CE 88: Fall 2017 Data Science Connector Course

Data Science for Smart Cities

Linear Algebra

Instructor: Milad Memarzadeh
GSI: Sangjae Bae
Mondays 12-2pm, 406 Davis Hall

Vectors

A vector is a 1D array of values

We use the notation $x \in \mathbb{R}^n$ to denote that x is an n-dimensional vector with real-valued entries:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_n \end{bmatrix}$$

We use the notation x_i to denote the ith entry of x

By default, we consider vectors to represent *column* vectors, if we want to consider a row vector, we use the transform notation x^T

Matrices

A matrix is a 2D array of values

We use the notation $A \in \mathbb{R}^{m \times n}$ to denote a real-valued matrix with m rows and n columns:

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ . & . & \dots & . \\ A_{m1} & A_{m2} & \dots & A_{mn} \end{bmatrix}$$

We use A_{ij} to denote the entry in row i and column j

Use the notation $A_{i:}$ to refer to row i, $A_{:j}$ to refer to column j

Matrices and linear algebra

Matrices are:

1. The “obvious” way to store tabular data (particularly numerical entries) in an efficient manner

2. The foundation of linear algebra, how we write down and operate upon (multi-variate) systems of linear equations

Understanding both these perspectives is critical for virtually all data science analysis algorithms.

Matrices as tabular data

Given the "grades" table of three sample students:

Student ID	HW1	HW2
5	100	80
12	60	80
40	100	100

Natural to represent this data as a matrix:

$$A \in \mathbb{R}^{3\times 2} = \begin{bmatrix} 100 & 80 \\ 60 & 80 \\ 100 & 100 \end{bmatrix}$$

Higher dimensional matrices

From a data storage standpoint, it is easy to generalize 1D vector and 2D matrices to higher dimensional ND storage

“Higher dimensional matrices” are called tensors

There is also an extension for linear algebra to tensors, but be aware:

most tensor use cases you see are *not* really talking about true tensors in the linear algebra sense

Systems of linear equations

Matrices and vectors also provide a way to express and analyze systems of linear equations

Consider two linear equations, two unknowns

$$\begin{aligned} 4x_1 \;-\; 5x_2 &= -13 \\ -2x_1 \;+\; 3x_2 &= 9 \end{aligned}$$

We can write this using matrix notation as

$$Ax = b$$

$$A = \begin{bmatrix} 4 & -5 \\ -2 & 3 \end{bmatrix}, \qquad b = \begin{bmatrix} -13 \\ 9 \end{bmatrix}, \qquad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Basic matrix operations

For A, $B \in \mathbb{R}^{m \times n}$, matrix addition/subtraction is just the elementwise addition or subtraction of entries

$$C \in \mathbb{R}^{m \times n} = A + B \iff C_{ij} = A_{ij} + B_{ij}$$

For $A \in \mathbb{R}^{m \times n}$, transpose is an operator that "flips" rows and columns

$$C \in \mathbb{R}^{n \times m} = A^T \iff C_{ji} = A_{ij}$$

For $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$ matrix multiplication is defined as

$$C \in \mathbb{R}^{m \times p} = AB \iff C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Matrix multiplication is associative ($A(BC) = (AB)C$), distributive ($A(B + C) = AB + AC$), not commutative ($AB \neq BA$)

Matrix inverse

The identity matrix $I \in \mathbb{R}^{n \times n}$, is a square matrix with ones on diagonal and zeros elsewhere, has property that for $A \in \mathbb{R}^{m \times n}$

$$AI = IA = A$$

For a *square* matrix $A \in \mathbb{R}^{n \times n}$, matrix inverse $A^{-1} \in \mathbb{R}^{n \times n}$ is the matrix such that

$$AA^{-1} = I = A^{-1}A$$

For example, for a system of linear equations $Ax = b$, solution is easily written using the inverse

$$x = A^{-1}b$$

Inverse does not exist for all matrices (conditions on linear independence of rows/columns of A)

Some useful properties

Transpose of matrix multiplication, $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$

$$(AB)^T = B^T A^T$$

Inverse of product, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times n}$ (both square and invertible)

$$(AB)^{-1} = B^{-1} A^{-1}$$

Inner product: for$x, y \in \mathbb{R}^n$, special case of matrix multiplication

$$x^T y \in \mathbb{R} = \sum_{i=1}^{n} x_i y_i$$

Numpy

In Python, the standard library for matrices, vectors, and linear algebra is Numpy

Numpy provides *both* a framework for storing tabular data as multidimensional arrays *and* linear algebra routines

Important note: numpy ndarrays are multi-dimensional arrays, not matrices and vectors (there are just routines that support them acting like matrices or vectors)