Fundamentals of Linear Algebra

Class 2  30 Aug 2012

Instructor: Bhiksha Raj

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, 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!
  \[ x \cdot \cdot A \cdot \cdot y \longleftrightarrow \sum_y \sum_x a_{xy} \]
  - 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
  ```

  \[ C=x*A*y \]

And other things you can do

- Manipulate Images
- Manipulate Sounds
Scalars, vectors, matrices, …

- A scalar is a number
  - \( a = 2, a = 3.14, a = -1000 \), etc.

- A vector is a linear arrangement of a collection of scalars
  - \( \mathbf{a} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \)

- A matrix is a rectangular arrangement of a collection of scalars
  - \( \mathbf{A} = \begin{bmatrix} 3.12 & -10.0 & 2 \end{bmatrix} \)

- MATLAB syntax: \( a = [1 2 3], A = [1 2; 3 4] \)

Vectors

- Vectors usually hold sets of numerical attributes
  - X, Y, Z coordinates
  - \([\text{a, b, c, d}]\)
  - Earnings, losses, suicides
  - \([\text{50, 51, 00, 00, 0 3}]\)
  - A location in Manhattan
  - \([\text{3, 4; 3, 1}]\)

- Vectors are either column or row vectors
  - \( \mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \)
  - A sound can be a vector, a series of daily temperatures can be a vector, etc...

Vectors in the abstract

- Ordered collection of numbers
  - Examples: \([3 4 5], [a b c d]\), ...
  - \([3 4 5] \rightarrow [4 3 5] \) → Order is important

- Typically viewed as identifying (the path from origin to) a location in an \(N\)-dimensional space

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

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

- A matrix can be vertical stacking of row vectors

- Or a horizontal arrangement of column vectors

Dimensions of a matrix

- The matrix size is specified by the number of rows and columns
  - \( \mathbf{c} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \mathbf{r} = \begin{bmatrix} a & b & c \end{bmatrix} \)
  - \( c = 3 \times 1 \) matrix: 3 rows and 1 column
  - \( r = 1 \times 3 \) matrix: 1 row and 3 columns

- \( \mathbf{S} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}, \mathbf{R} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} \)

- \( S = 2 \times 2 \) matrix
- \( R = 2 \times 3 \) matrix
- Pacman = 321 x 399 matrix

Matrices

- Matrices can be square or rectangular

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

- A matrix can be vertical stacking of row vectors

- Or a horizontal arrangement of column vectors

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
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**
    
    \[
    \begin{align*}
    \mathbf{a} + \mathbf{b} & = \begin{bmatrix} a_1 + b_1 \\ a_2 + b_2 \\ a_3 + b_3 \end{bmatrix}  \\
    \mathbf{a} - \mathbf{b} & = \begin{bmatrix} a_1 - b_1 \\ a_2 - b_2 \\ a_3 - b_3 \end{bmatrix}  \\
    \mathbf{A} + \mathbf{B} & = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{bmatrix}
    \end{align*}
    \]

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

### Operations example

#### Adding random values to different representations of the image

### Vector norm

- **Measure of how big a vector is:**
  - Represented as \( \| \mathbf{v} \| \)
  - 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)
  - \( \mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \)
    - \( \mathbf{x}' = \begin{bmatrix} a & b & c \end{bmatrix} \)
    - \( \mathbf{y} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \)
    - \( \mathbf{y}' = \begin{bmatrix} a & b & c \end{bmatrix} \)
- A transposed matrix gets all its row (or column) vectors transposed in order
  - \( \mathbf{X} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \)
    - \( \mathbf{X}' = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix} \)
  - **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 \\
    d & e & f \\
    
    \end{bmatrix}
    \begin{bmatrix}
    & \\
    a & \\
    d & \\
    
    \end{bmatrix}
    = a + d + e + c + f
    \]
- Outer product or vector direct product
  - Column vector times row vector = matrix
  \[
  \begin{bmatrix}
  a & b & d & e & f \\
  
  \end{bmatrix}
  \begin{bmatrix}
  a & d & e & f \\
  b & c & d & e \\
  
  \end{bmatrix}
  = \begin{bmatrix}
  a & b & c & d & e & f \\
  
  \end{bmatrix}
  \]
- MATLAB syntax: \texttt{a \* b}

Vector dot product in Manhattan
- Example:
  - Coordinates are yards, not ave/st
  - \( \mathbf{a} = [2001600] \\
  \mathbf{b} = [770300] \)
- The dot product of the two vectors relates to the length of a projection
  - How much of \( \mathbf{a} \) is projected onto \( \mathbf{b} \)?
  - Must normalize by the length of the "target" vector
    \[
    \mathbf{x}_1 \cdot \mathbf{x}_2 = \frac{[200\, 1600] \cdot [770\, 300]}{\sqrt{200^2 + 1600^2} \cdot \sqrt{770^2 + 300^2}} = 99.54
    \]

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 outer 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

Multiplying a vector by a matrix
- Generalization of vector multiplication
  - Left multiplication: Dot product of each vector pair
    \[
    \mathbf{A} \cdot \mathbf{B} = \begin{bmatrix}
    a_1 & a_2 & \cdots & a_n \\
    b_1 & b_2 & \cdots & b_n \\
    \end{bmatrix}
    \]
  - Dimensions must match!!
    - No. of columns of matrix = size of vector
    - Result inherits the number of rows from the matrix
- MATLAB syntax: \texttt{a \* b}
Multiplying a vector by a matrix

- **Generalization of vector multiplication**
  - **Right multiplication**: Dot product of each vector pair
    \[ A \cdot B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix} = [a, b, a, b] \]
  - Dimensions must match!!
    - No. of rows of matrix = size of vector
    - Result inherits the number of columns from the matrix
- **MATLAB syntax**: `a*b`

Multiplication of vector space by matrix

- **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

- 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

- 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

Multiplying matrices

- Generalization of vector multiplication
  - **Outer product of dot products!!**
  
  $A \cdot B = \begin{bmatrix} a_1 \cdot b_1 & a_1 \cdot b_2 & \cdots & a_1 \cdot b_n \\ a_2 \cdot b_1 & a_2 \cdot b_2 & \cdots & a_2 \cdot b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_m \cdot b_1 & a_m \cdot b_2 & \cdots & a_m \cdot b_n \end{bmatrix}$

- 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: $a*b$

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

- 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

Diagonal Matrix

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

Diagonal matrix to transform images

- How?

Stretching

- 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

\[ D = \begin{bmatrix} 1 & .5 & 0 & 0 \\ .5 & 1 & .5 & 0 \\ 0 & 0 & 0 & .5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

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

\[ \text{Newpic} = EA \]

- Better way

---

Modifying color

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

\[ \text{Newpic} = P \]

- Scale only Green

---

Permutation Matrix

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

- 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

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

- Reflections and 90 degree rotations of images and objects

---

Permutation Matrix

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

- 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

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

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

\[ R_\theta X = X_{\text{new}} \]

- A rotation matrix rotates the vector by some angle \( \theta \)
  - Object represented as a matrix of 3 Dimensional “position” vectors
  - Positions identify each point on the surface
  - The new axes are at an angle \( \theta \) to the old one
Rotating a picture

- 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

- $A + B = B + A$
- $AB \neq BA$

Projections

- 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

Projection Matrix

- Consider any plane specified by a set of vectors $W_1, W_2$.
  - 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

Projection Matrix

- Given a set of vectors $W_1, W_2$, which form a matrix $W = [W_1 \ W_2\ldots]$
  - The projection matrix that transforms any vector $X$ to its projection on the plane is $P = W(W'W)^{-1}W'$
  - 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
  - $P = V(V'V)^{-1}V'$
**Draw any two vectors \( W_1 \) and \( W_2 \) that lie on the plane.

**Compose a matrix \( W = [W_1 \ W_2] \).

**Compose the projection matrix \( P = W \ (W^T \ W)^{-1} \ W^T \).

**Multiply every point on the cone by \( P \) to get its projection.

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.
- \( P = W \ (W^T \ W)^{-1} \ W^T \).
- 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 \( W_1 \) and \( W_2 \)).

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.

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.
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).

Representing Perspective

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

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 \(W_1, W_2, ..., W_k\) (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 = \(P \ast M\)

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
- \[ P = W (W^TW)^{-1} W^T \]
- Projected Spectrogram = \( P \cdot M \)

Projection and Least Squares

- Projection actually computes a least squared error estimate
- For each vector \( V \) in the music spectrogram matrix
  - Approximation: \( V_{\text{approx}} = a \cdot \text{note1} + b \cdot \text{note2} + c \cdot \text{note3} \)
  - \( V_{\text{approx}} = \begin{bmatrix} a & b & c \end{bmatrix} \)
- Error vector \( E = V - V_{\text{approx}} \)
- Squared error energy for \( V \): \( \|e(V)\|^2 = \text{norm}(E)^2 \)
- Total error = sum over all \( V \): \( \sum \|e(V)\|^2 \)
- 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