Fundamentals of Linear Algebra

Class 2-3. 1 Sep 2011

Instructor: Bhiksha Raj

Administrivia

- Registration: Anyone on waitlist still?

- Homework 1: Will be handed out with class 3.
  - Linear algebra

Overview

- Vectors and matrices
- Basic vector/matrix operations
- Vector products
- Matrix products
- Various matrix types
- Matrix inversion
- Matrix interpretation
- Eigenanalysis
- Singular value decomposition

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, particularly when applied to images and sound
- Today's lecture: Definitions
  - Very small subset of all that's used
  - Important subset, intended to help you recollect

Incentive to use linear algebra

- Pretty 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*
  - Operations easy to describe verbally
- Easy code translation!

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

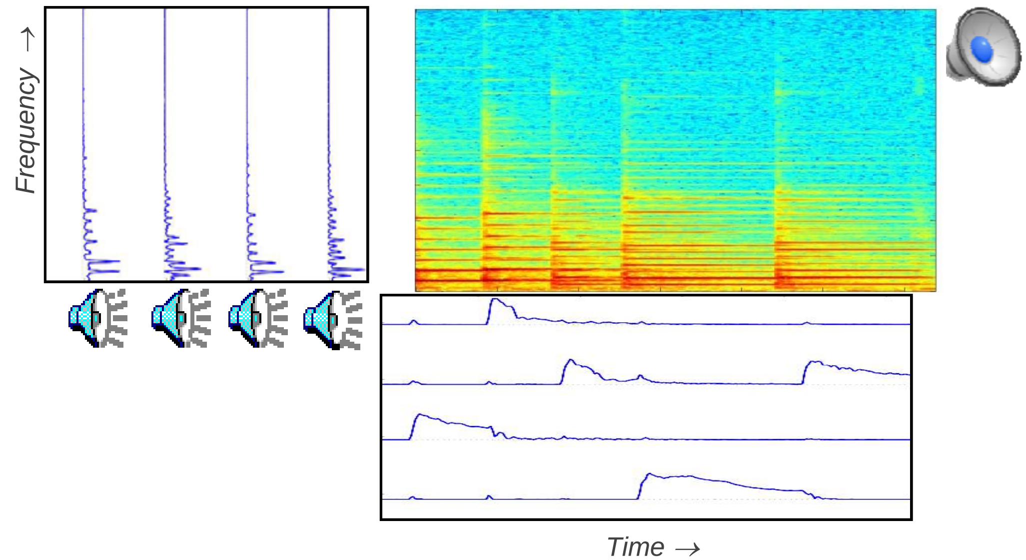
$\longleftrightarrow$

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

And other things you can do

Rotation + Projection + Scaling

From Bach's Fugue in Gm



Decomposition (NMF)

- Manipulate Images
- Manipulate Sounds

Scalars, vectors, matrices, …

- A *scalar* a is a number
  - a = 2, a = 3.14, a = -1000, etc.
- A *vector* **a** is a linear arrangement of a collection of scalars

$$\mathbf{a} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}, \quad \mathbf{a} = \begin{bmatrix} 3.14 \\ -32 \end{bmatrix}$$

- A *matrix* **A** is a rectangular arrangement of a collection of vectors

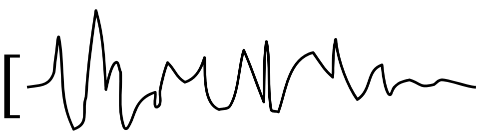
$$\mathbf{A} = \begin{bmatrix} 3.12 & -10 \\ 10.0 & 2 \end{bmatrix}$$

- MATLAB syntax: `a=[1 2 3], A=[1 2;3 4]`

Vector/Matrix types and shapes

- Vectors are either column or row vectors

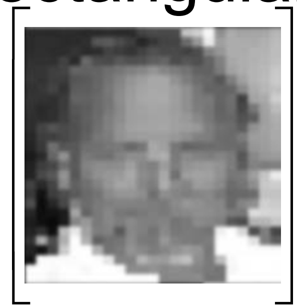
$$\mathbf{c} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad \mathbf{r} = \begin{bmatrix} a & b & c \end{bmatrix}, \quad \mathbf{s} = [$$



$$]$$

  - A sound can be a vector, a series of daily temperatures can be a vector, etc …
- Matrices can be square or rectangular

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



  - Images can be a matrix, collections of sounds can be a matrix, etc …

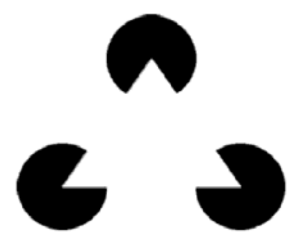
Dimensions of a matrix

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

$$\mathbf{c} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \ \mathbf{r} = \begin{bmatrix} a & b & c \end{bmatrix}$$

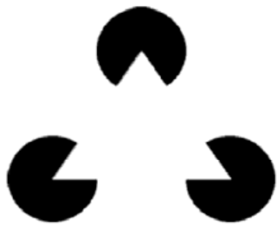
  - c = 3x1 matrix: 3 rows and 1 column
  - r = 1x3 matrix: 1 row and 3 columns

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



  - S = 2 x 2 matrix
  - R = 2 x 3 matrix
  - Pacman = 321 x 399 matrix

Representing an image as a matrix



```
>> 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
```

$$
\begin{matrix} Y \\ X \\ v \end{matrix}
\begin{bmatrix}
1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & 10 \\
1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & 10 \\
1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1
\end{bmatrix}
$$

$$
\begin{bmatrix} 1 & 1 & . & 1 & 1 & . & 0 & 0 & 0 & . & . & 1 \end{bmatrix}
$$

Values only; X and Y are implicit

- 3 pacmen
- A 321x399 matrix
  - Row and Column = position
- A 3x128079 matrix
  - Triples of x,y and value
- A 1x128079 vector
  - “Unraveling” the matrix

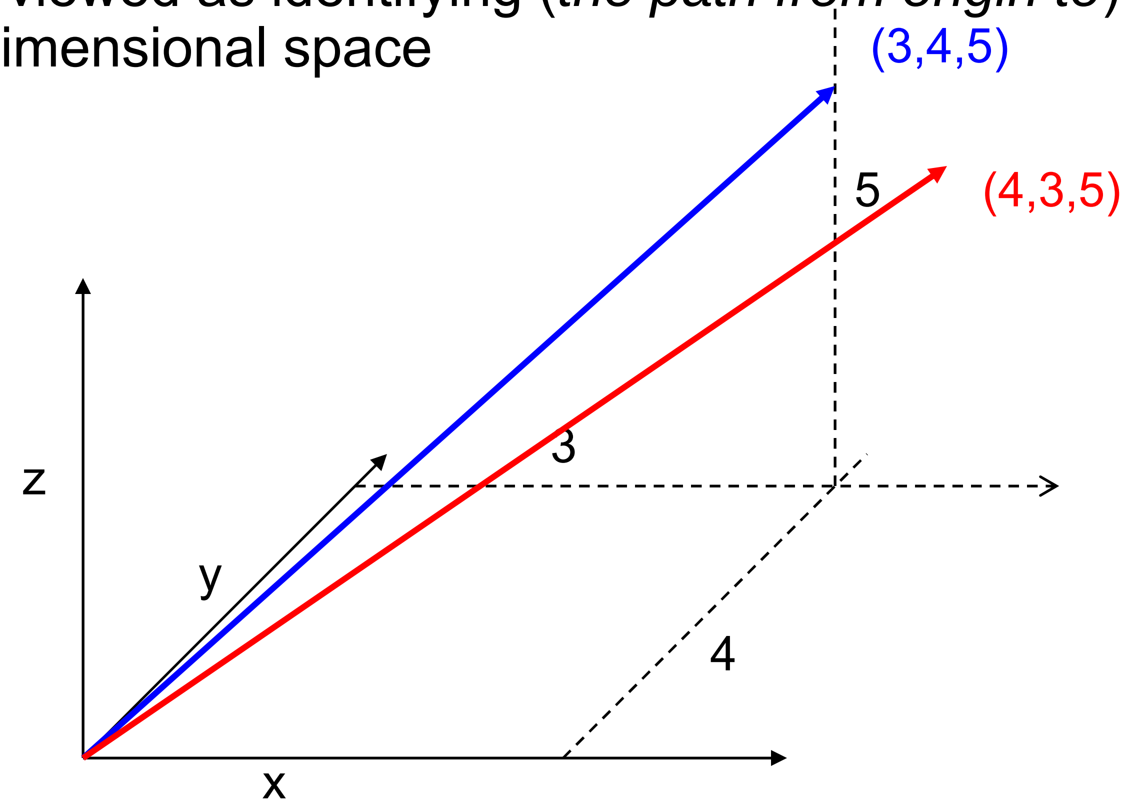
- Note: All of these can be recast as the matrix that forms the image
  - Representations 2 and 4 are equivalent
    - The position is not represented

Example of a vector

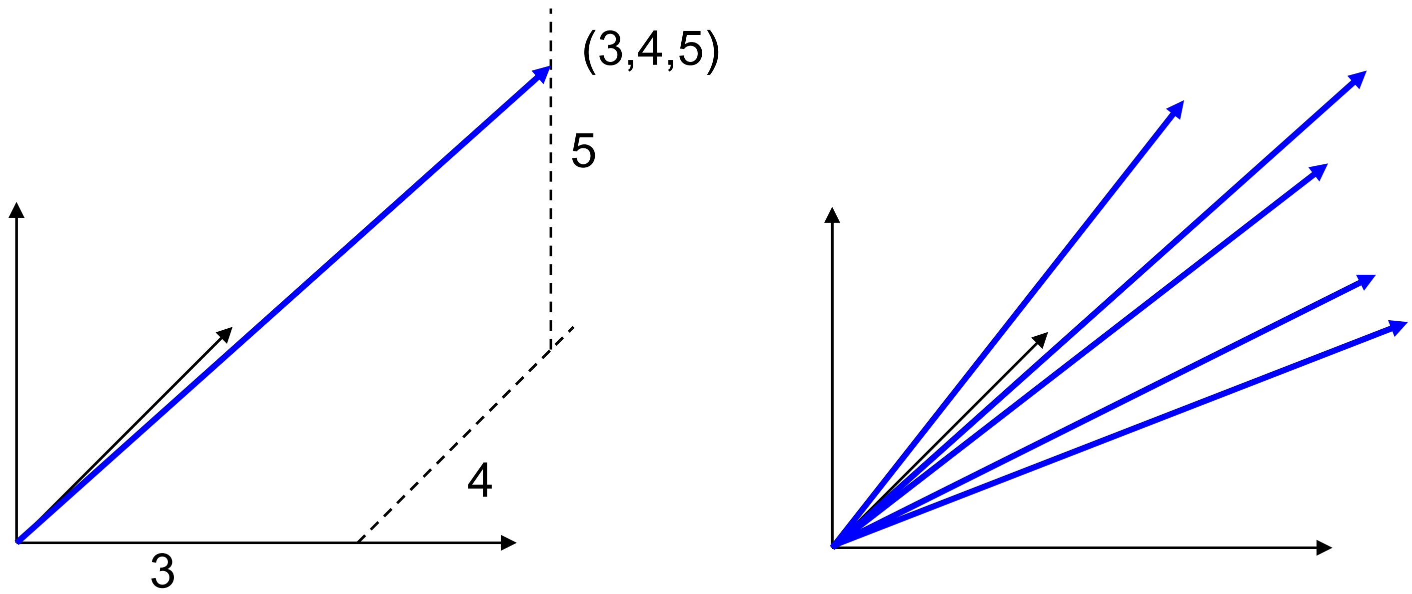
- Vectors usually hold sets of numerical attributes
  - X, Y, value
    - [1, 2, 0]
  - Earnings, losses, suicides
    - [$0 $1.000.000 3]
  - Etc …
- Consider a “relative Manhattan” vector
  - Provides a relative position by giving a number of avenues and streets to cross, e.g. [3av 33st]

Vectors

- Ordered collection of numbers
  - Examples: [3 4 5], [a b c d], ..
  - [3 4 5] != [4 3 5] → **Order is important**
- Typically viewed as identifying (*the path from origin to*) a location in an N-dimensional space



Vectors vs. Matrices



- A vector is a geometric notation for how to get from (0,0) to some location in the space
- A matrix is simply a collection of destinations!
  - Properties of matrices are *average* properties of the traveller's path to these destinations

Basic arithmetic operations

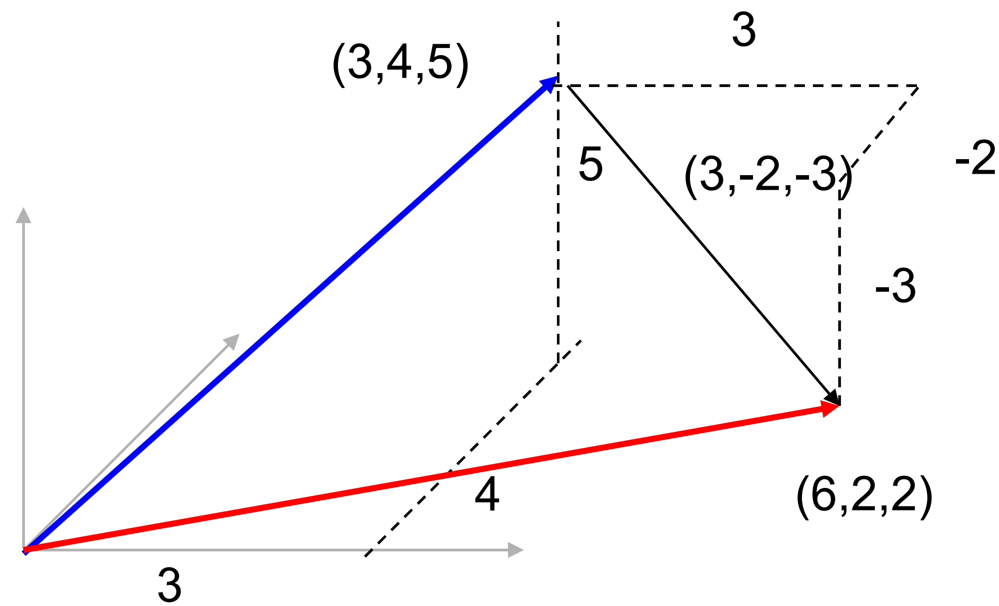
- Addition and subtraction
  - Element-wise operations

$$\mathbf{a}+\mathbf{b}=\begin{bmatrix}a_1\\a_2\\a_3\end{bmatrix}+\begin{bmatrix}b_1\\b_2\\b_3\end{bmatrix}=\begin{bmatrix}a_1+b_1\\a_2+b_2\\a_3+b_3\end{bmatrix}\qquad \mathbf{a}-\mathbf{b}=\begin{bmatrix}a_1\\a_2\\a_3\end{bmatrix}-\begin{bmatrix}b_1\\b_2\\b_3\end{bmatrix}=\begin{bmatrix}a_1-b_1\\a_2-b_2\\a_3-b_3\end{bmatrix}$$

$$\mathbf{A}+\mathbf{B}=\begin{bmatrix}a_{11}&a_{12}\\a_{21}&a_{22}\end{bmatrix}+\begin{bmatrix}b_{11}&b_{12}\\b_{21}&b_{22}\end{bmatrix}=\begin{bmatrix}a_{11}+b_{11}&a_{12}+b_{12}\\a_{21}+b_{21}&a_{22}+b_{22}\end{bmatrix}$$

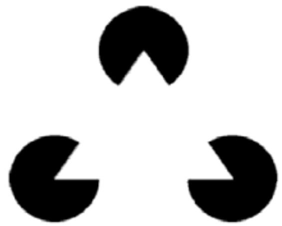
- MATLAB syntax: `a+b` and `a-b`

Vector Operations



- Operations tell us how to get from ({0}) to the result of the vector operations
  - (3,4,5) + (3,-2,-3) = (6,2,2)

Operations example



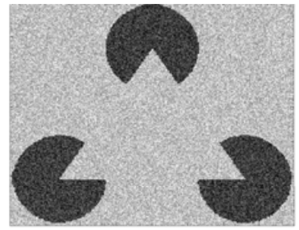
```
>> X(1:32:end,1:40:end)

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
```

$$\begin{bmatrix} 1 & 1 & . & 1 & 1 & . & 0 & 0 & 0 & . & . & 1 \end{bmatrix}$$

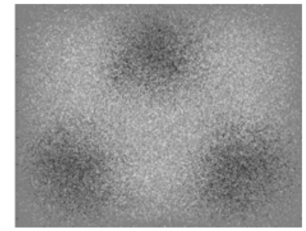
$$\begin{bmatrix} 1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & 10 \\ 1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & 10 \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$



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

+

0.6265	0.4839	0.7874	0.1749	0.3661	0.7716	0.9012	0.5111	0.1518	0.3528
0.2116	0.3506	0.4380	0.0707	0.2653	0.1263	0.6099	0.7610	0.8772	0.4465
0.1194	0.8242	0.2729	0.8895	0.0681	0.8501	0.4507	0.1967	0.6585	0.3526
0.9257	0.3736	0.5879	0.1068	0.9746	0.6336	0.1589	0.8425	0.0656	0.3100
0.6203	0.5834	0.5060	0.3234	0.5002	0.1915	0.2964	0.0938	0.4636	0.5114
0.5782	0.9409	0.5144	0.3392	0.4970	0.6307	0.4717	0.9053	0.4850	0.2467
0.8887	0.4086	0.7580	0.9547	0.7566	0.9898	0.5251	0.6518	0.8996	0.7946
0.0616	0.3867	0.4978	0.5169	0.0529	0.8565	0.5613	0.3270	0.8976	0.6088
0.6095	0.1383	0.6277	0.5475	0.6145	0.5146	0.1139	0.1981	0.3401	0.4395
0.3806	0.2752	0.5621	0.9369	0.6252	0.0422	0.6518	0.8614	0.5366	0.9572
0.1412	0.2130	0.6296	0.9251	0.6635	0.7859	0.2164	0.5335	0.1640	0.3909



$$\begin{bmatrix} 1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & 10 \\ 1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & 10 \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$

+

Random(3,columns(M))

- Adding random values to different representations of the image

Vector norm

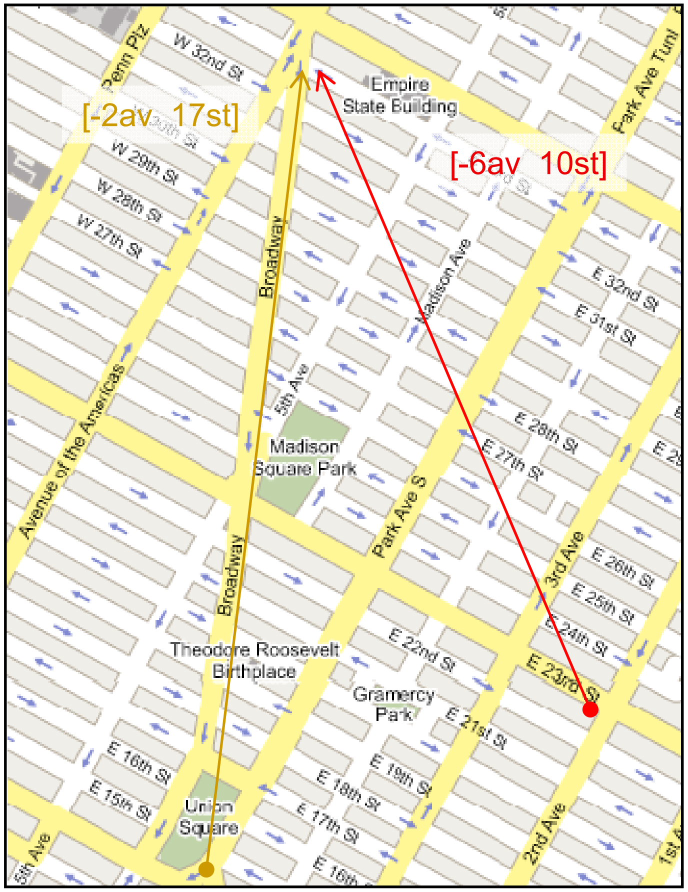
- Measure of how big a vector is:
  - Notated as $\|\mathbf{x}\|$

$$\left\|\begin{bmatrix} a & b & \ldots \end{bmatrix}\right\| = \sqrt{a^2 + b^2 + \ldots^2}$$

- In Manhattan vectors a measure of distance

$$\left\|\begin{bmatrix} -2 & 17 \end{bmatrix}\right\| = 17.11$$

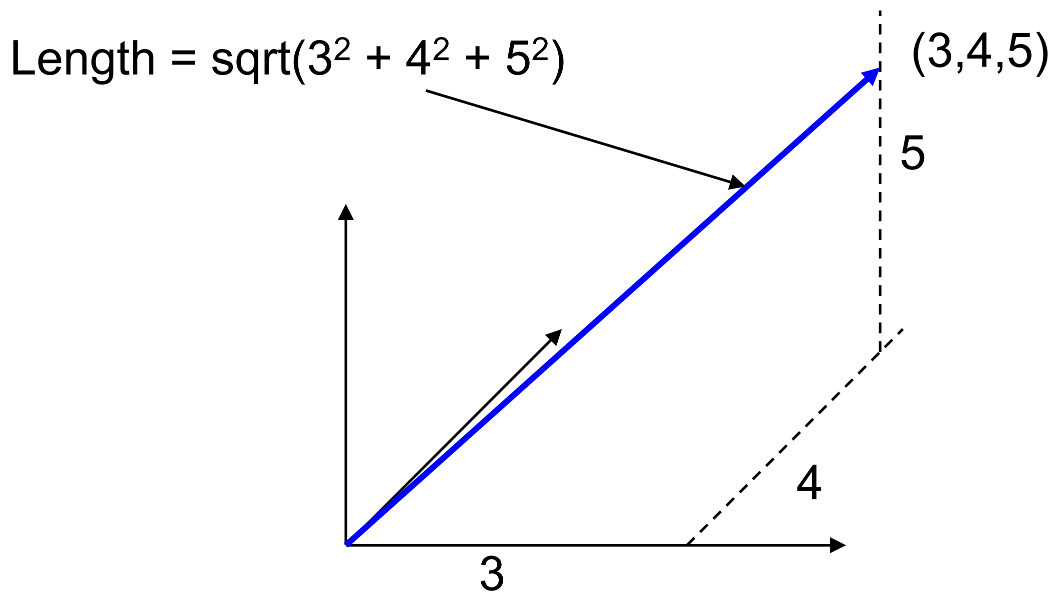
$$\left\|\begin{bmatrix} -6 & 10 \end{bmatrix}\right\| = 11.66$$



- MATLAB syntax:

```
norm(x)
```

Vector Norm



- Geometrically the shortest distance to travel from the origin to the destination
  - As the crow flies
  - Assuming Euclidean Geometry

Transposition

- A transposed row vector becomes a column
  (and vice versa)

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

- A transposed matrix gets all its row (or column) vectors transposed in order

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

$\mathbf{M} =$ , $\mathbf{M}^T =$ 

- MATLAB syntax: `a'`

Vector multiplication

- Multiplication is not element-wise!
- Dot product, or inner product
  - Vectors must have the same number of elements
  - Row vector times column vector = scalar

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

- Cross product, outer product or vector direct product
  - Column vector times row vector = matrix

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

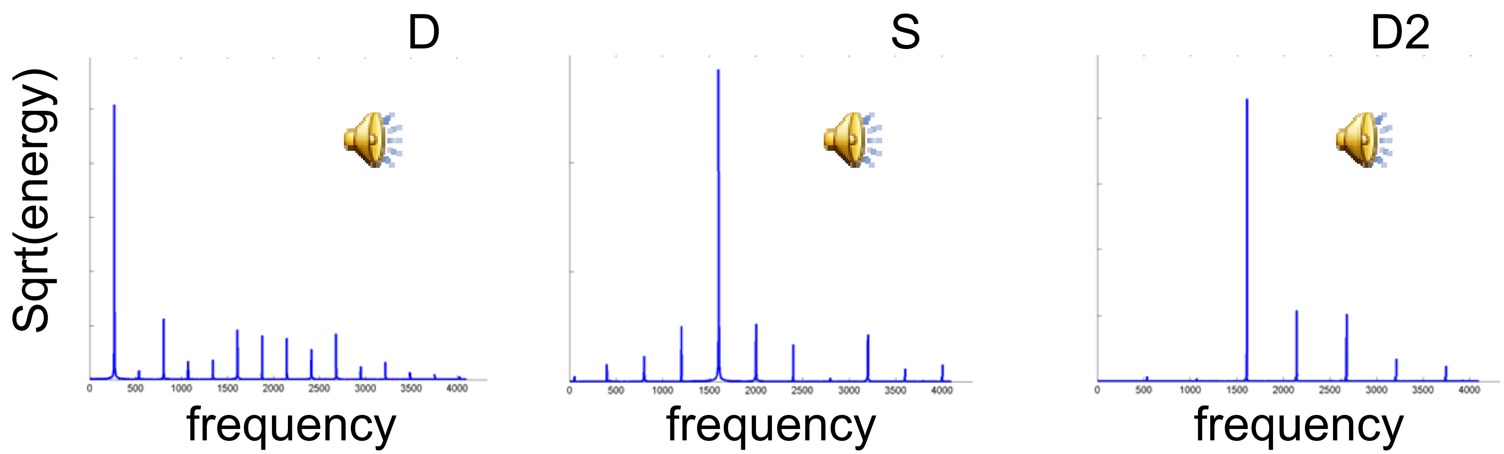
- MATLAB syntax: `a*b`

Vector ***dot product*** in Manhattan

- Multiplying the “yard” vectors
  - Instead of avenue/street we’ll use yards
  - a = [200 1600], 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?
  - The answer comes back as a unit of the first vector so we divide by its length

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

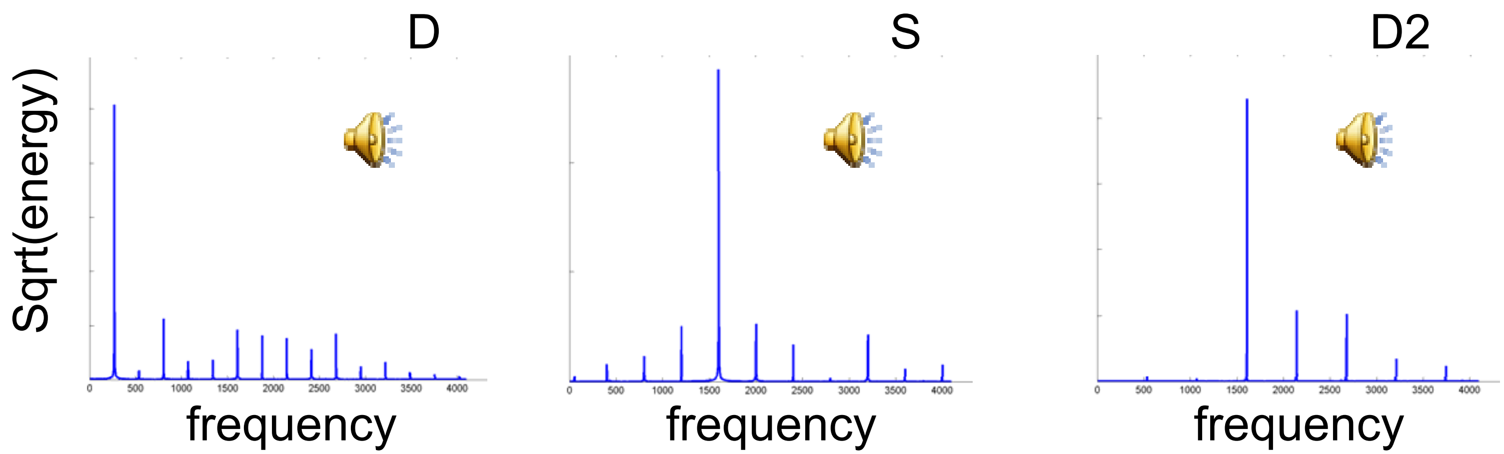
Vector dot product



[11 9 . 0 54 1 . . . 1]   [3 . 24 . . 16 . 14 . 1]   [0 . 0 . 3 0 . 13 . 0]

- Vectors are spectra
  - Energy at a discrete set of frequencies
  - Actually 1x4096
  - 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

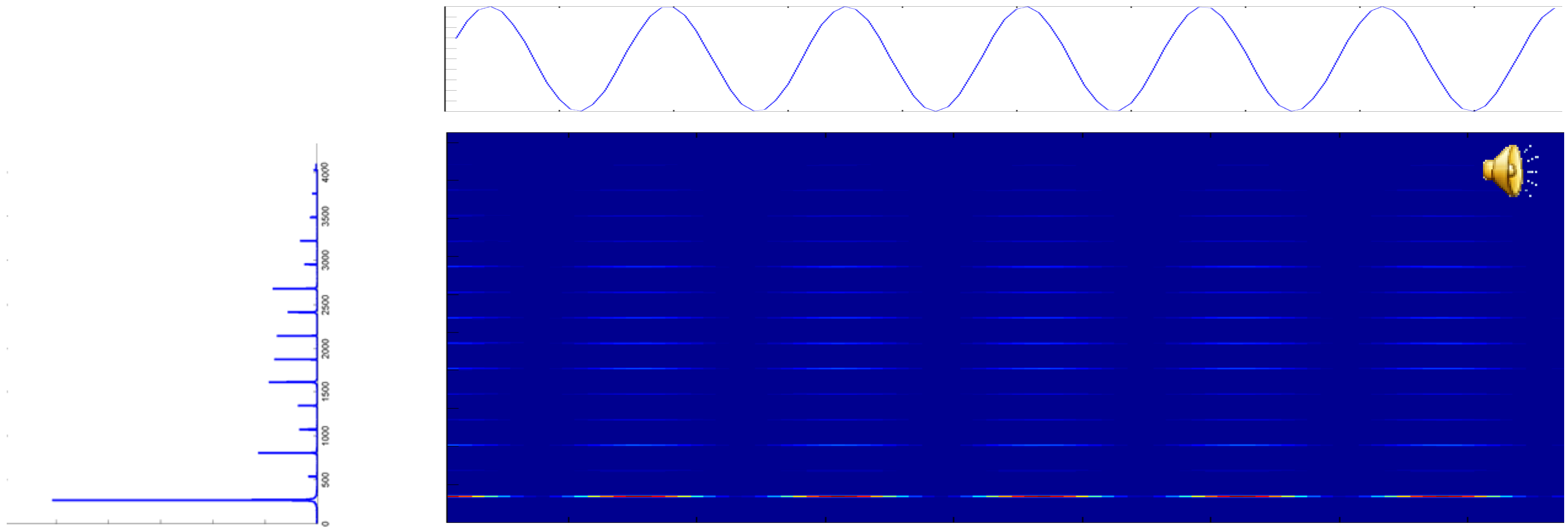
Vector dot product



[11 9 . 0 54 1 . . . 1]   [3 . 24 . . 16 . 14 . 1]   [0 . 0 . 3 0 . 13 . 0]

- How much of D is also in S
  - How much can you fake a D by playing an S
  - D.S / |D||S| = 0.1
  - Not very much
- How much of D is in D2?
  - D.D2 / |D| /|D2| = 0.5
  - Not bad, you can fake it
- To do this, D, S, and D2 *must be the same size*

Vector cross product



- The column vector is the spectrum
- The row vector is an amplitude modulation
- The crossproduct 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

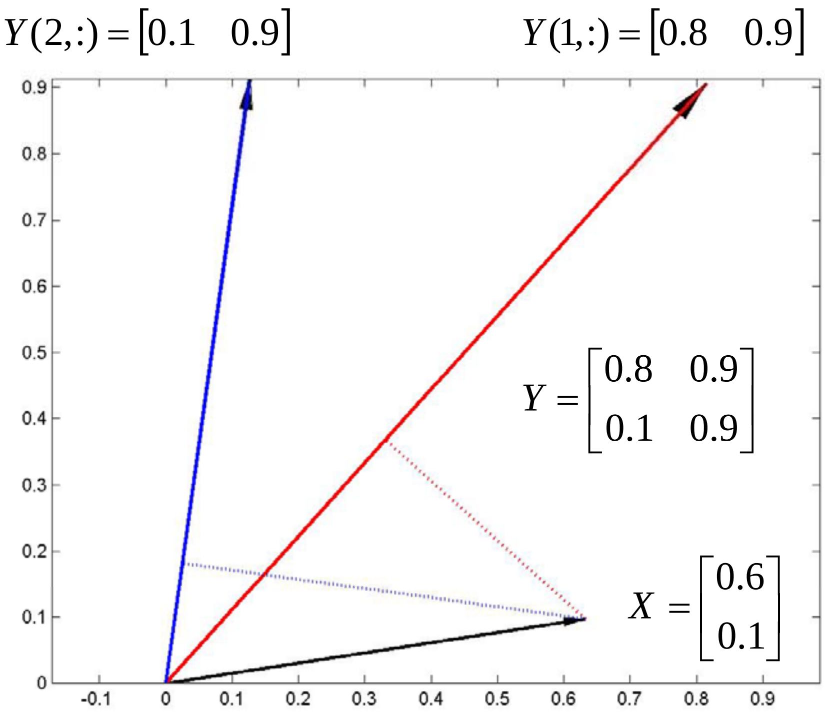
Matrix multiplication

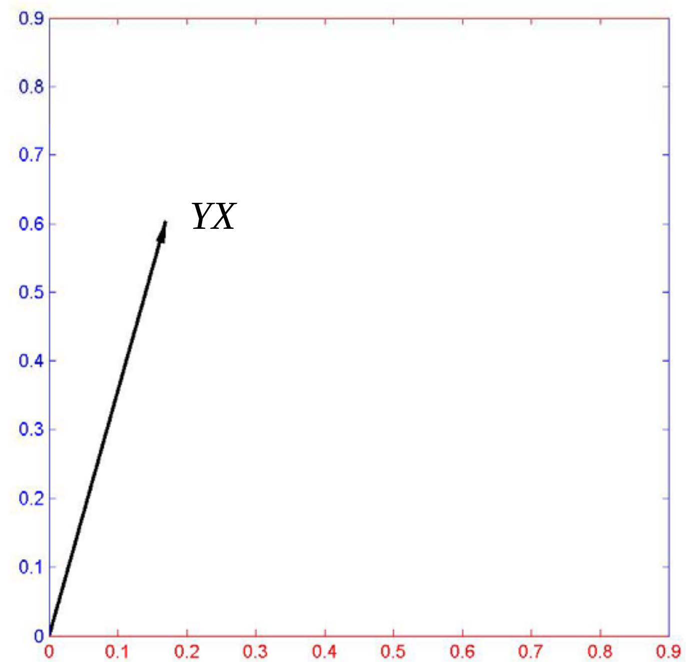
- Generalization of vector multiplication
  - Dot product of each vector pair

$$\mathbf{A} \cdot \mathbf{B} = \begin{bmatrix} \leftarrow & \mathbf{a}_1 & \rightarrow \\ \leftarrow & \mathbf{a}_2 & \rightarrow \end{bmatrix} \cdot \begin{bmatrix} \uparrow & \uparrow \\ \mathbf{b}_1 & \mathbf{b}_2 \\ \downarrow & \downarrow \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \cdot \mathbf{b}_1 & \mathbf{a}_1 \cdot \mathbf{b}_2 \\ \mathbf{a}_2 \cdot \mathbf{b}_1 & \mathbf{a}_2 \cdot \mathbf{b}_2 \end{bmatrix}$$

  - Dimensions must match!!
    - Columns of first matrix = rows of second
    - Result inherits the number of rows from the first matrix and the number of columns from the second matrix
- MATLAB syntax: `a*b`

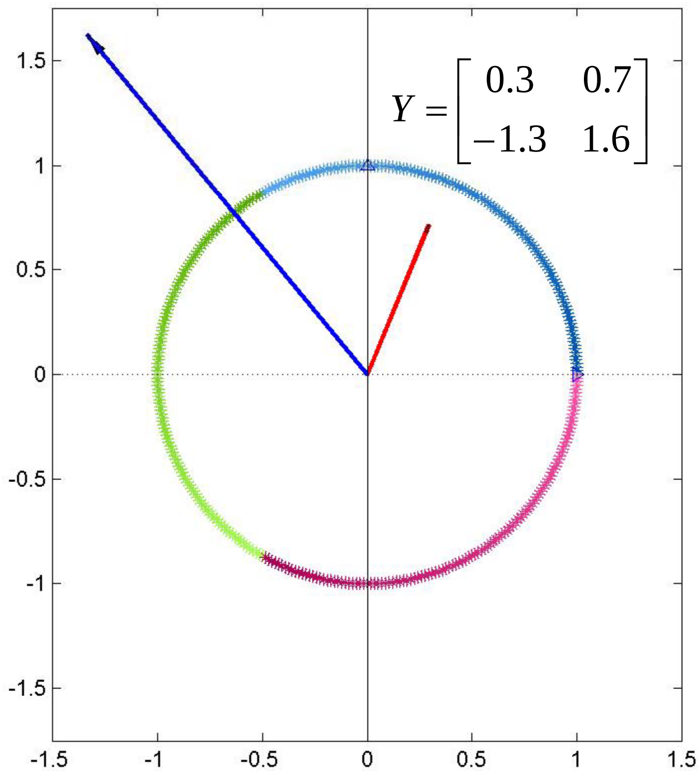
Multiplying a Vector by a Matrix

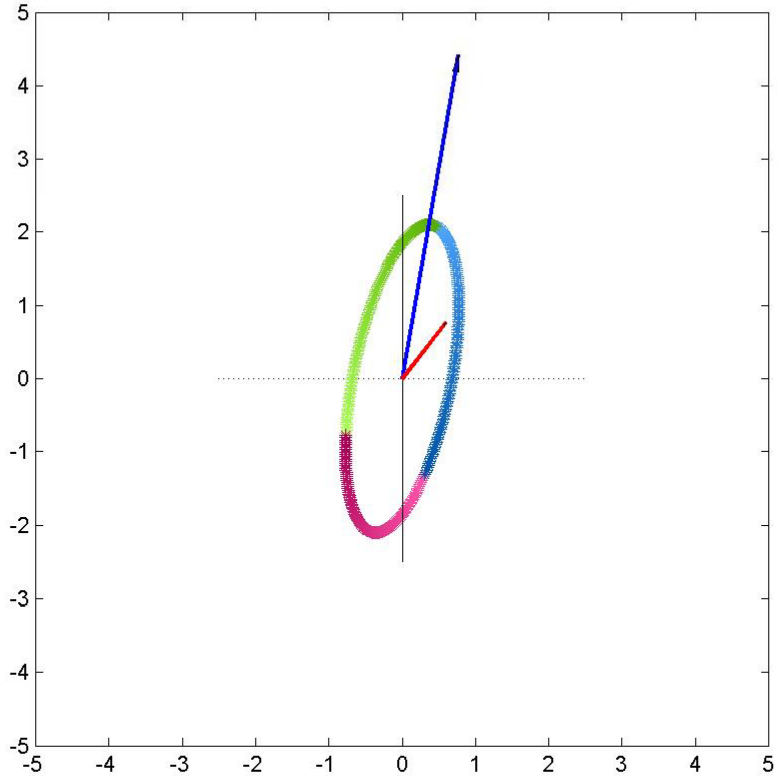




- Multiplication of a vector X by a matrix Y expresses the vector X in terms of projections of X on the row vectors of the matrix Y
  - It scales and rotates the vector
  - Alternately viewed, it scales and rotates the space – the underlying plane

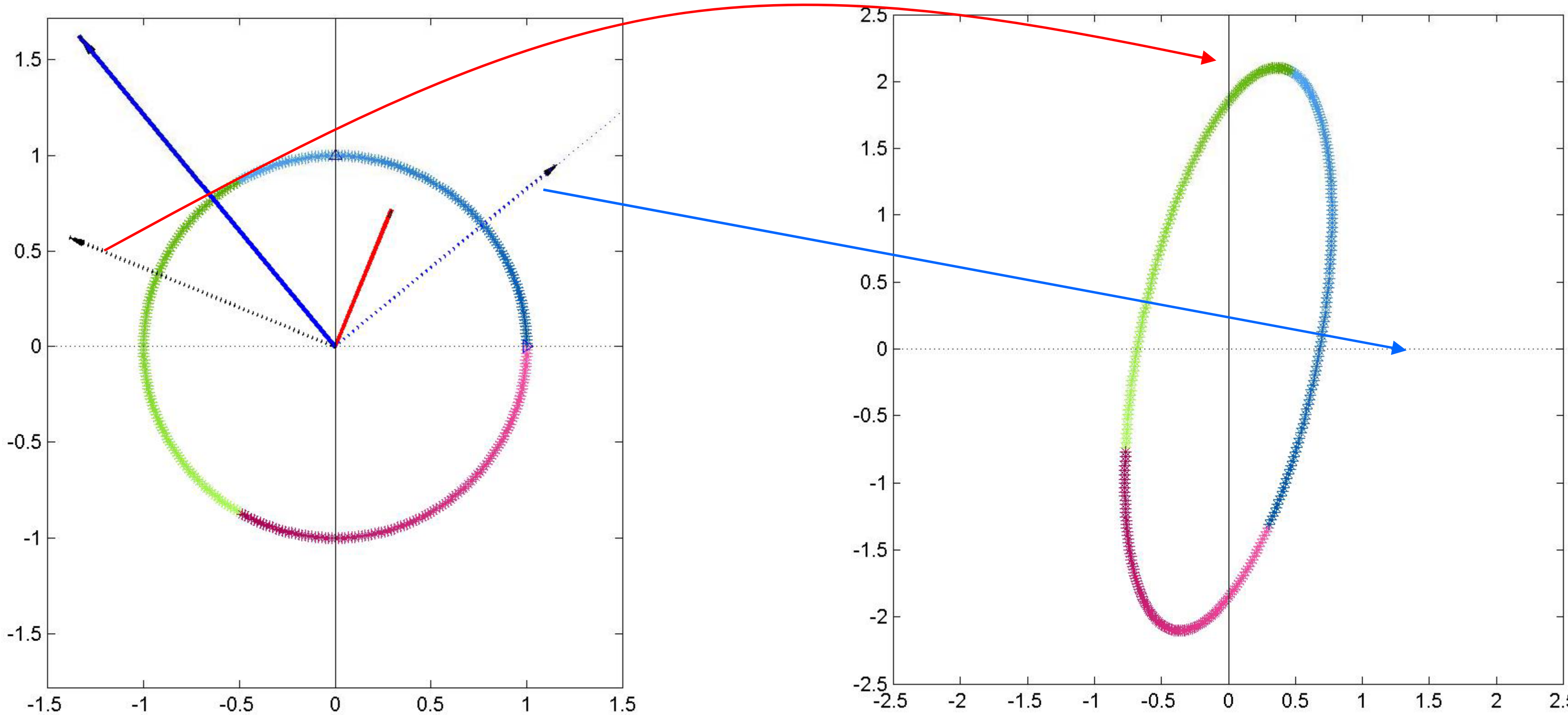
Matrix Multiplication





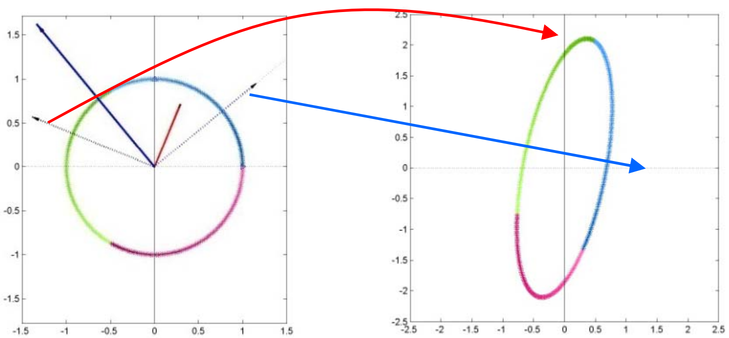
- The matrix rotates and scales the space
  - Including its own vectors

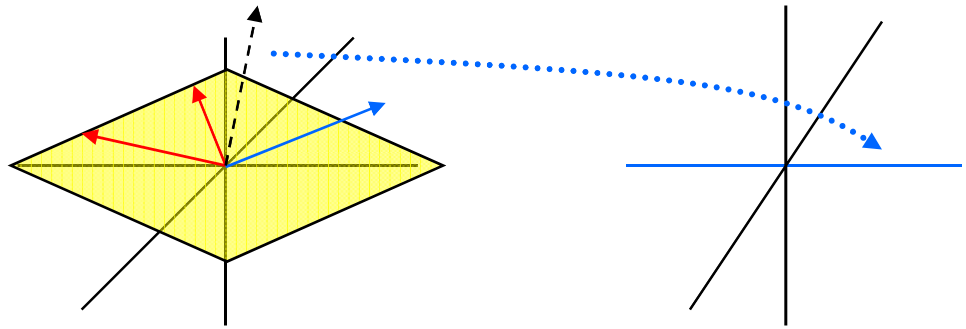
Matrix Multiplication



- 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 is projection





- 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

Matrix Multiplication: Column space

$$\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = x\begin{bmatrix} a \\ d \end{bmatrix} + y\begin{bmatrix} b \\ e \end{bmatrix} + z\begin{bmatrix} c \\ f \end{bmatrix}$$

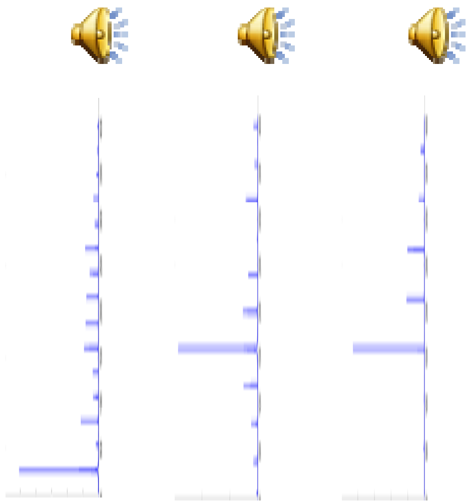
- So much for spaces .. what does multiplying a matrix by a vector really do?
- It *mixes* the column vectors of the matrix using the numbers in the vector
- The *column space* of the Matrix is the complete set of all vectors that can be formed by mixing its columns

Matrix Multiplication: Row space

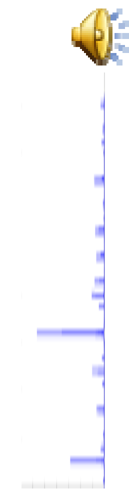
$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} = x\begin{bmatrix} a & b & c \end{bmatrix} + y\begin{bmatrix} d & e & f \end{bmatrix}$$

- Left multiplication mixes the *row vectors* of the matrix.
- The *row space* of the Matrix is the complete set of all vectors that can be formed by mixing its rows

Matrix multiplication: Mixing vectors

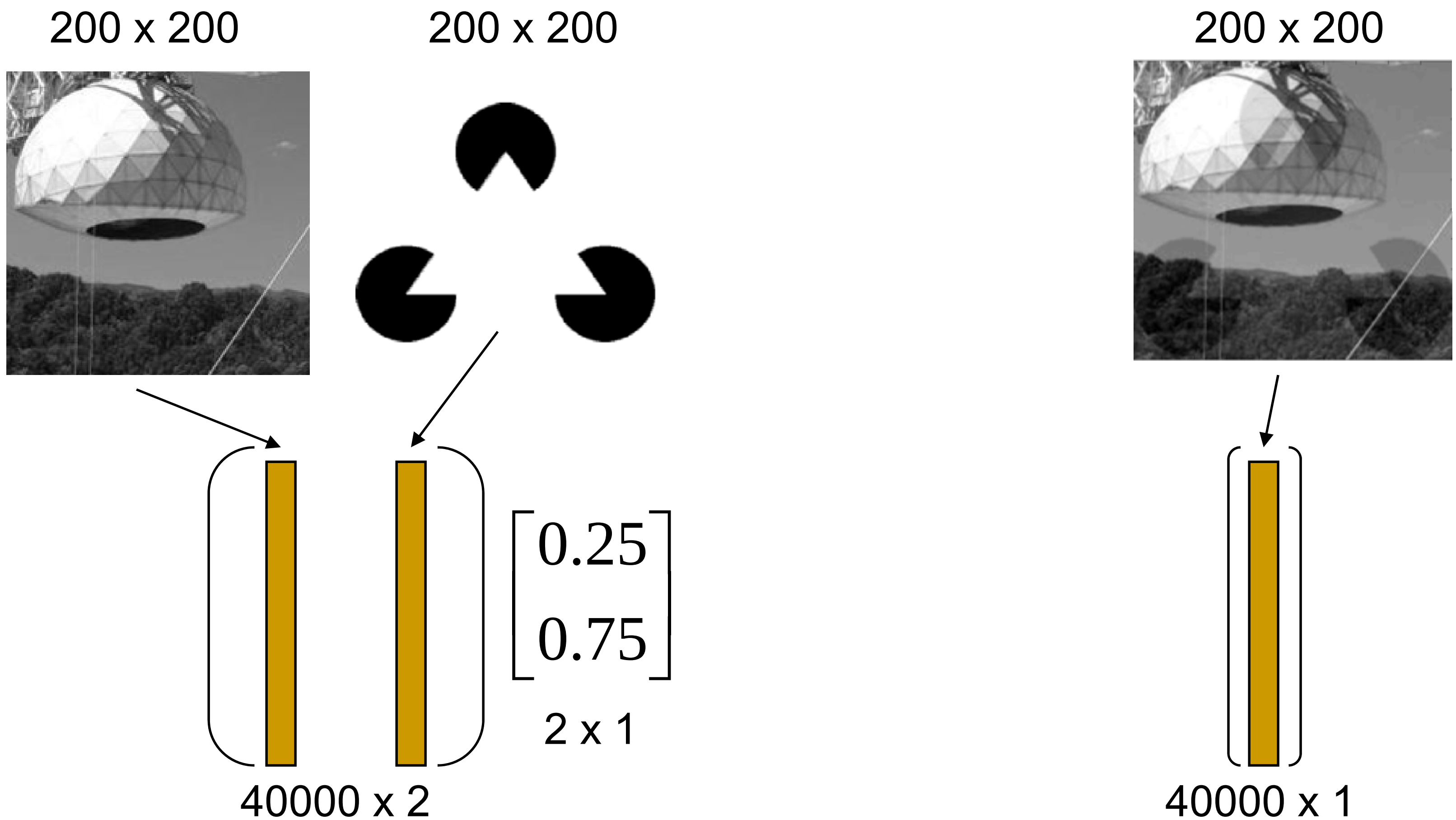


$$\overset{X}{\begin{bmatrix} 1 & 3 & 0 \\ . & . & 0 \\ 9 & 24 & . \\ . & . & 1 \end{bmatrix}} \overset{Y}{\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}} = \begin{bmatrix} 7 \\ . \\ . \\ 2 \end{bmatrix}$$



- 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

Matrix multiplication: another view

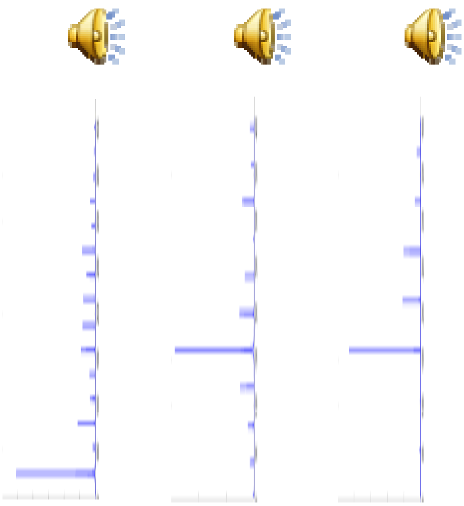
$$\mathbf{A}\cdot\mathbf{B} = \begin{bmatrix} a_{11} & . & . & a_{1N} \\ a_{21} & . & . & a_{2N} \\ . & . & . & . \\ a_{M1} & . & . & a_{MN} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & . & b_{NK} \\ . & . & . \\ b_{N1} & . & b_{NK} \end{bmatrix} = \begin{bmatrix} \sum_k a_{1k}b_{k1} & . & \sum_k a_{1k}b_{kK} \\ . & . & . \\ \sum_k a_{Mk}b_{k1} & . & \sum_k a_{Mk}b_{kK} \end{bmatrix}$$

- What does this mean?

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

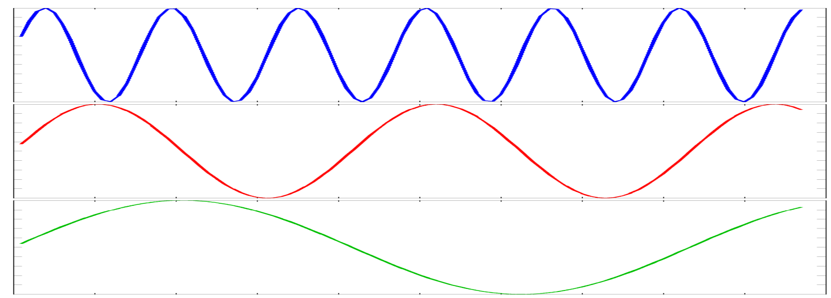
- The outer product of the first column of A and the first row of B + outer product of the second column of A and the second row of B + ….

Why is that useful?



$$\begin{bmatrix} 1 & 3 & 0 \\ . & . & 0 \\ 9 & 24 & . \\ . & . & 1 \end{bmatrix}$$

X

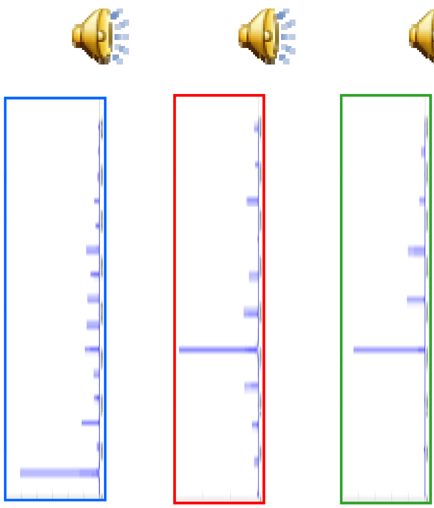


$$\begin{bmatrix} 0 & 0.5 & 0.75 & 1 & 0.75 & 0.5 & 0 & . & . & . & . & . \\ 1 & 0.9 & 0.7 & 0.5 & 0 & 0.5 & . & . & . & . & . & . \\ 0.5 & 0.6 & 0.7 & 0.8 & 0.9 & 0.95 & 1 & . & . & . & . & . \end{bmatrix}$$

Y

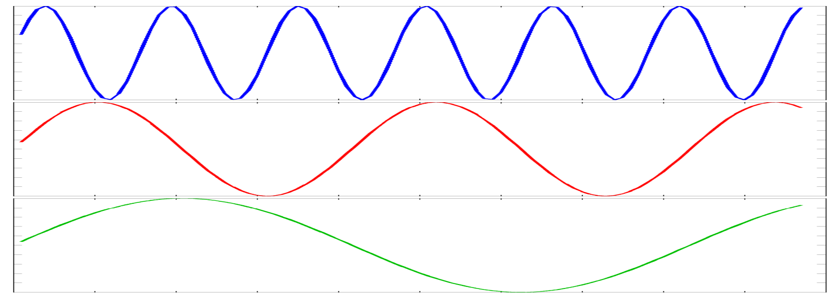
- Sounds: Three notes modulated independently

Matrix multiplication: Mixing modulated spectra



$$\begin{bmatrix} 1 & 3 & 0 \\ . & . & 0 \\ 9 & 24 & . \\ . & . & 1 \end{bmatrix}$$

X

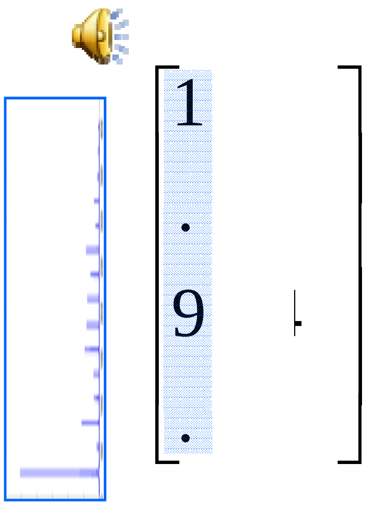


$$\begin{bmatrix} 0 & 0.5 & 0.75 & 1 & 0.75 & 0.5 & 0 & . & . & . & . & . \\ 1 & 0.9 & 0.7 & 0.5 & 0 & 0.5 & . & . & . & . & . & . \\ 0.5 & 0.6 & 0.7 & 0.8 & 0.9 & 0.95 & 1 & . & . & . & . & . \end{bmatrix}$$

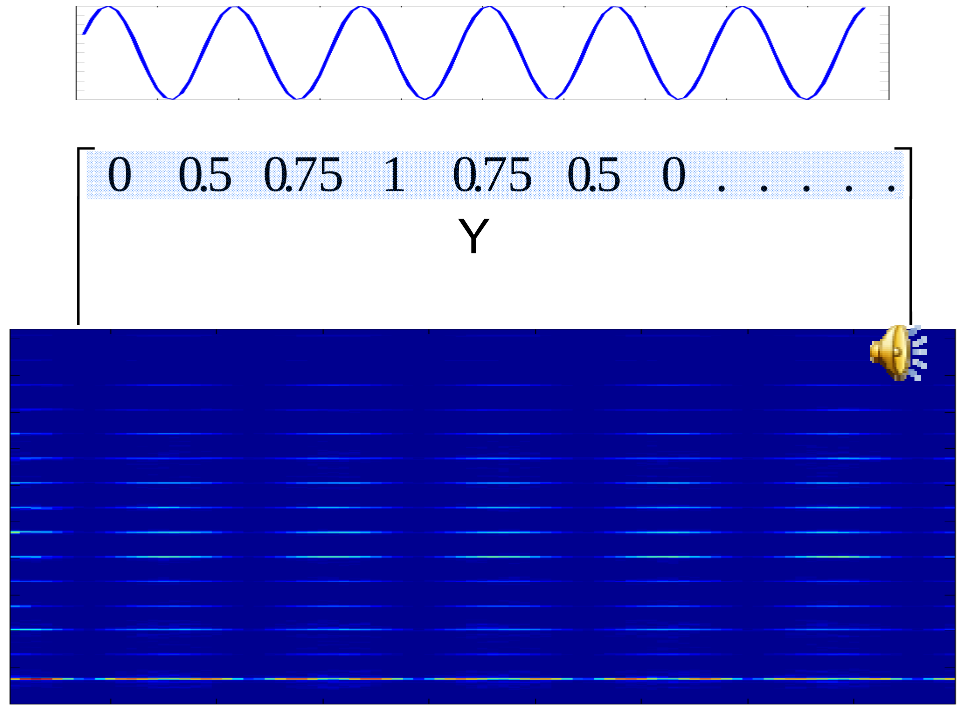
Y

- Sounds: Three notes modulated independently

Matrix multiplication: Mixing modulated spectra

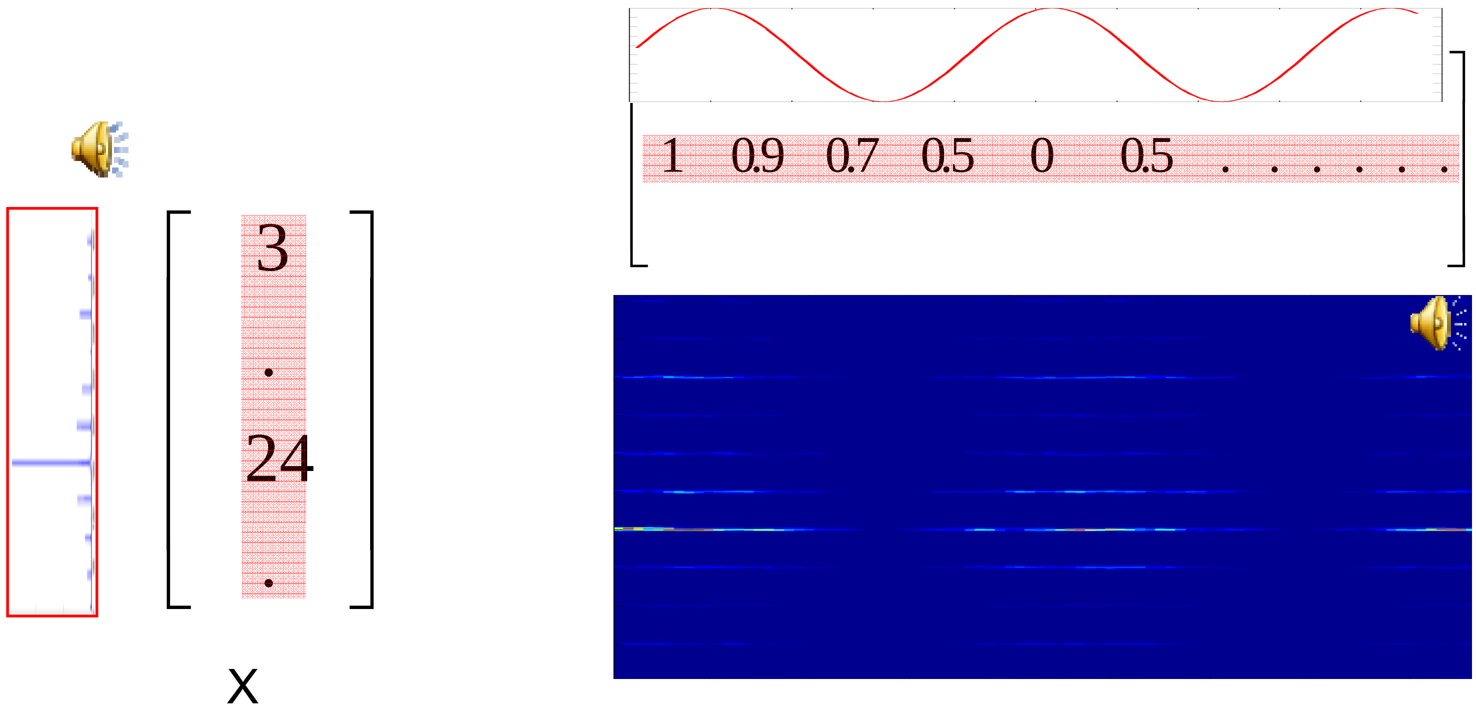


X



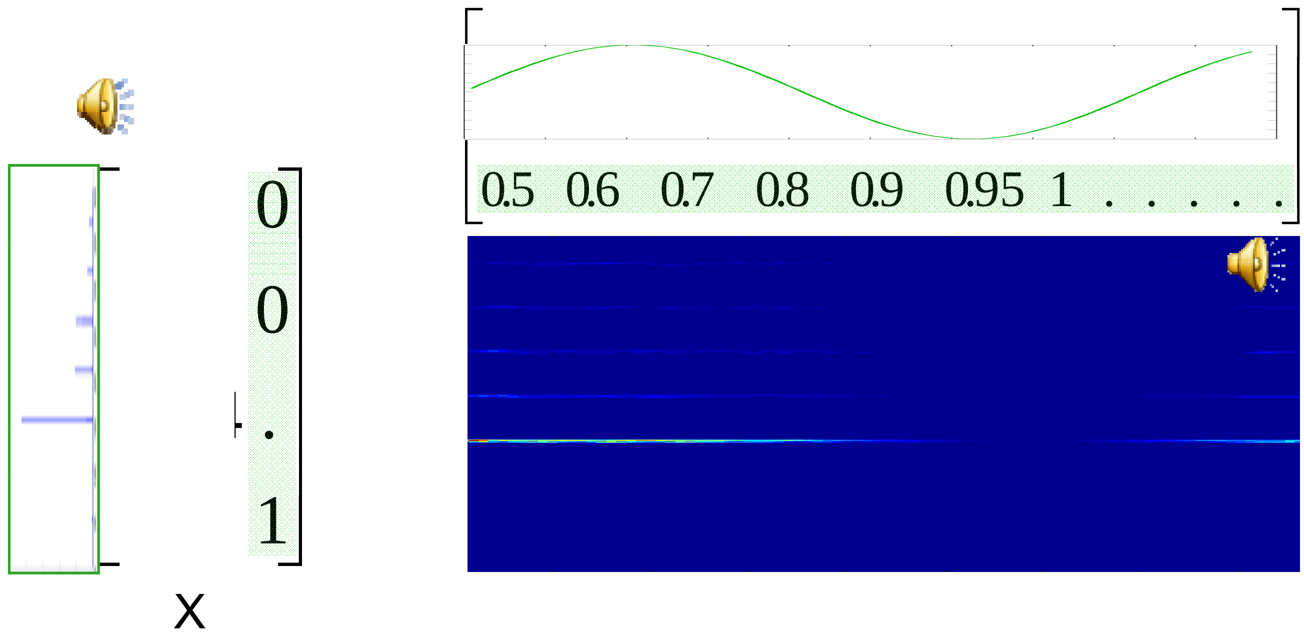
- Sounds: Three notes modulated independently

Matrix multiplication: Mixing modulated spectra



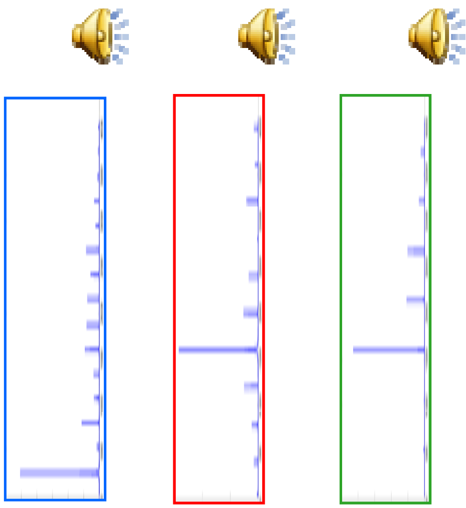
- Sounds: Three notes modulated independently

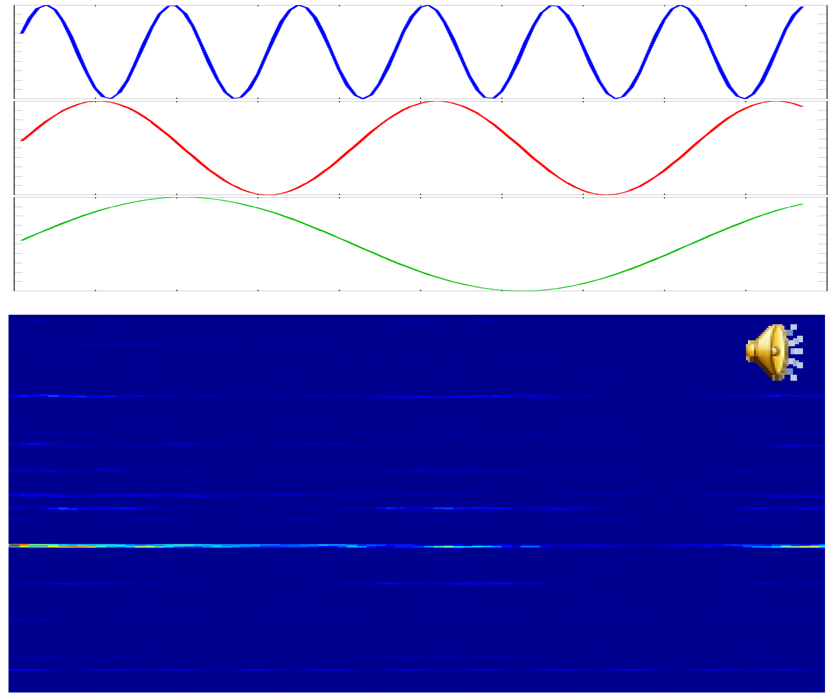
Matrix multiplication: Mixing modulated spectra



- Sounds: Three notes modulated independently

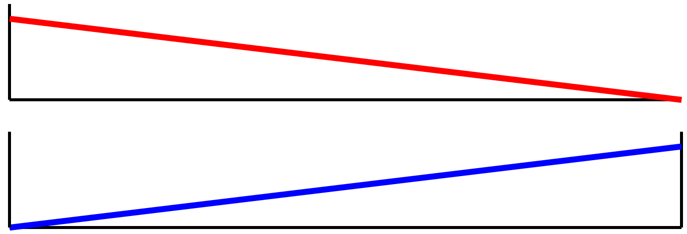
Matrix multiplication: Mixing modulated spectra



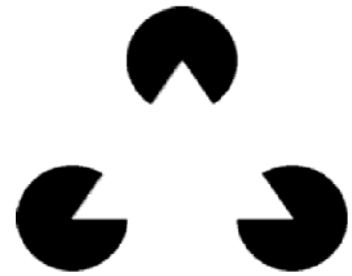


- Sounds: Three notes modulated independently

Matrix multiplication: Image transition

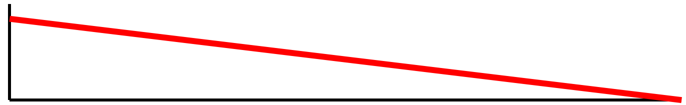


$$\begin{bmatrix} i_1 & j_1 \\ i_2 & j_2 \\ . & . \\ . & . \\ . & . \end{bmatrix} \begin{bmatrix} 1 & .9 & .8 & .7 & .6 & .5 & .4 & .3 & .2 & .1 & 0 \\ 0 & .1 & .2 & .3 & .4 & .5 & .6 & .7 & .8 & .9 & 1 \end{bmatrix}$$



- Image1 fades out linearly
- Image 2 fades in linearly

Matrix multiplication: Image transition

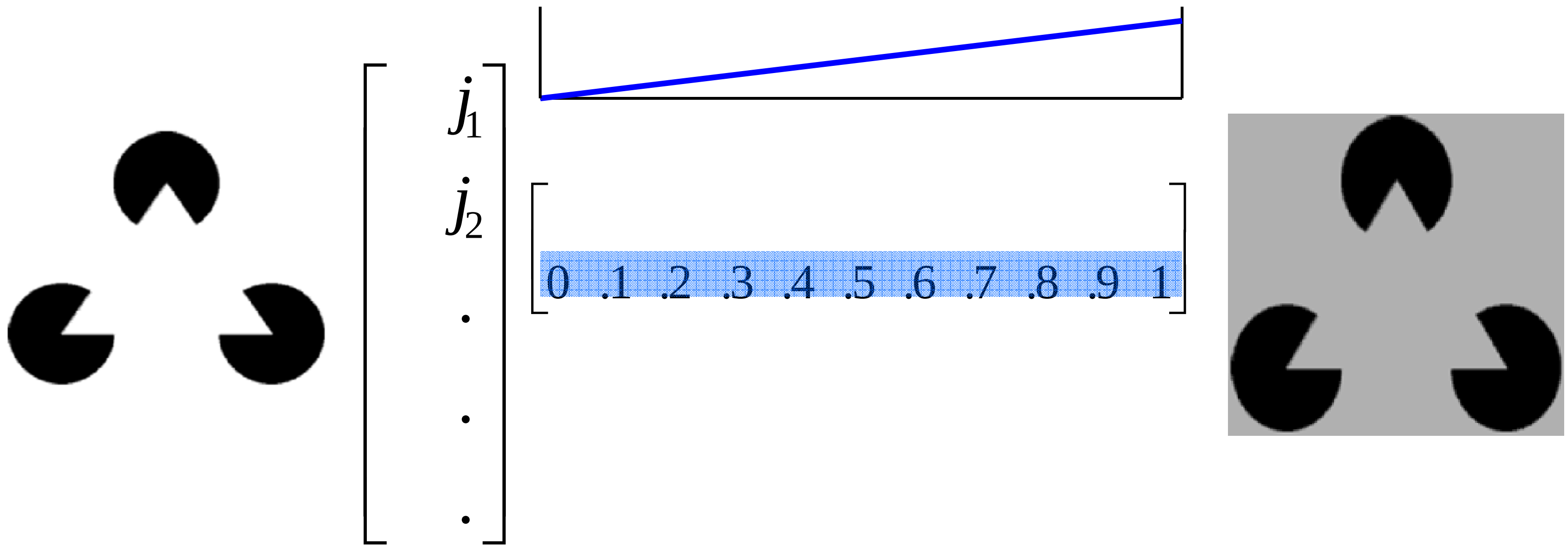


$$\begin{bmatrix} i_1 \\ i_2 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \begin{bmatrix} 1 & .9 & .8 & .7 & .6 & .5 & .4 & .3 & .2 & .1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} i_1 & 0.9i_1 & 0.8i_1 & . & . & . & . & . & . & 0 \\ i_2 & 0.9i_2 & 0.8i_2 & . & . & . & . & . & . & 0 \\ . & . & . & . & . & . & . & . & . & 0 \\ . & . & . & . & . & . & . & . & . & 0. \\ i_N & 0.9i_N & 0.8i_N & . & . & . & . & . & . & 0 \end{bmatrix}$$

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

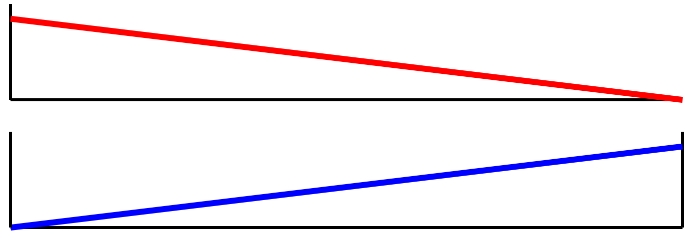
Matrix multiplication: Image transition



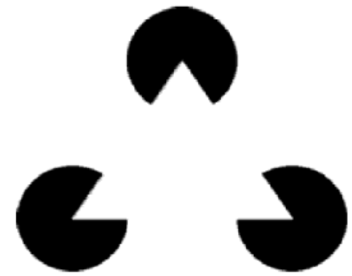
- Image 2 fades in linearly

Matrix multiplication: Image transition

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



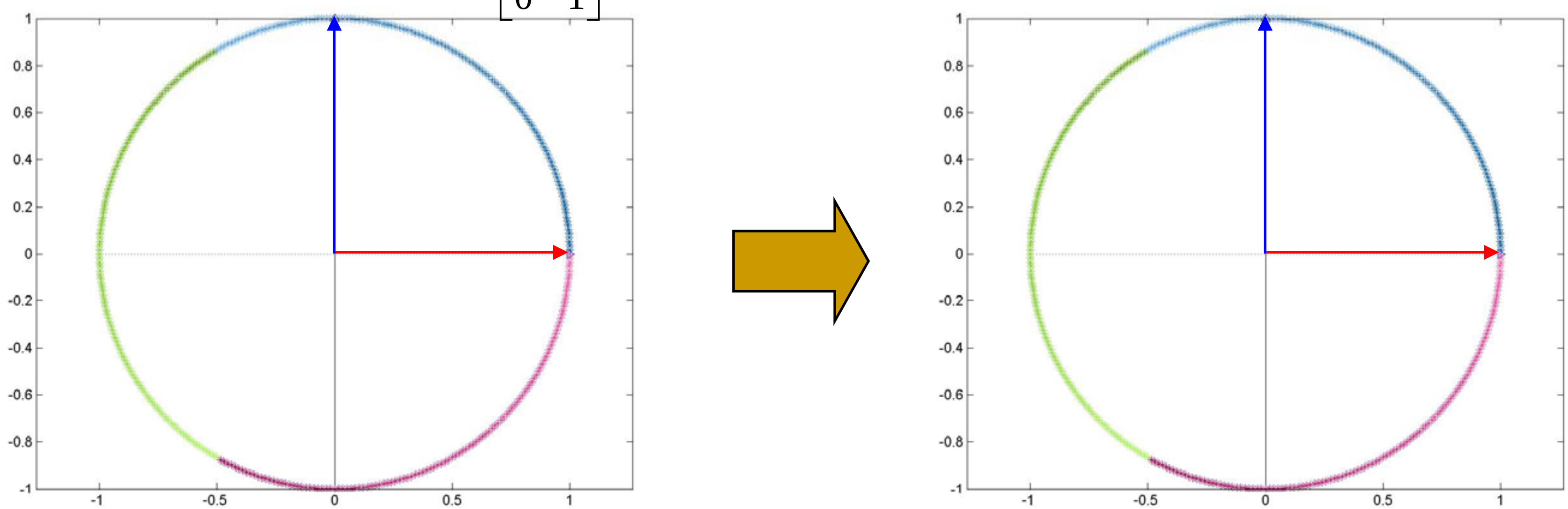
$$\begin{bmatrix} 1 & .9 & .8 & .7 & .6 & .5 & .4 & .3 & .2 & .1 & 0 \\ 0 & .1 & .2 & .3 & .4 & .5 & .6 & .7 & .8 & .9 & 1 \end{bmatrix}$$



- Image1 fades out linearly
- Image 2 fades in linearly

The Identity Matrix

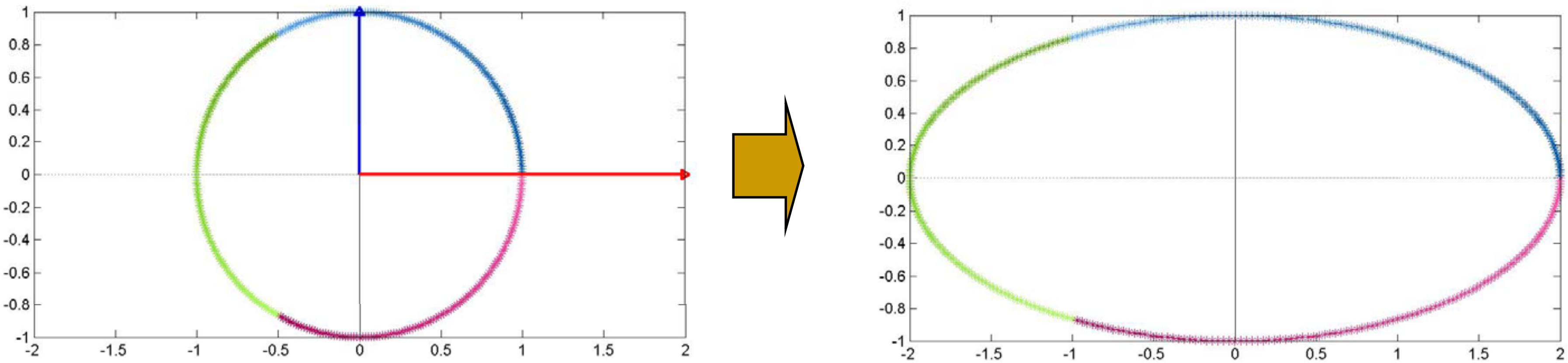
$$Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



- 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

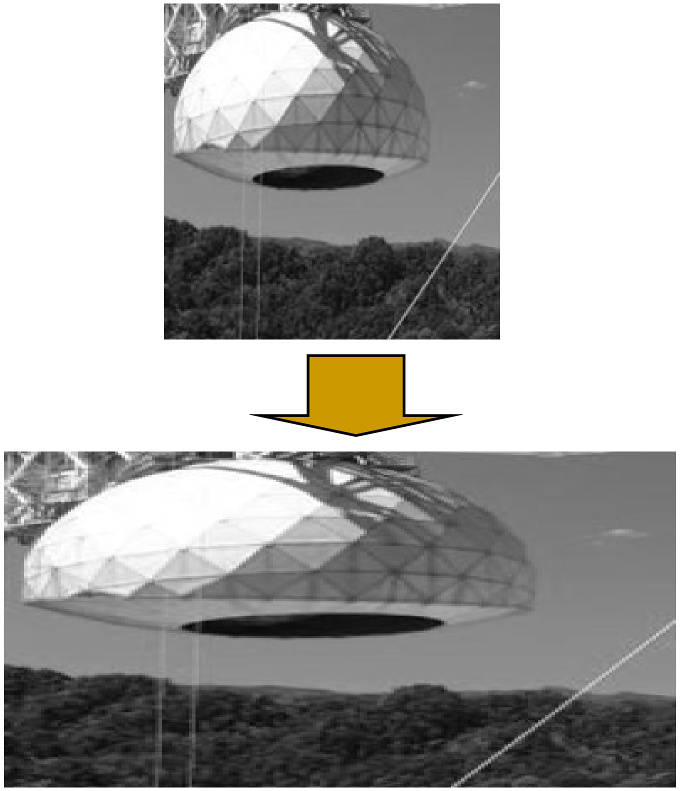
Diagonal Matrix

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



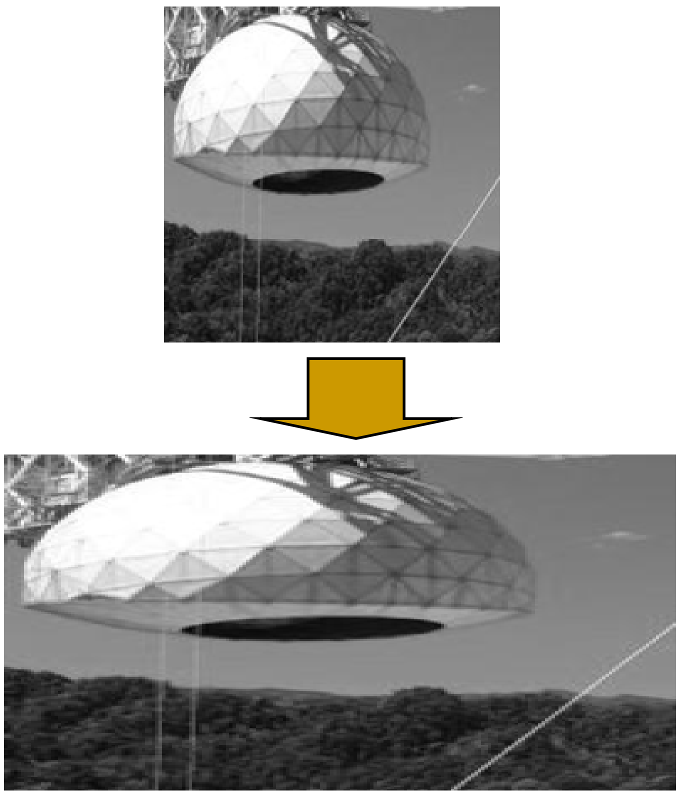
- All off-diagonal elements are zero
- Diagonal elements are non-zero
- Scales the axes
  - May flip axes

Diagonal matrix to transform images



- How?

Stretching



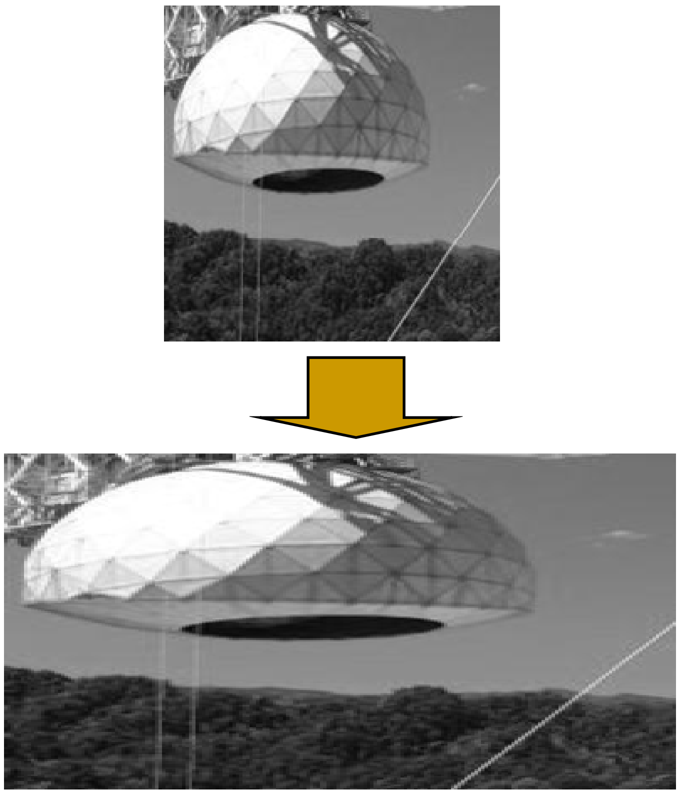
$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & 10 \\ 1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & 10 \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$

- Location-based representation
- Scaling matrix – only scales the X axis
  - The Y axis and pixel value are scaled by identity
- Not a good way of scaling.

Stretching

D =

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



$$A = \begin{bmatrix} 1 & .5 & 0 & 0 & . \\ 0 & .5 & 1 & .5 & . \\ 0 & 0 & 0 & .5 & . \\ 0 & 0 & 0 & 0 & . \\ . & . & . & . & . \end{bmatrix} (N\mathrm{x}2N)$$

$$\text{Newpic} = EA$$

- Better way

Modifying color

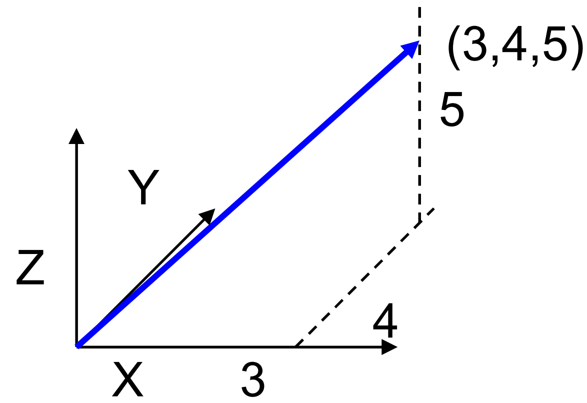
$$P = \begin{bmatrix} R & G & B \end{bmatrix}$$

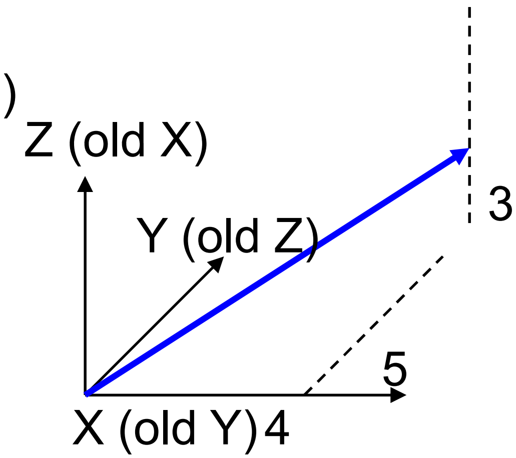
$$Newpic = P \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Scale only Green

Permutation Matrix

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

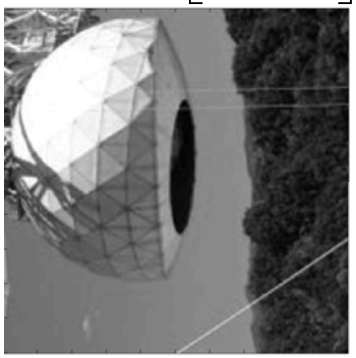




- A permutation matrix simply rearranges the axes
  - The row entries are axis vectors in a different order
  - The result is a combination of rotations and reflections
- The permutation matrix effectively *permutes* the arrangement of the elements in a vector

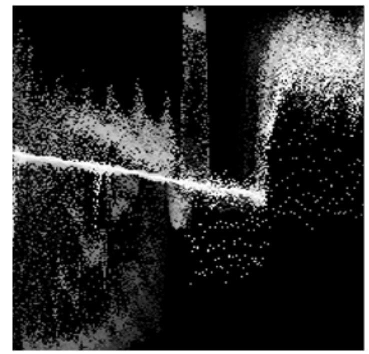
Permutation Matrix

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



$$\begin{bmatrix} 1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & 10 \\ 1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & 10 \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$

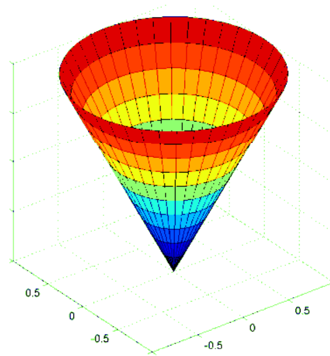
$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$



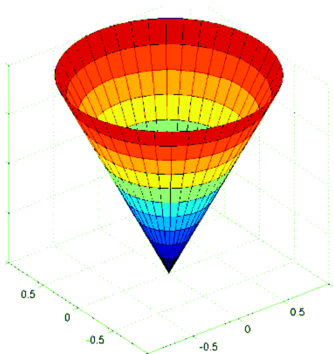
- Reflections and 90 degree rotations of images and objects

Permutation Matrix

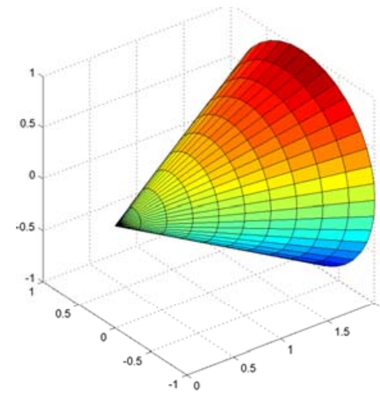
$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



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



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



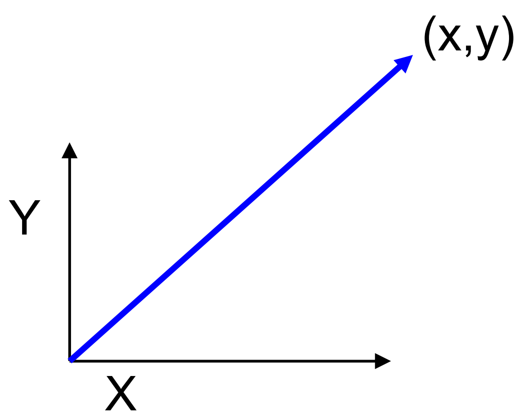
- Reflections and 90 degree rotations of images and objects
  - Object represented as a matrix of 3-Dimensional "position" vectors
  - Positions identify each point on the surface

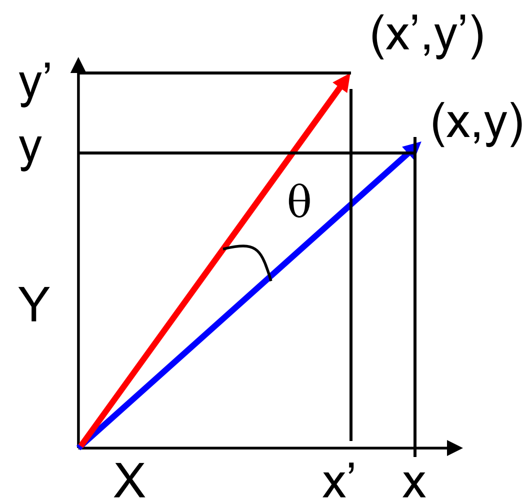
Rotation Matrix

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

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

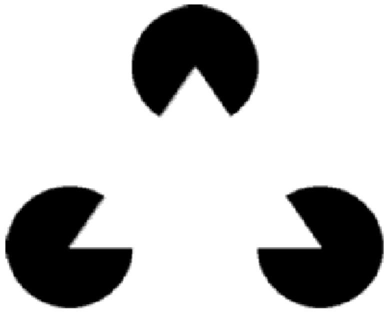
$$R_\theta X = X_{new}$$



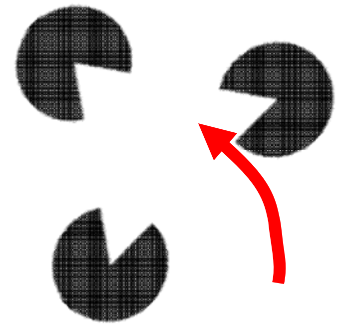


- A rotation matrix *rotates* the vector by some angle θ
- Alternately viewed, it rotates the axes
  - The new axes are at an angle θ to the old one

Rotating a picture



$$R = \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

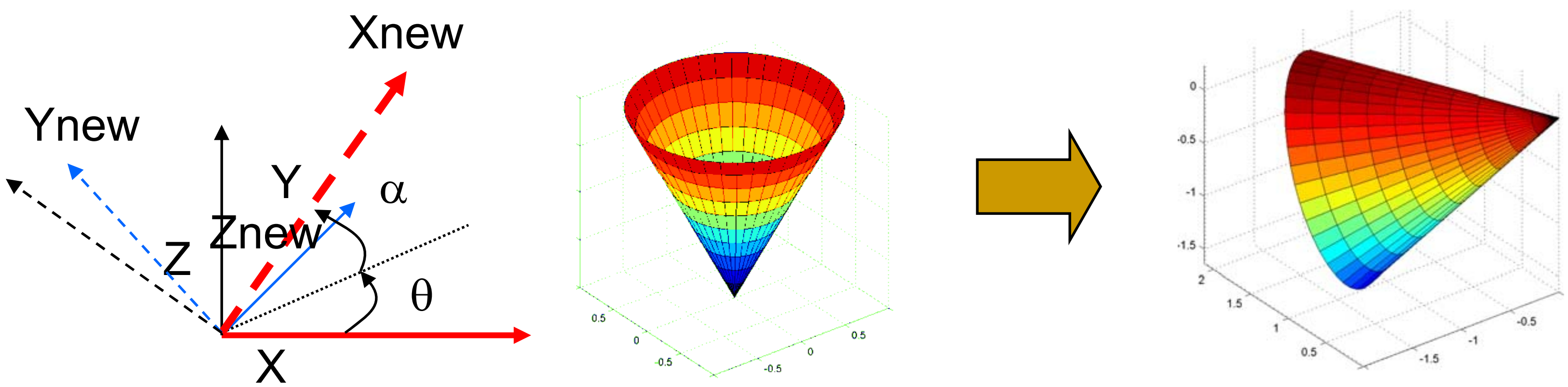


$$\begin{bmatrix} 1 & 1 & . & 2 & . & 2 & 2 & . & 2 & . & . \\ 1 & 2 & . & 1 & . & 5 & 6 & . & 10 & . & . \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -\sqrt{2} & . & \sqrt{2} & . & -3\sqrt{2} & -4\sqrt{2} & . & -8\sqrt{2} & . & . \\ \sqrt{2} & 3\sqrt{2} & . & 3\sqrt{2} & . & 7\sqrt{2} & 8\sqrt{2} & . & 12\sqrt{2} & . & . \\ 1 & 1 & . & 1 & . & 0 & 0 & . & 1 & . & 1 \end{bmatrix}$$

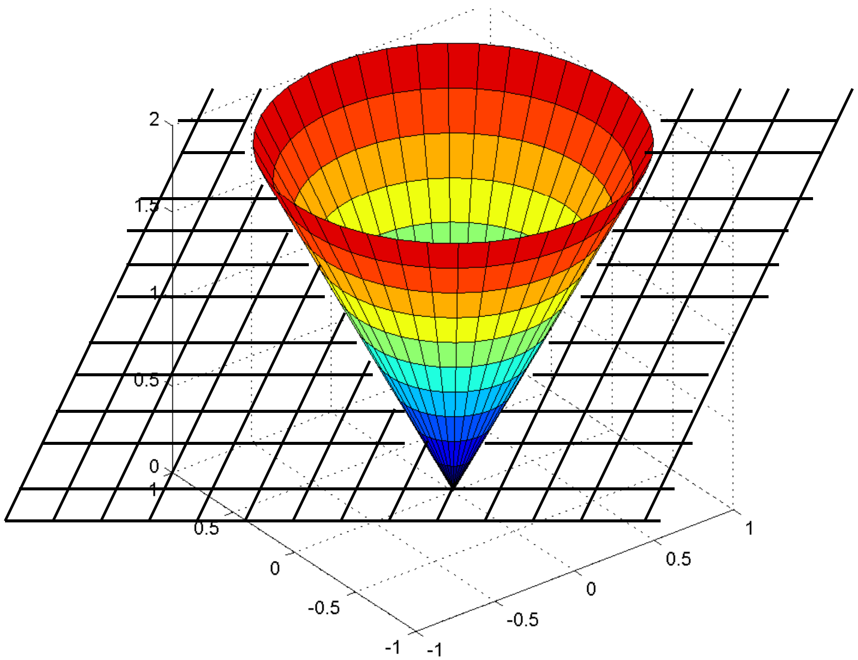
- Note the representation: 3-row matrix
  - Rotation only applies on the “coordinate” rows
  - The value does not change
  - Why is pacman grainy?

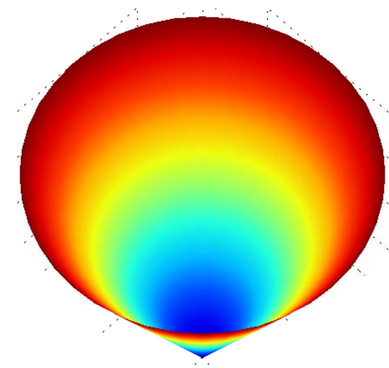
3-D Rotation



- 2 degrees of freedom
  - 2 separate angles
- What will the rotation matrix be?

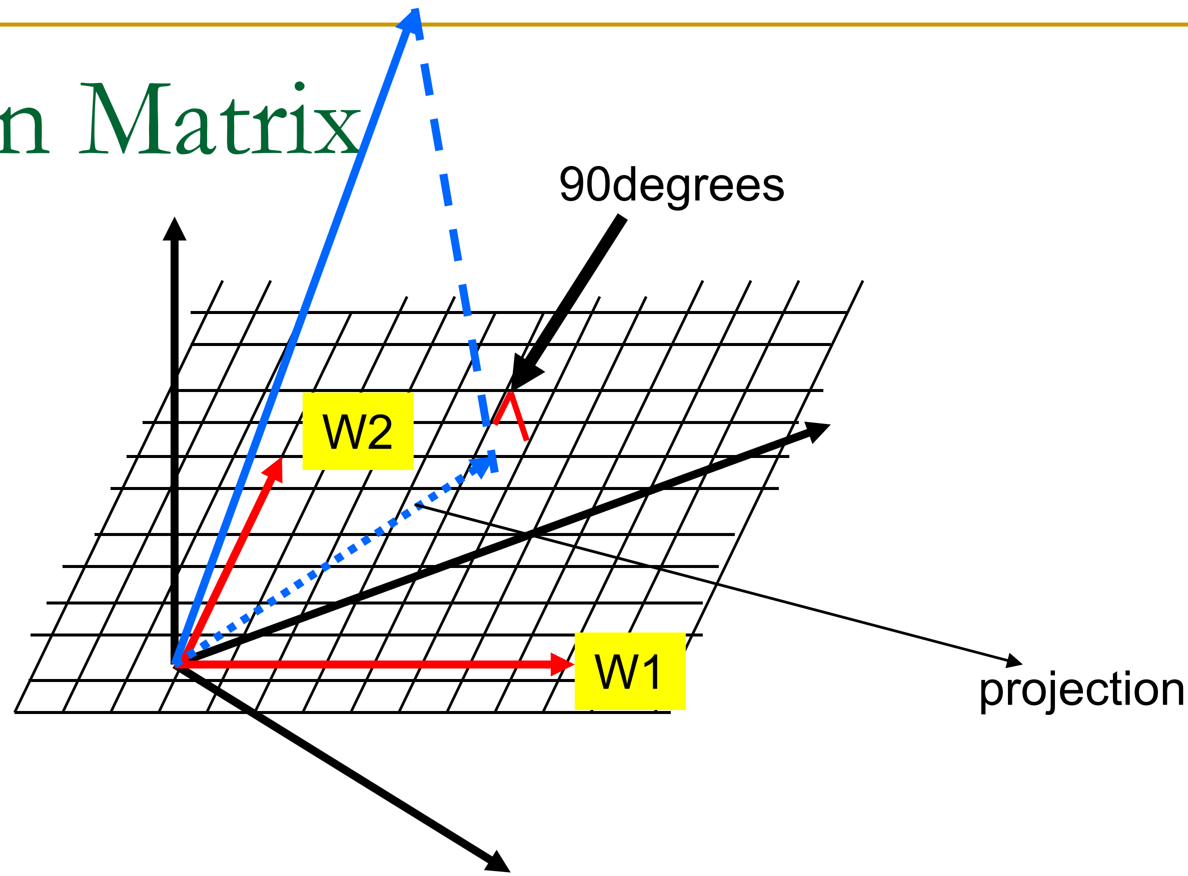
Projections





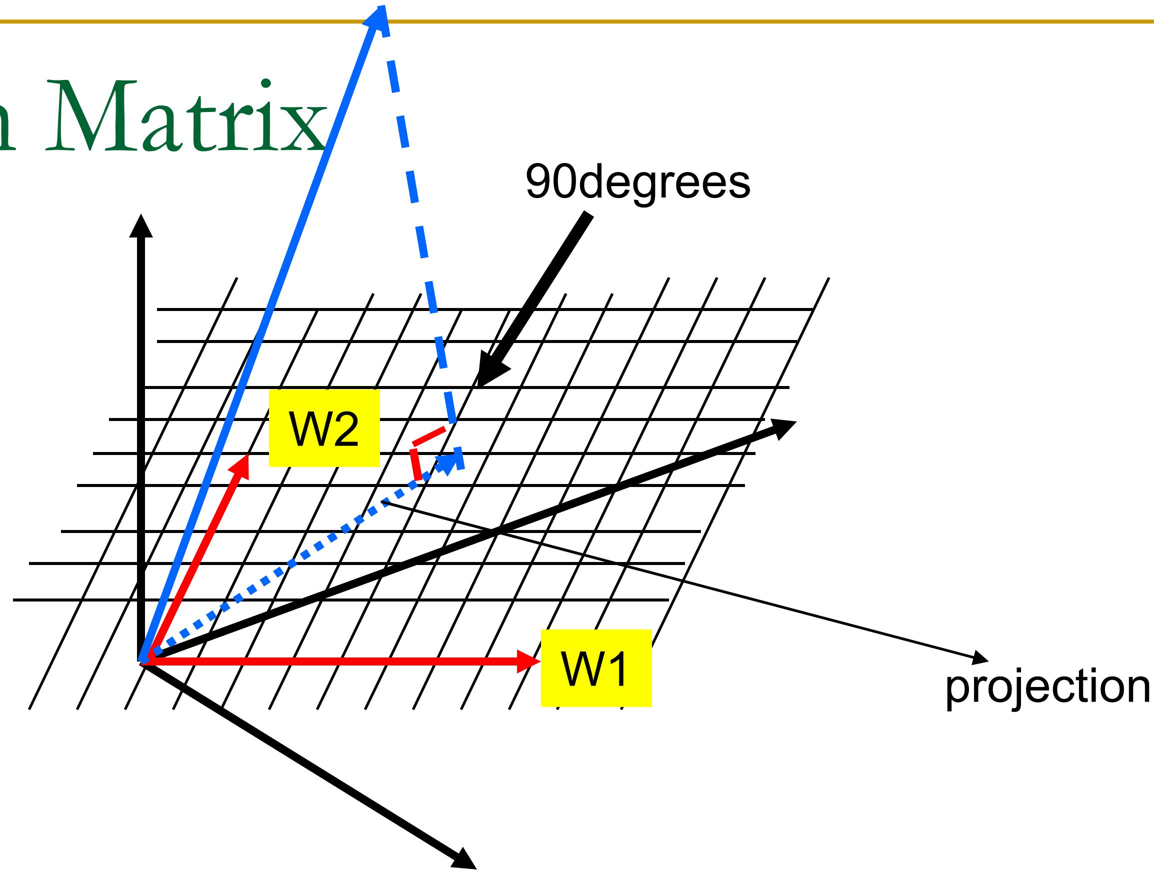
- What would we see if the cone to the left were transparent if we looked at it along the normal to the plane
  - The plane goes through the origin
  - Answer: the figure to the right
- How do we get this? Projection

Projection Matrix



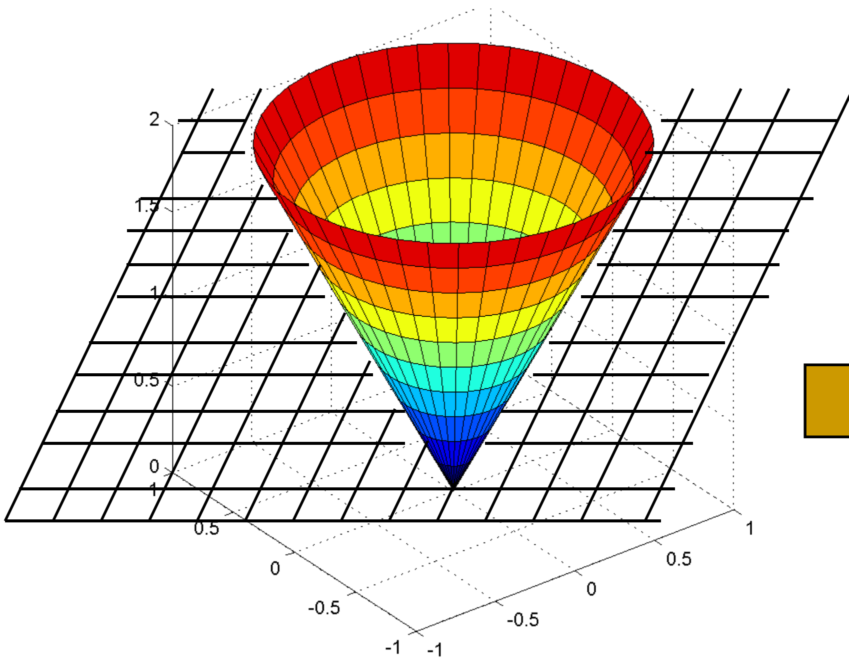
- Consider any plane specified by a set of vectors W_1, W_2..
  - Or matrix $[W_1\ W_2\ ..]$
  - Any vector can be projected onto this plane
  - The matrix A that rotates and scales the vector so that it becomes its projection is a projection matrix

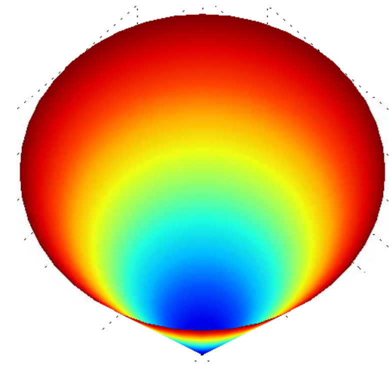
Projection Matrix



- Given a set of vectors W1, W2, which form a matrix W = [W1 W2..]
- The projection matrix that transforms any vector X to its projection on the plane is
  - $P = W (W^T W)^{-1} W^T$
    - We will visit matrix inversion shortly
- Magic – any set of vectors from the same plane that are expressed as a matrix will give you the same projection matrix
  - $P = V (V^T V)^{-1} V^T$

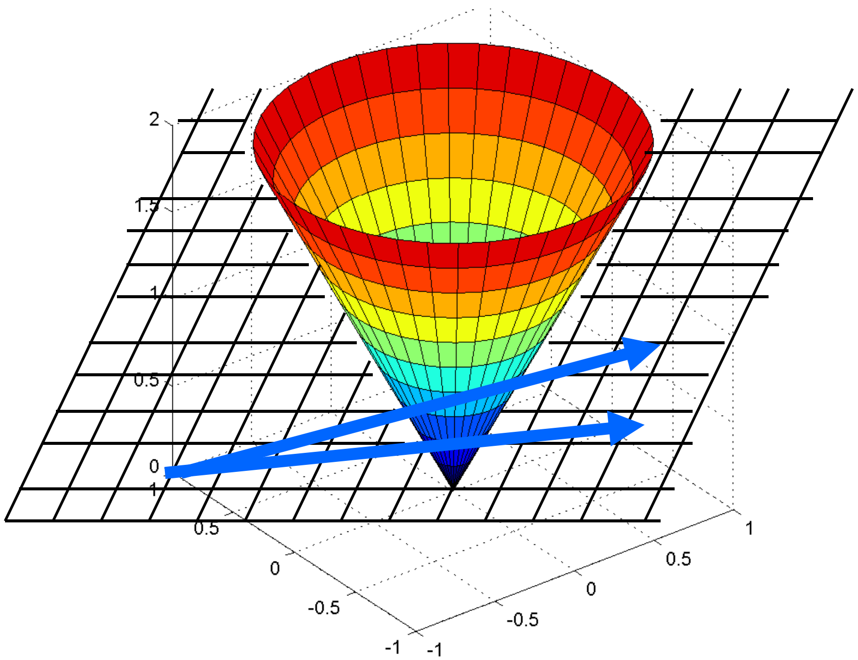
Projections

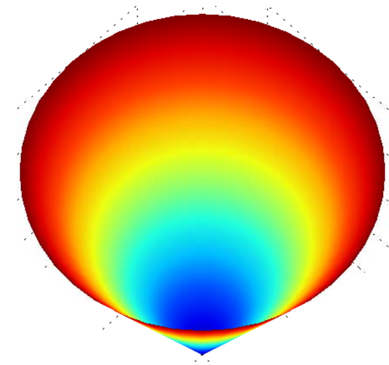




- HOW?

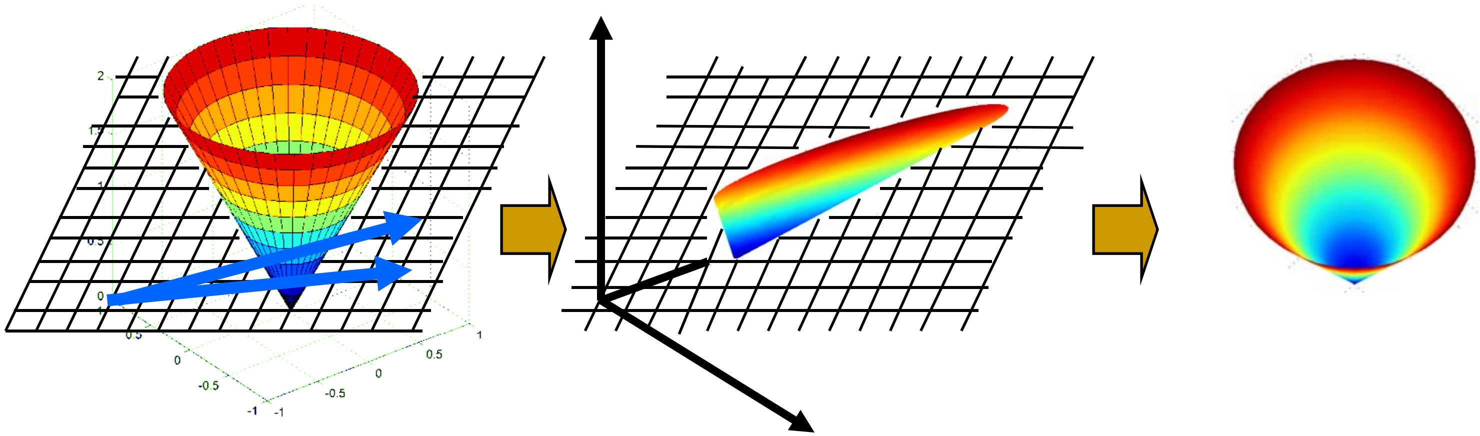
Projections





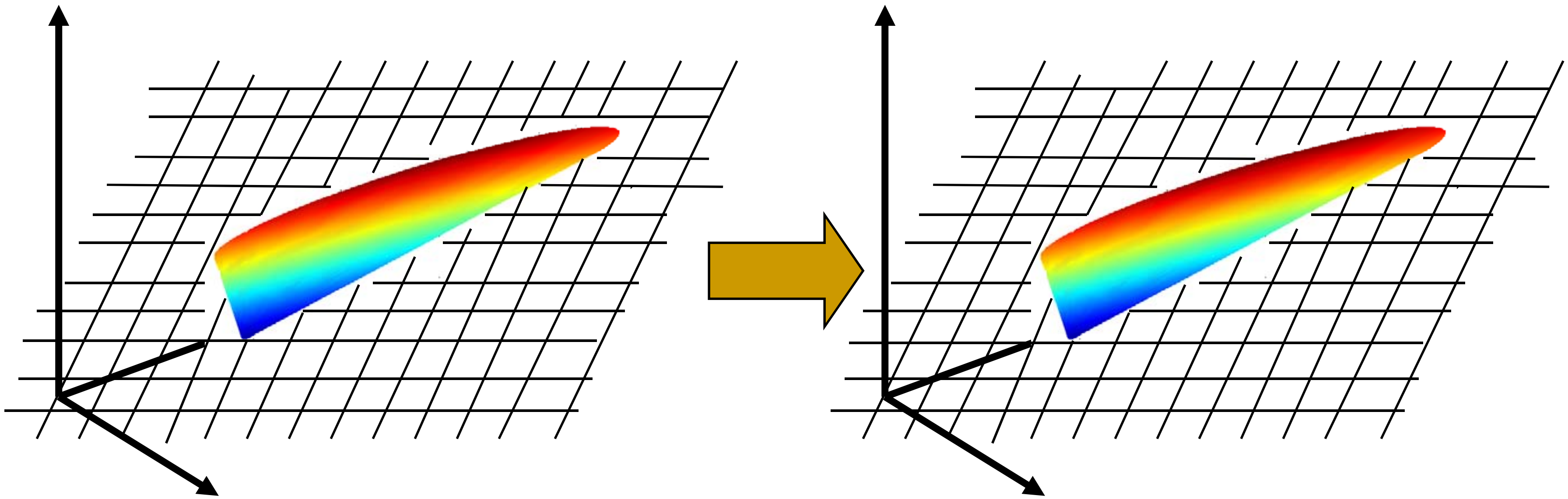
- Draw any two vectors W1 and W2 that lie on the plane
  - ***ANY two* so long as they have different angles**
- Compose a matrix W = [W1 W2]
- Compose the projection matrix $P = W (W^T W)^{-1} W^T$
- Multiply every point on the cone by P to get its projection
- View it ☺
  - I'm missing a step here – what is it?

Projections



- The projection actually projects it onto the plane, but you're still seeing the plane in 3D
  - The result of the projection is a 3-D vector
  - $P = W (W^TW)^{-1} W^T$ = 3x3, P*Vector = 3x1
  - The image must be rotated till the plane is in the plane of the paper
    - The Z axis in this case will always be zero and can be ignored
    - How will you rotate it? (remember you know W1 and W2)

Projection matrix properties



- The projection of any vector that is already on the plane is the vector itself
  - Px = x if 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
  - P (Px) = Px
  - That is because Px is already on the plane
- Projection matrices are *idempotent*
  - $P^2 = P$
    - Follows from the above