Machine Learning for Signal Processing
Fundamentals of Linear Algebra

Class 2. 2 Sep 2014

Instructor: Bhiksha Raj
Administrivia

• Info on second TA still awaited from ECE

• Registration: Anyone on waitlist still?

• Homework 1: Will appear on Thursday.
  – Linear algebra
Overview

• Vectors and matrices
• Basic vector/matrix operations
• Vector products
• Matrix products
• Various matrix types
• Projections
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!

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

• Easier intuition

– Really convenient geometric interpretations

• Easy code translation!

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

\[ C = x \cdot A \cdot y \]
And other things you can do

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

Rotation + Projection + Scaling + Perspective

From Bach’s Fugue in Gm

Decomposition (NMF)
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
  
  $a = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$

- A **matrix** $A$ is a rectangular arrangement of a collection of scalars
  
  $A = \begin{bmatrix} 3.12 & -10 \\ 10.0 & 2 \end{bmatrix}$
Vectors in the abstract

• 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 in reality

- Vectors usually hold sets of numerical attributes
  - X, Y, Z coordinates
    - [1, 2, 0]
  - [height(cm) weight(kg)]
    - [175 72]
  - A location in Manhattan
    - [3av 33st]
- A series of daily temperatures
- Samples in an audio signal
- Etc.
Matrices

- Matrices can be square or rectangular

\[ S = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad R = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}, \quad M = \begin{bmatrix} \vdots \\ \vdots \end{bmatrix} \]

- Can hold data
  - Images, collections of sounds, etc.
  - Or represent *operations* as we shall see
- A matrix can be vertical stacking of row vectors

\[ R = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \]

- Or a horizontal arrangement of column vectors

\[ R = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \]
Dimensions of a matrix

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

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

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

\[ S = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad R = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \]

- \( S \) = 2 x 2 matrix
- \( R \) = 2 x 3 matrix
- Pacman = 321 x 399 matrix
Representing an image as a matrix

- 3 pacmen
- A 321 x 399 matrix
  - Row and Column = position
- A 3 x 128079 matrix
  - Triples of x,y and value
- A 1 x 128079 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
Basic arithmetic operations

- Addition and subtraction
  - Element-wise operations

\[
\begin{align*}
a + b &= \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_1 + b_1 \\ a_2 + b_2 \\ a_3 + b_3 \end{bmatrix} \\
a - b &= \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_1 - b_1 \\ a_2 - b_2 \\ a_3 - b_3 \end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
A + B &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{bmatrix}
\end{align*}
\]
Vector Operations

- Operations tell us how to get from origin to the result of the vector operations
  - \((3,4,5) + (3,-2,-3) = (6,2,2)\)
Operations example

• Adding random values to different representations of the image

2 Sep 2014
11-755/18-797
Vector norm

- Measure of how long a vector is: 
  - Represented as $\|\mathbf{x}\|$
  - Geometrically the shortest distance to travel from the origin to the destination 
    - As the crow flies 
    - Assuming Euclidean Geometry

- MATLAB syntax: 
  norm(x)
Transposition

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

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

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

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

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

\[ M = \begin{bmatrix} \text{image} \\ \text{image} \end{bmatrix}, \quad M^T = \begin{bmatrix} \text{image} \\ \text{image} \end{bmatrix} \]

• MATLAB syntax: a’
Vector multiplication

• Multiplication by scalar

\[ d \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} da \\ db \\ dc \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} d \\ e \\ f \end{bmatrix} = a \cdot d + b \cdot e + c \cdot f \]

• Dot product, or inner product
  – Vectors must have the same number of elements
  – Row vector times column vector = scalar

\[ \begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} d \\ e \\ f \end{bmatrix} = a \cdot d + b \cdot e + c \cdot f \]

• Outer product or vector direct product
  – Column vector times row vector = matrix

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

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

\([200\text{yd} \ 1600\text{yd}]\)
\(\text{norm} \approx 1612\)

\([770\text{yd} \ 300\text{yd}]\)
\(\text{norm} \approx 826\)
Vector dot product

• Vectors are spectra
  – Energy at a discrete set of frequencies
  – Actually 1 x 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
  - $\frac{C \cdot E}{|C||E|} = 0.1$
  - Not very much
- How much of C is in C2?
  - $\frac{C \cdot C2}{|C|/|C2|} = 0.5$
  - Not bad, you can fake it
- To do this, C, E, and C2 must be the same size
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
Multiplying a vector by a matrix

• Generalization of vector scaling

\[
\begin{bmatrix}
a \\
b \\
c
\end{bmatrix} \cdot d = \begin{bmatrix}
ad \\
bd \\
bd
\end{bmatrix}
\]

– **Left multiplication**: Dot product of each vector pair

\[
A \cdot B = \begin{bmatrix}
\leftarrow a_1 \\
\leftarrow a_2 \\
\rightarrow
\end{bmatrix} \cdot \begin{bmatrix}
\uparrow b \\
\downarrow
\end{bmatrix} = \begin{bmatrix}
a_1 \cdot b \\
a_2 \cdot b
\end{bmatrix}
\]

– Dimensions must match!!
  • No. of columns of matrix = size of vector
  • Result inherits the number of rows from the matrix
Multiplying a vector by a matrix

• Generalization of vector multiplication

\[ d \begin{bmatrix} a & b & c \end{bmatrix} = \begin{bmatrix} da & db & dc \end{bmatrix} \]

– Right multiplication: Dot product of each vector pair

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

– Dimensions must match!!
  • No. of rows of matrix = size of vector
  • Result inherits the number of columns from the matrix
Multiplication of vector space by matrix

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

- 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

\[
\begin{bmatrix}
1 & 3 & 0 \\
\cdot & \cdot & 0 \\
9 & 24 & \cdot \\
\cdot & \cdot & 1
\end{bmatrix}
\begin{bmatrix}
1 \\
2 \\
1
\end{bmatrix}
= 
\begin{bmatrix}
7 \\
\cdot \\
2
\end{bmatrix}
\]
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
Multiplying matrices

- Simple vector multiplication: Vector outer product

\[
\mathbf{a} \mathbf{b} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \cdot \begin{bmatrix} b_1 & b_2 \end{bmatrix} = \begin{bmatrix} a_1 b_1 & a_1 b_2 \\ a_2 b_1 & a_2 b_2 \end{bmatrix}
\]
Multiplying matrices

• Generalization of vector multiplication
  – Outer product of dot products!!

\[
A \cdot B = \begin{bmatrix}
\leftarrow a_1 & \rightarrow \\
\leftarrow a_2 & \rightarrow
\end{bmatrix} \begin{bmatrix}
\uparrow b_1 \\
\downarrow b_2
\end{bmatrix} = \begin{bmatrix}
a_1 \cdot b_1 & a_1 \cdot b_2 \\
a_2 \cdot b_1 & a_2 \cdot b_2
\end{bmatrix}
\]

– 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
Multiplying matrices: Another view

• Simple vector multiplication: Vector inner product

\[ \mathbf{a} \mathbf{b} = \begin{bmatrix} a_1 & a_2 \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = a_1 b_1 + a_2 b_2 \]
Matrix multiplication: another view

\[
A \cdot B = \begin{bmatrix}
\uparrow & \uparrow \\
\downarrow & \downarrow \\
\end{bmatrix}
\begin{bmatrix}
\leftarrow b_1 & \rightarrow \\
\leftarrow b_2 & \rightarrow \\
\end{bmatrix}
= a_2 b_2 + a_2 b_2
\]

\[
\begin{bmatrix}
a_{11} & \cdots & a_{1N} \\
a_{21} & \cdots & a_{2N} \\
\vdots & \ddots & \vdots \\
a_{M1} & \cdots & a_{MN}
\end{bmatrix}
\begin{bmatrix}
b_{11} & \cdots & b_{NK} \\
\vdots & \ddots & \vdots \\
b_{N1} & \cdots & b_{NK}
\end{bmatrix}
= \begin{bmatrix}
a_{11} \\
a_{12} \\
\vdots \\
a_{1N}
\end{bmatrix}
+ \begin{bmatrix}
a_{11} \\
a_{12} \\
\vdots \\
a_{1N}
\end{bmatrix}
+ \cdots + \begin{bmatrix}
a_{11} \\
a_{12} \\
\vdots \\
a_{1N}
\end{bmatrix}
+ \begin{bmatrix}
a_{11} \\
a_{12} \\
\vdots \\
a_{1N}
\end{bmatrix}
+ \cdots + \begin{bmatrix}
a_{11} \\
a_{12} \\
\vdots \\
a_{1N}
\end{bmatrix}
\]

- 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 + ....

- **Sum of outer products**
Why is that useful?

- 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: Mixing modulated spectra

• Sounds: Three notes modulated independently
Matrix multiplication: Mixing modulated spectra

- Sounds: Three notes modulated independently
Matrix multiplication: Image transition

- Image 1 fades out linearly
- Image 2 fades in linearly
Matrix multiplication: Image transition

- 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

- Image 1 fades out linearly
- Image 2 fades in linearly
The Identity Matrix

- 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

\[ Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]
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

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

- 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

\[ D = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \]

\[
A = \begin{bmatrix} 1 & .5 & 0 & 0 & . \\ 0 & .5 & 1 & .5 & . \\ 0 & 0 & 0 & .5 & . \\ 0 & 0 & 0 & 0 & . \\ \end{bmatrix} \quad (N \times 2N)
\]

\[ \text{Newpic} = EA \]

- Better way
- **Interpolate**
Modifying color

\[ P = \begin{bmatrix} R \\ G \\ B \end{bmatrix} \]

\[ \text{Newpic} = P \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

• Scale only Green
Permutation Matrix

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

- Reflections and 90 degree rotations of images and objects
Permutation Matrix

\[ P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

\[ P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \]

- 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

\[ x' = x \cos \theta - y \sin \theta \]
\[ y' = x \sin \theta + y \cos \theta \]

\[ R_\theta = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \]

\[ X = \begin{bmatrix} x \\ y \end{bmatrix} \]
\[ X_{\text{new}} = \begin{bmatrix} x' \\ y' \end{bmatrix} \]

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
Rotating a picture

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

\[
\begin{bmatrix}
1 & 1 & 2 & 2 & 2 & 2 & \ldots \\
1 & 2 & 1 & 5 & 6 & 10 & \ldots \\
1 & 1 & 1 & 0 & 0 & 1 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & -\sqrt{2} & \sqrt{2} & -3\sqrt{2} & -4\sqrt{2} & -8\sqrt{2} & \ldots \\
\sqrt{2} & 3\sqrt{2} & 3\sqrt{2} & 7\sqrt{2} & 8\sqrt{2} & 12\sqrt{2} & \ldots \\
1 & 1 & 1 & 0 & 0 & 1 & 1
\end{bmatrix}
\]

- 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?
Matrix Operations: Properties

• $A + B = B + A$
• $AB \neq BA$
• 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
Consider any plane specified by a set of vectors $W_1, W_2, \ldots$

- Or matrix $[W_1 \ W_2 \ldots]$.
- 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.
• Given a set of vectors $W_1, W_2$, which form a matrix $W = [W_1 \ W_2\ldots]$.

• The projection matrix to transform a 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?
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
- $P = W(W^T W)^{-1} W^T = 3 \times 3$, $P*Vector = 3 \times 1$
- 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
  - $P_x = x$ 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
  - $P(P_x) = P_x$
  - That is because $P_x$ is already on the plane

- Projection matrices are *idempotent*
  - $P^2 = P$

2 Sep 2014 • Follows from the above
Projections: A more physical meaning

- Let $W_1, W_2 .. W_k$ 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 $W_1, W_2, .. W_k$, is the projection of the data on the $W_1 .. W_k$ (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

- The spectrogram (matrix) of a piece of music

\[
M = \text{spectrogram; } W = \text{note}
\]

\[
P = W(W^TW)^{-1}W^T
\]

Projected Spectrogram = P * M
Projection: one note – cleaned up

- The spectrogram (matrix) of a piece of music

Floored all matrix values below a threshold to zero
Projection: multiple notes

- The spectrogram (matrix) of a piece of music

\[ M = \]

\[ W = \]

\[ P = W (W^T W)^{-1} W^T \]

- Projected Spectrogram = \( P \times M \)
Projection: multiple notes, cleaned up

- The spectrogram (matrix) of a piece of music

\[
\begin{align*}
P &= W (W^T W)^{-1} W^T \\
\text{Projected Spectrogram} &= P \ast M
\end{align*}
\]
Projection and Least Squares

• Projection actually computes a *least squared error* estimate

• For each vector V in the music spectrogram matrix
  – Approximation: $V_{\text{approx}} = a \cdot \text{note1} + b \cdot \text{note2} + c \cdot \text{note3}$

  \[
  V_{\text{approx}} = \begin{bmatrix}
  \text{note1} \\
  \text{note2} \\
  \text{note3}
  \end{bmatrix} \begin{bmatrix}
  a \\
  b \\
  c
  \end{bmatrix}
  \]

  – Error vector $E = V - V_{\text{approx}}$

  – Squared error energy for $V$ : $e(V) = \|E\|^2$

  – Total error = sum over all $V$ : $\sum_v e(V) = \sum_v e(V)$

• Projection computes $V_{\text{approx}}$ for all vectors such that Total error is minimized
  – It does not give you “a”, “b”, “c”.. Though
    • That needs a different operation – the inverse / pseudo inverse
Perspective

• The picture is the equivalent of “painting” the viewed scenery on a glass window
• Feature: The lines connecting any point in the scenery and its projection on the window merge at a common point
  – The eye
  – As a result, parallel lines in the scene *apparently* merge to a point
An aside on Perspective..

- Perspective is the result of convergence of the image to a point
- Convergence can be to multiple points
  - Top Left: One-point perspective
  - Top Right: Two-point perspective
  - Right: Three-point perspective
Representing Perspective

- Perspective was not always understood.
- Carefully represented perspective can create illusions.
Central Projection

- The positions on the “window” are scaled along the line.
- To compute \((x,y)\) position on the window, we need \(z\) (distance of window from eye), and \((x',y',z')\) (location being projected).

\[
\frac{x}{x'} = \frac{y}{y'} = \frac{z}{z'} \quad \text{Property of a line through origin}
\]

\[
\alpha = \frac{z}{z'}
\]

\[
x = \alpha x'
\]

\[
y = \alpha y'
\]
Homogeneous Coordinates

- Represent points by a triplet
  - Using yellow window as reference:
    - \((x,y) = (x,y,1)\)
    - \((x',y') = (x,y,c') \quad c' = \alpha'/\alpha\)
  - Locations on line generally represented as \((x,y,c)\)
    - \(x' = x/c' \quad y' = y/c'\)
Homogeneous Coordinates in 3-D

- Points are represented using FOUR coordinates
  - \((X,Y,Z,c)\)
  - “c” is the “scaling” factor that represents the distance of the actual scene
- Actual Cartesian coordinates:
  - \(X_{\text{actual}} = X/c,\ Y_{\text{actual}} = Y/c,\ Z_{\text{actual}} = Z/c\)
Homogeneous Coordinates

- In both cases, constant “c” represents distance along the line with respect to a reference window
  - In 2D the plane in which all points have values (x,y,1)
- Changing the reference plane changes the representation
- I.e. there may be multiple Homogenous representations (x,y,c) that represent the same cartesian point (x’, y’)

2 Sep 2014