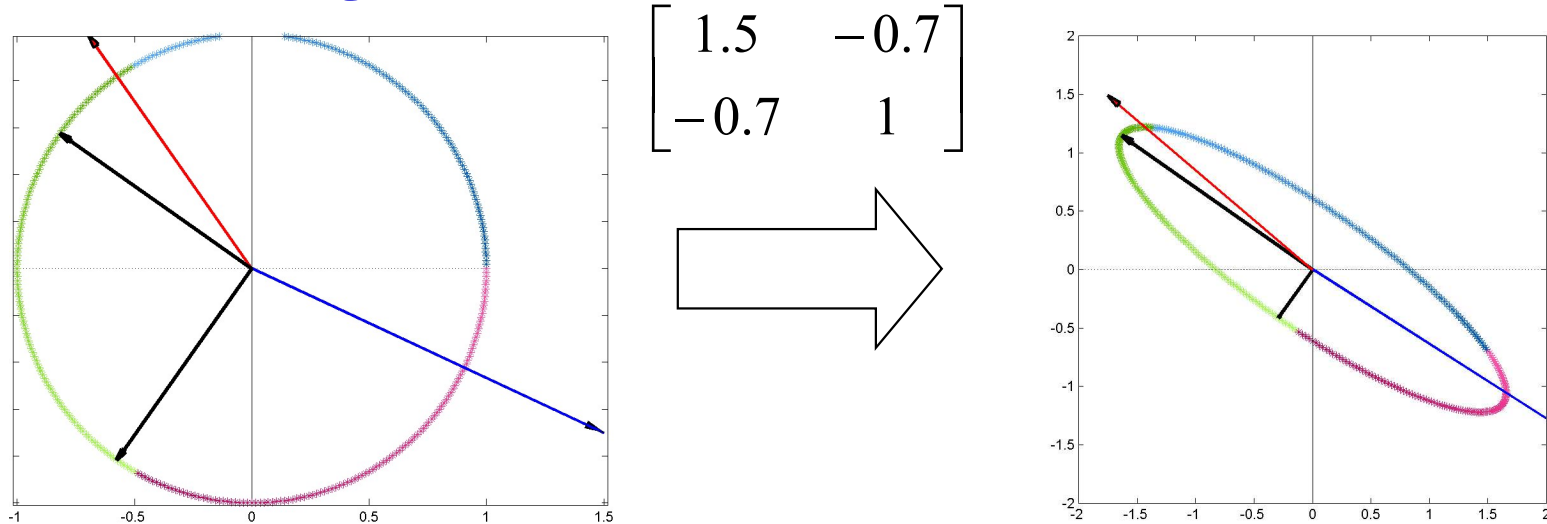
$$\begin{bmatrix} 1.5 & -0.7 \\ -0.7 & 1 \end{bmatrix}$$



- Matrices that do not change on transposition
  - Row and column vectors are identical
- Symmetric matrix: Eigen vectors and Eigen values are always real
- Eigen vectors are always orthogonal
  - At 90 degrees to one another



# Symmetric Matrices



- Eigen vectors point in the direction of the major and minor axes of the ellipsoid resulting from the transformation of a spheroid
  - The eigen values are the lengths of the axes

# Symmetric matrices

- Eigen vectors  $V_i$  are orthonormal
  - $V_i^T V_i = 1$
  - $V_i^T V_j = 0, i \neq j$
- Listing all eigen vectors in matrix form  $V$ 
  - $V^T = V^{-1}$
  - $V^T V = I$
  - $V V^T = I$
- $M V_i = \lambda V_i$
- In matrix form :  $M V = V \Lambda$ 
  - $\Lambda$  is a diagonal matrix with all eigen values
- $M = V \Lambda V^T$

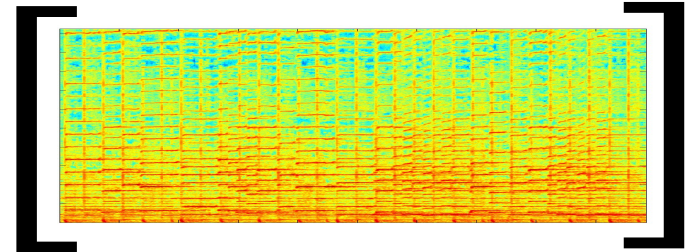
# Definiteness..

- SVD: Singular values are always positive!
- Eigen Analysis: Eigen values can be real or imaginary
  - Real, positive Eigen values represent stretching of the space along the Eigen vector
  - Real, *negative* Eigen values represent stretching and *reflection* (across origin) of Eigen vector
  - Complex Eigen values occur in conjugate pairs
- A square (symmetric) matrix is **positive definite** if all Eigen values are real and positive, and are greater than 0
  - Transformation can be explained as **stretching** along orthogonal axes
    - Transformation has no permutation or rotation
  - If any Eigen value is **zero**, the matrix is positive *semi-definite*

# Positive Definiteness..

- Property of a positive definite matrix: Defines inner product norms
  - $\mathbf{x}^T \mathbf{A} \mathbf{x}$  is always positive for any vector  $\mathbf{x}$  if  $\mathbf{A}$  is positive definite
- Positive definiteness is a test for validity of *Gram* matrices
  - Such as correlation and covariance matrices
  - We will encounter these and other gram matrices later

# SVD on data-container matrices

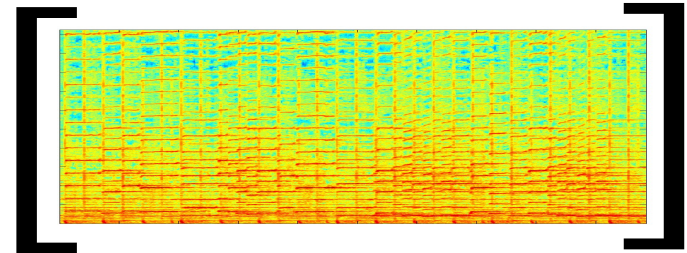


$$\mathbf{X} = [X_1 \ X_2 \ \cdots \ X_N]$$

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

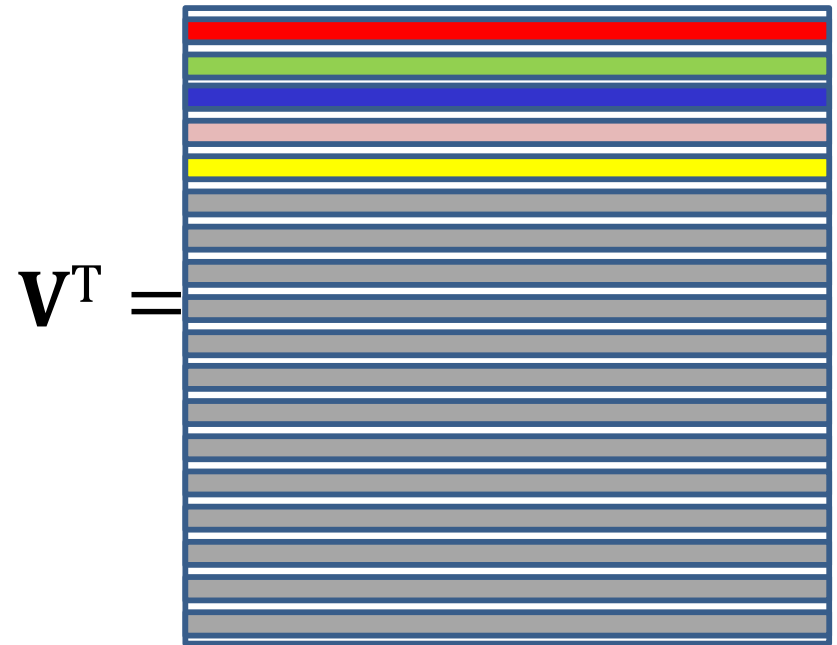
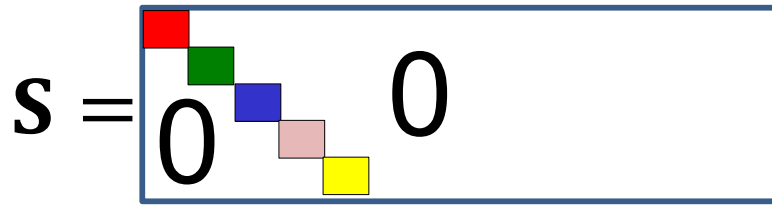
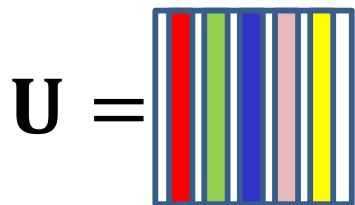
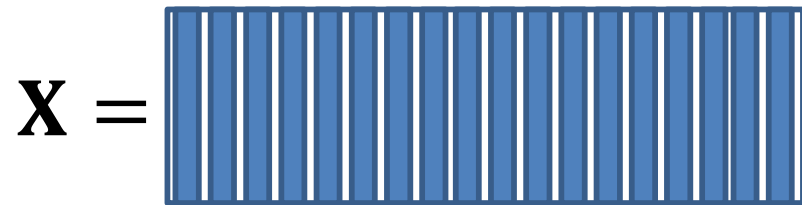
- We can also perform SVD on matrices that are *data containers*
- $\mathbf{S}$  is a  $d \times N$  rectangular matrix
  - $N$  vectors of dimension  $d$
- $\mathbf{U}$  is an orthogonal matrix of  $d$  vectors of size  $d$ 
  - All vectors are length 1
- $\mathbf{V}$  is an orthogonal matrix of  $N$  vectors of size  $N$
- $\mathbf{S}$  is a  $d \times N$  diagonal matrix with non-zero entries only on diagonal

# SVD on data-container matrices



$$\mathbf{X} = [X_1 \ X_2 \ \dots \ X_N]$$

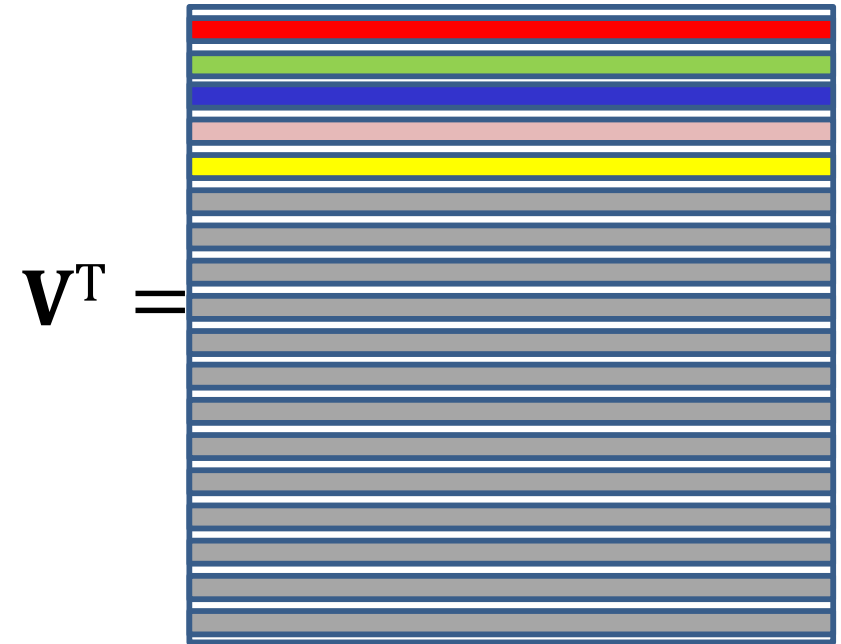
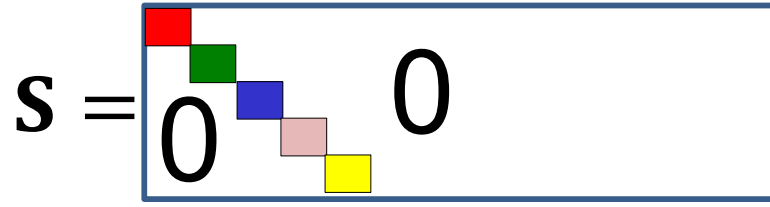
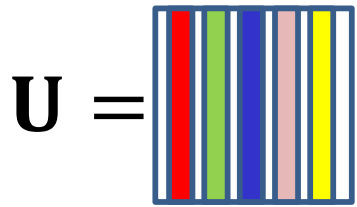
$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$



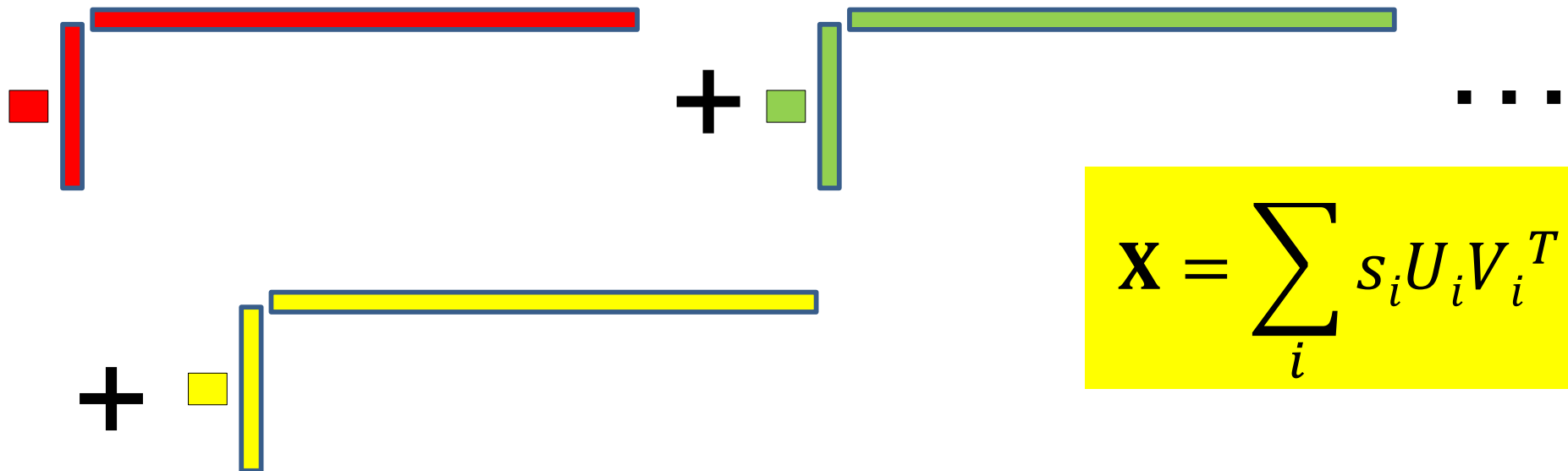
$|U_i| = 1.0$  for every vector in  $\mathbf{U}$

$|V_i| = 1.0$  for every vector in  $\mathbf{V}$

# SVD on data-container matrices

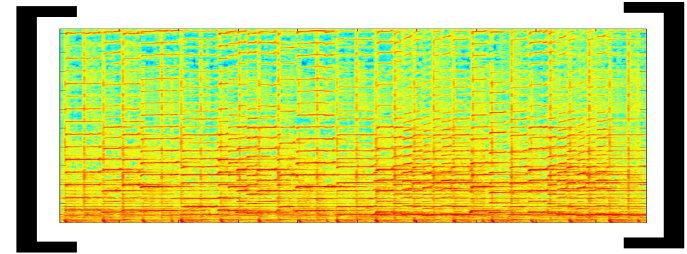


$X = USV^T =$



$$X = \sum_i s_i U_i V_i^T$$

# Expanding the SVD



$$\mathbf{X} = [X_1 \ X_2 \ \cdots \ X_N]$$

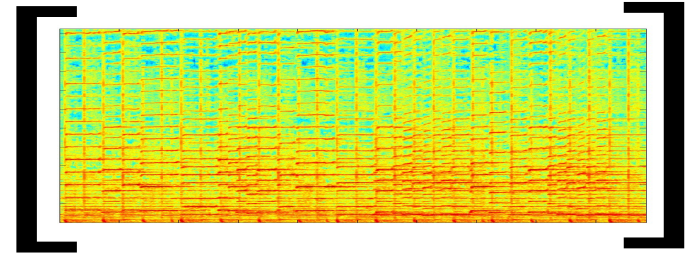
$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

$$\mathbf{X} = s_1 U_1 V_1^T + s_2 U_2 V_2^T + s_3 U_3 V_3^T + s_4 U_4 V_4^T + \dots$$

- Each left singular vector and the corresponding right singular vector contribute on “basic” component to the data
- The “magnitude” of its contribution is the corresponding singular value



# Expanding the SVD



$$\mathbf{X} = [X_1 \ X_2 \ \cdots \ X_N]$$

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

$$\mathbf{X} = s_1 U_1 V_1^T + s_2 U_2 V_2^T + s_3 U_3 V_3^T + s_4 U_4 V_4^T + \dots$$

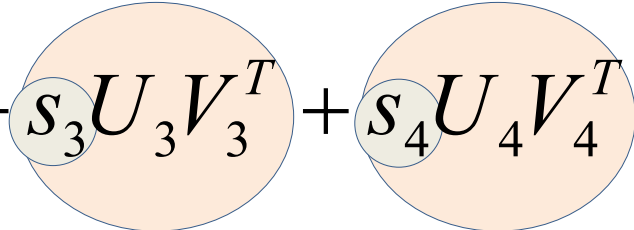
magnitude

basis

modulation

- Each left singular vector and the corresponding right singular vector contribute on “basic” component to the data
- The “magnitude” of its contribution is the corresponding singular value

# Expanding the SVD

$$\mathbf{X} = s_1 U_1 V_1^T + s_2 U_2 V_2^T + s_3 U_3 V_3^T + s_4 U_4 V_4^T + \dots$$


- Each left singular vector and the corresponding right singular vector contribute on “basic” component to the data
- The “magnitude” of its contribution is the corresponding singular value
- Low singular-value components contribute little, if anything
  - Carry little information
  - Are often just “noise” in the data

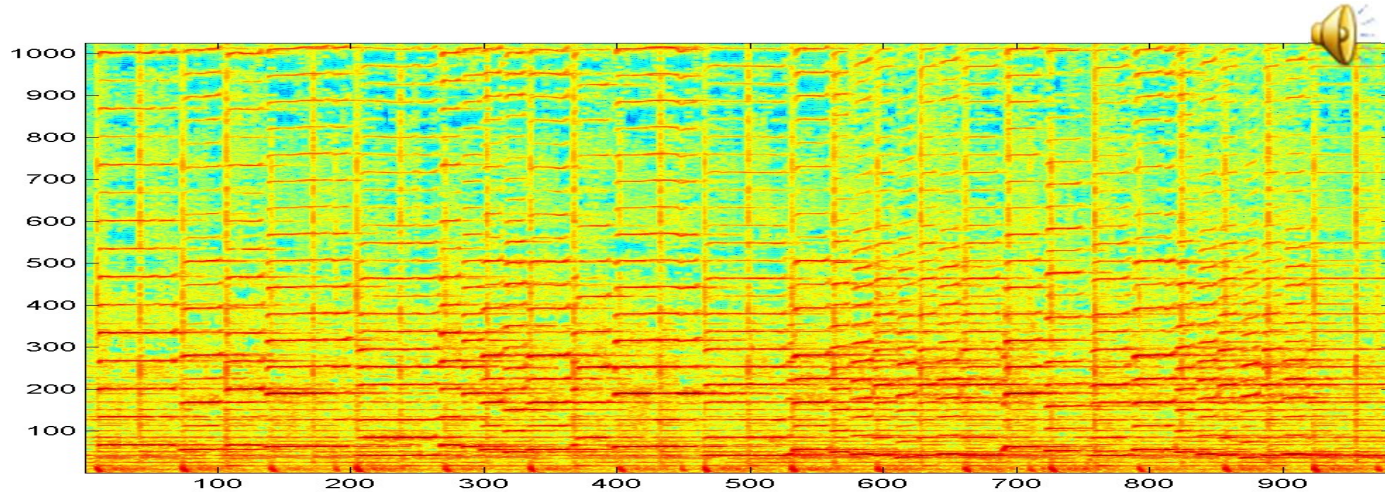
# Expanding the SVD

$$\mathbf{X} = s_1 U_1 V_1^T + s_2 U_2 V_2^T + s_3 U_3 V_3^T + s_4 U_4 V_4^T + \dots$$

$$\mathbf{X} \approx s_1 U_1 V_1^T + s_2 U_2 V_2^T$$

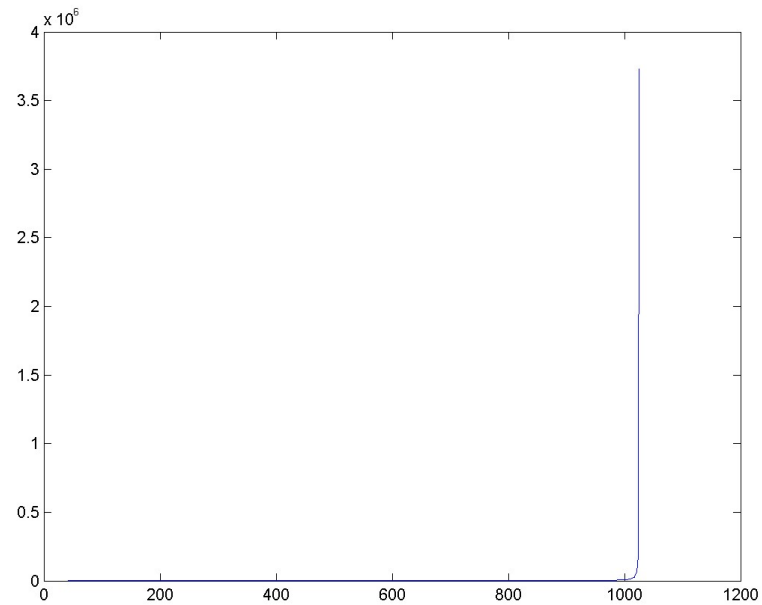
- Low singular-value components contribute little, if anything
  - Carry little information
  - Are often just “noise” in the data
- Data can be recomposed using only the “major” components with minimal change of value
  - Minimum squared error between original data and recomposed data
  - Sometimes eliminating the low-singular-value components will, in fact “clean” the data

# An audio example



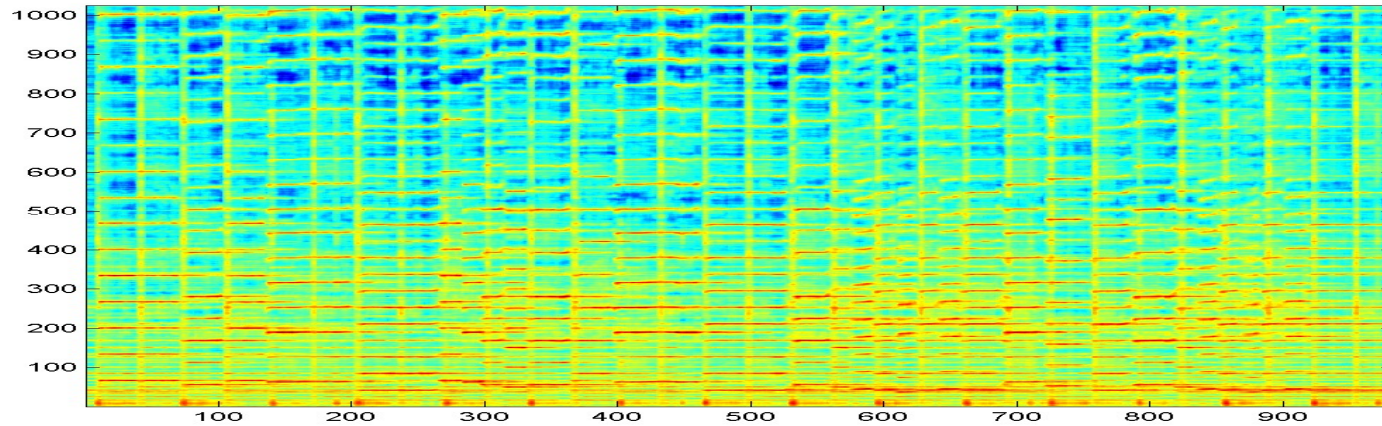
- The spectrogram has 974 vectors of dimension 1025
  - A 1024x974 matrix!
- Decompose:  $\mathbf{M} = \mathbf{USV}^T = \sum_i s_i \mathbf{U}_i \mathbf{V}_i^T$
- $\mathbf{U}$  is 1024 x 1024
- $\mathbf{V}$  is 974 x 974
- There are 974 non-zero singular values  $S_i$

# Singular Values



- Singular values for spectrogram **M**
  - Most Singular values are close to zero
  - The corresponding components are “unimportant”

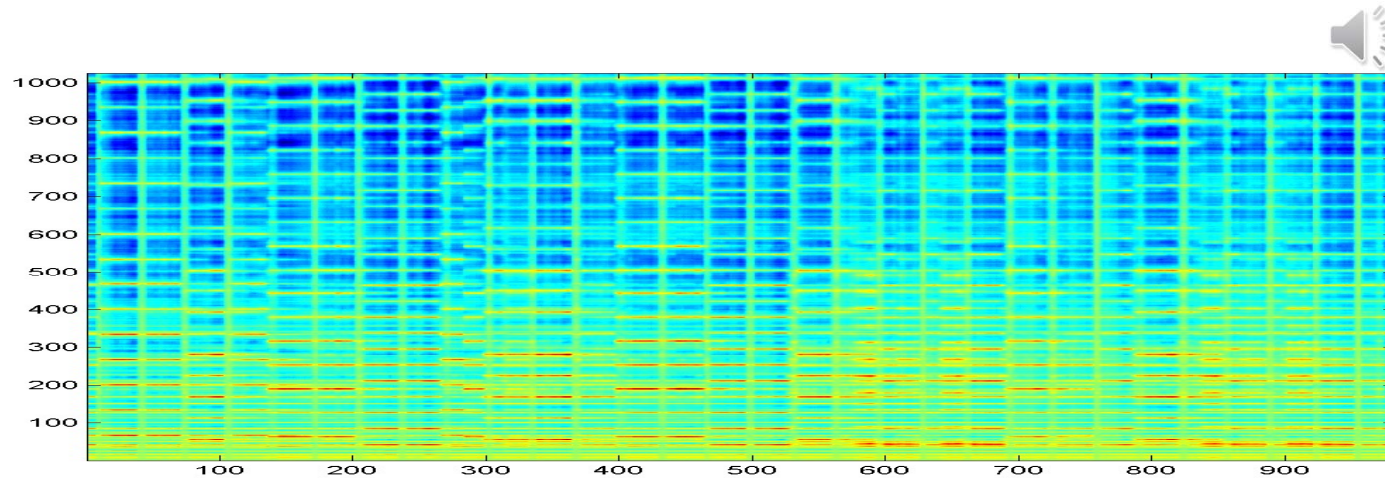
# An audio example



- The same spectrogram constructed from only the 25 highest singular-value components
  - Looks similar
    - With 100 components, it would be indistinguishable from the original
  - Sounds pretty close
  - Background “cleaned up”



# With only 5 components



- The same spectrogram constructed from only the 5 highest-valued components
  - Corresponding to the 5 largest singular values
  - Highly recognizable
  - Suggests that there are actually only 5 significant unique note combinations in the music

- Next up: A brief trip through optimization..