2.1. Linear Algebra Preliminaries

Lecture based on "Dive into Deep Learning" **http://D2L.AI** (Zhang et al., 2020)

Prof. Dr. Christoph Lippert

Digital Health & Machine Learning

Scalars

Example

The temperature in Palo Alto is 52 degrees Fahrenheit.
$c = \frac{5}{9}(f - 32)$
5, 9, and 32 are **scalar** values.
c and f are **variables** representing unknown scalar values.

Example

$x \in \mathbb{R}$

Example

$x, y \in \{0, 1\}$

A **vector** $\mathbf{x} \in \mathbb{R}^n$ is a list of n scalars.

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

- x_i or $[\mathbf{x}]_i$: i^{th} **element**, **entry**, or **component** of $\mathbf{x}$
- Vectors are by convention **columns**.
- We obtain **row** vector by **transposing** a column vector.

$$\mathbf{x}^\top = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}$$

A **matrix** $\mathbf{A} \in \mathbb{R}^{m \times n}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

- Each **element** a_{ij} or $[\mathbf{A}]_{ij}$ belongs to the i^{th} row and j^{th} column
- The **shape** of $\mathbf{A}$ is (m, n) or $m \times n$.
- If $m = n$ then $\mathbf{A}$ is a **square matrix**.
- $\mathbf{A}^\top$ is $\mathbf{A}$'s **transpose**

$$\mathbf{A}^\top = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{mn} \end{bmatrix}.$$

Tensors are denoted with capital letters sans serifs (e.g., X, Y, and Z)

- x_{ijk} scalar entry of third-order tensor with indices i, j, and k
- $[\mathsf{X}]_{1,2i-1,3}$ (Dimensionality not specified without order of tensor)
- vectors are first-order tensors
- matrices are send-order tensors

Example

Images are third-order tensors with 3 axes corresponding to the height, width, and an RGB **channel** axis.

The result of **elementwise operations** over tensors of the same shape will have the same shape.

Adding two matrices $\mathbf{A}$ and $\mathbf{B}$ of the same shape (m, n) performs elementwise addition over these two matrices.

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \dots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & \dots & a_{2n} + b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \dots & a_{mn} + b_{mn} \end{bmatrix}.$$

The **Hadamard product** of matrices $\mathbf{A}$ and $\mathbf{B}$

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11} b_{11} & a_{12} b_{12} & \dots & a_{1n} b_{1n} \\ a_{21} b_{21} & a_{22} b_{22} & \dots & a_{2n} b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} b_{m1} & a_{m2} b_{m2} & \dots & a_{mn} b_{mn} \end{bmatrix}.$$

Reduction

The sum of the elements of an $m \times n$ matrix $\mathbf{A}$ could be written

$$\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}$$

.

The sum **reduces** a tensor along all its axes to a scalar.

We can also reduced along individual axes.

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_m^\top \end{bmatrix}$$

$$\begin{aligned} \mathbf{c}^\top &= \sum_{i=1}^{m} \mathbf{a}_i^\top \\ &= \begin{bmatrix} \sum_{i=1}^{m} a_{i1} & \sum_{i=1}^{m} a_{i2} & \dots & \sum_{i=1}^{m} a_{in} \end{bmatrix} \end{aligned}$$

Dot Products

Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, their **dot product** $\mathbf{x}^\top \mathbf{y}$ (or $\langle \mathbf{x}, \mathbf{y} \rangle$) is

$$\mathbf{x}^\top \mathbf{y} = \sum_{i=1}^{d} x_i y_i$$

.

The **matrix-vector product** of $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{x} \in \mathbb{R}^n$ is

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_m^\top \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{a}_1^\top \mathbf{x} \\ \mathbf{a}_2^\top \mathbf{x} \\ \vdots \\ \mathbf{a}_m^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^m$$

Matrix-Matrix Multiplication

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & \cdots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{k1} & b_{k2} & \cdots & b_{km} \end{bmatrix}.$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_n^\top \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \end{bmatrix}.$$

$$\mathbf{C} = \mathbf{AB} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_n^\top \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1^\top \mathbf{b}_1 & \mathbf{a}_1^\top \mathbf{b}_2 & \cdots & \mathbf{a}_1^\top \mathbf{b}_m \\ \mathbf{a}_2^\top \mathbf{b}_1 & \mathbf{a}_2^\top \mathbf{b}_2 & \cdots & \mathbf{a}_2^\top \mathbf{b}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_n^\top \mathbf{b}_1 & \mathbf{a}_n^\top \mathbf{b}_2 & \cdots & \mathbf{a}_n^\top \mathbf{b}_m \end{bmatrix}.$$

In deep learning, objectives are often expressed as **norms**:

- **minimize** the distance between predictions and the ground-truth observations.
- Assign vector representations to items (words, products, news articles) such that the distance between similar items is minimized, and the distance between dissimilar items is maximized.

A **norm** is a function $\|\cdot\| \to \mathbb{R}$ that satisfies

(i) $\|\mathbf{x}\| \geq 0$, with equality if and only if $\mathbf{x} = \mathbf{0}$

(ii) $\|\alpha\mathbf{x}\| = |\alpha|\|\mathbf{x}\|$

(iii) $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (the **triangle inequality**)

for all vectors $\mathbf{x}, \mathbf{y}$ and scalars α.

We will typically only be concerned with a few specific norms on $\mathbb{R}^n$:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \qquad (p \geq 1)$$

$$\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

1- and 2-norms are special cases of the p-norm.

The ∞-norm is the limit of the p-norm as p tends to infinity.

We require $p \geq 1$ for the general definition of the p-norm because the triangle inequality fails to hold if $p < 1$.

The **Frobenius norm** of a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ is

$$\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}^2}.$$

Analogus to ℓ_2 norms of vectors.

Summary

- Scalars, vectors, matrices, and tensors are basic mathematical objects in linear algebra.
- Vectors generalize scalars, and matrices generalize vectors.
- Scalars, vectors, matrices, and tensors have 0, 1, 2, and an arbitrary number of axes, respectively.
- A tensor can be reduced along the specified axes by ‘sum‘ and ‘mean‘.
- Elementwise multiplication of two matrices is called their Hadamard product.
  It is different from matrix multiplication.
- In deep learning, we often work with norms such as the ℓ_1 norm, the ℓ_2 norm, and the Frobenius norm.