

Unit 01

Chapter - Linear Algebra

Scalars and Vectors

A **scalar** is just a single number, in contrast to most of the other objects studied in linear algebra, which are usually arrays of multiple numbers. For example, we might say “Let $s \in \mathbb{R}$ be the slope of the line,” while defining a real-valued scalar, or “Let $n \in \mathbb{N}$ be the number of units,” while defining a natural number scalar.

A **vector** is an array of numbers. The numbers are arranged in order. We can identify each individual number by its index in that ordering. If each element is in \mathbb{R} , and the vector has n elements, then the vector lies in the set formed by taking the Cartesian product of \mathbb{R} n times, denoted as \mathbb{R}^n . We can think of vectors as identifying points in space, with each element living the coordinate along a different axis.

Matrices and Tensors

Matrices: A matrix is a 2-D array of numbers, so each element is identified by two indices instead of just one. We usually give matrices upper-case variable names with bold typeface, such as A . If a real-valued matrix A has a height of m and a width of n , then we say that $A \in \mathbb{R}^{m \times n}$

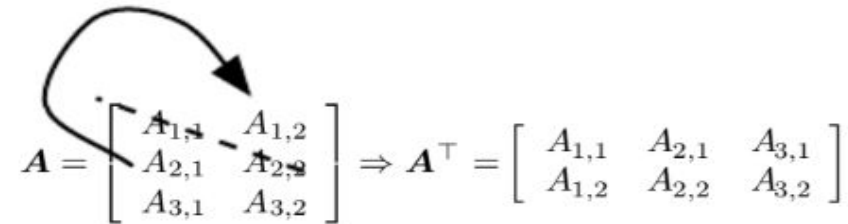
For example, $A_{1,1}$ is the upper left entry of A and $A_{m,n}$ is the bottom right entry. $A_{i,:}$ denotes the horizontal cross section of A with vertical coordinate i . This is known as the i -th row of A . Likewise, $A_{:,i}$ is the i -th column of A .

Tensors: In some cases we will need an array with more than two axes. In the general case, an array of numbers arranged on a regular grid with a variable number of axes is known as a tensor.

Transpose: The transpose of a matrix is the mirror image of the matrix across a diagonal line, called the main diagonal, running down and to the right, starting from its upper left corner. $\mathbf{x} = [x_1, x_2, x_3]^\top$

Vectors can be thought of as matrices that contain only one column. The transpose of a vector is therefore a matrix with only one row. $\mathbf{a} = \mathbf{a}^\top$

$[x_1, x_2, x_3]$. A scalar can be thought of as a matrix with only a single entry. From this, we can see that a scalar is its own transpose



$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} \Rightarrow \mathbf{A}^\top = \begin{bmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \end{bmatrix}$$

Figure 2.1: The transpose of the matrix can be thought of as a mirror image across the main diagonal.

Multiplying Matrices and Vectors

If \mathbf{A} is of shape $m \times n$ and \mathbf{B} is of shape $n \times p$, then \mathbf{C} is of shape $m \times p$. We can write the matrix product just by placing two or more matrices together, e.g.

$$\mathbf{C} = \mathbf{AB}. \quad (2.4)$$

Such operation exists is called as element-wise product or Hadamard product, and is denoted as $\mathbf{A} \odot \mathbf{B}$.

The dot product between two vectors \mathbf{x} and \mathbf{y} of the same dimensionality is the matrix product $\mathbf{x}^\top \mathbf{y}$.

We can think of the matrix product $\mathbf{C} = \mathbf{AB}$ as computing $C_{i,j}$ as the dot product between row i of \mathbf{A} and column j of \mathbf{B} .

Properties of matrix multiplication

Distributive: $A(B + C) = AB + AC$.

Associative: $A(BC) = (AB)C$.

Matrix multiplication is not commutative (the condition $AB = BA$ does not always hold), unlike scalar multiplication.

However, the dot product between two vectors is commutative: $\mathbf{x}^\top \mathbf{y} = \left(\mathbf{x}^\top \mathbf{y}\right)^\top = \mathbf{y}^\top \mathbf{x}$.

The transpose of a matrix product has a simple form: $(AB)^\top = B^\top A^\top$.

Linear equations: $A\mathbf{x} = \mathbf{b}$ where $A \in \mathbb{R}^{m \times n}$ is a known matrix, $\mathbf{b} \in \mathbb{R}^m$ is a known vector, and $\mathbf{x} \in \mathbb{R}^n$ is a vector of unknown variables we would like to solve for. Each element x_i of \mathbf{x} is one of these unknown variables.

Identity and Inverse Matrices

An identity matrix is a matrix that does not change any vector when we multiply that vector by that matrix. We denote the identity matrix that preserves n -dimensional vectors as \mathbf{I}_n . Formally, $\mathbf{I}_n \in \mathbb{R}^{n \times n}$, and

$$\forall \mathbf{x} \in \mathbb{R}^n, \mathbf{I}_n \mathbf{x} = \mathbf{x}.$$

The matrix inverse of \mathbf{A} is denoted as \mathbf{A}^{-1} and it is defined as the matrix such that

$$\mathbf{A}^{-1} \mathbf{A} = \mathbf{I}_n$$

Linear Dependence and Span

Linear dependence refers to a situation where one vector in a set of vectors can be represented as a combination of the others. For example, if you have two vectors:

$$\mathbf{v}_1 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$$

Here, \mathbf{v}_2 is linearly dependent on \mathbf{v}_1 because you can express \mathbf{v}_2 as $2 \cdot \mathbf{v}_1$.

The span of a set of vectors is the set of all possible linear combinations that can be formed using those vectors. For example, if we have two vectors in \mathbb{R}^2 :

$$\mathbf{v}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The span of $\{\mathbf{v}_1, \mathbf{v}_2\}$ is all of \mathbb{R}^2 because any point in the plane can be reached by scaling \mathbf{v}_1 and \mathbf{v}_2 appropriately (for instance, $2\mathbf{v}_1 + 3\mathbf{v}_2$ gives $\begin{pmatrix} 2 \\ 3 \end{pmatrix}$).

Norms

Vector Norms are defined as a set of functions that take a vector as an input and output a positive value against it. This is called the magnitude of a vector.

Scaler loss value (positive) - which is the average of the sum of the difference between the predicted values and the ground truth values squared. This scaler loss value is nothing but an output of a norm function. The way we compute the loss is shown below:

$$\left(\begin{array}{c} \text{Predicted} \\ \text{Vector} \end{array} \begin{bmatrix} 3 \\ 4 \\ 1 \\ 2 \end{bmatrix} - \begin{array}{c} \text{True} \\ \text{Vector} \end{array} \begin{bmatrix} 2 \\ 4 \\ 3 \\ 5 \end{bmatrix} \right)^2 = \begin{array}{c} \text{Loss} \\ \text{Vector} \end{array} \begin{bmatrix} 1 \\ 0 \\ 4 \\ 9 \end{bmatrix} \rightarrow \mathbf{14}$$

Loss
(or Norm)

The Standard Norm Equation — P-norm

For different values of the parameter p (p should be a real number greater than or equal to 1), we obtain a different norm function. The generalized equation, however, is shown below:

$$\underbrace{\|x\|_p}_{p\text{-Norm}} = \left(\sum_i^n |x_i|^p \right)^{\frac{1}{p}} = (|x_1|^p + |x_2|^p + \dots + |x_n|^p)^{\frac{1}{p}}$$

L0 norm

Here $P = 0$ that is from the equation, $1/p = 1/0$ is not defined. To handle this, the standard way of defining the L0 norm is to count the number of non-zero elements in the given vector. The image below shows the output of the L-0 norm function for the given vector:



L1 norm

Substituting $p=1$ in the standard equation of p -norm, we get the following:

$$\underbrace{\|x\|_1}_{L1\ Norm} = \left(\sum_i^n |x_i| \right) = (|x_1| + |x_2| + \cdots + |x_n|)$$

- When used to compute the loss, the L1 norm is also referred to as the Mean Absolute Error.
- L1 norm varies linearly for all locations, whether far or near the origin.

The image below shows the output of the L1 norm function for the given vector:



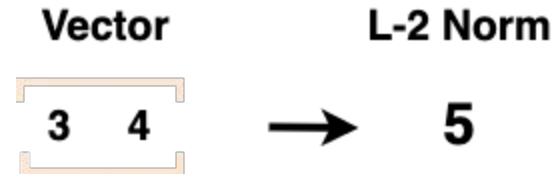
L2 Norm

Of all norm functions, the most common and important is the L2 Norm. Substituting $p=2$ in the standard equation of p -norm, which we discussed above, we get the following equation for the L2 Norm:

$$\underbrace{\|x\|_2}_{L-2 \text{ Norm}} = \left(\sum_i^n |x_i|^2 \right)^{\frac{1}{2}} = \sqrt{|x_1|^2 + |x_2|^2 + \dots + |x_n|^2}$$

- The above equation is often referred to as the root mean squared error when used to compute the error.
- L2 norm measures the distance from the origin, also known as Euclidean distance.

The image below shows the output of the L2 norm function for the given vector:



The Squared L2 Norm is also referred to as the Mean Squared Error.

Max Norm (or L- ∞ Norm)

As infinity is an abstract concept in Mathematics, we can't just substitute $p=\infty$ in the standard p-norm equation. However, we can study the function's behavior as p approaches infinity using limits. A simple derivation for the equation of Max-norm can be found [here](#).

$$\underbrace{\|x\|_{\infty}}_{\text{Max Norm}} = \max_i |x_i|$$

Max norm returns the absolute value of the largest magnitude element. The image below shows the output of the Max norm function for the given vector:



Eigendecomposition

- Integers can be decomposed into prime factors, similarly also decompose matrices in ways that show us information about their functional properties that is not obvious from the representation of the matrix as an array of elements.
- Eigendecomposition, in which we decompose a matrix into a set of eigenvectors and eigenvalues.
- An **eigenvector** of a square matrix \mathbf{A} is a nonzero vector \mathbf{v} such that multiplication by \mathbf{A} alters only the scale of \mathbf{v} : $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$
- The scalar λ is known as the **eigenvalue** corresponding to this eigenvector.

Singular Value Decomposition

The singular value decomposition (SVD) provides another way to factorize a matrix, into singular vectors and singular values.

Every real matrix has a singular value decomposition, but the same is not true of the eigenvalue decomposition. For example, if a matrix is not square, the eigendecomposition is not defined, and we must use a singular value decomposition instead.

Recall that the eigendecomposition involves analyzing a matrix A to discover

a matrix V of eigenvectors and a vector of eigenvalues λ such that we can rewrite A as

$$\mathbf{A} = \mathbf{V} \text{diag}(\lambda) \mathbf{V}^{-1}$$

The singular value decomposition is similar, except this time we will write A as a product of three matrices:

$$\mathbf{A} = \mathbf{U} \mathbf{D} \mathbf{V}^{-1}$$

The elements along the diagonal of D are known as the singular values of the matrix A . The columns of U are known as the left-singular vectors. The columns of V are known as the right-singular vectors.

$$\begin{array}{c}
 \begin{array}{|c|} \hline \text{Matrix} \\ \hline \end{array} \\
 \mathbf{M} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^* \\
 \begin{array}{ccc} m \times n & m \times m & m \times n \quad n \times n \end{array}
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{|c|} \hline \text{Matrix} \\ \hline \end{array} \\
 \mathbf{U} \mathbf{U}^* = \mathbf{I}_m \\
 \begin{array}{ccc} m \times m & m \times m & m \times m \end{array}
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{|c|} \hline \text{Matrix} \\ \hline \end{array} \\
 \mathbf{V} \mathbf{V}^* = \mathbf{I}_n \\
 \begin{array}{ccc} n \times n & n \times n & n \times n \end{array}
 \end{array}$$

Chapter - Numerical Computation

Overflow and Underflow

Rounding error is problematic, especially when it compounds across many operations, and can cause algorithms that work in theory to fail in practice if they are not designed to minimize the accumulation of rounding error.

Underflow occurs when numbers near zero are rounded to zero. Many functions behave qualitatively differently when their argument is zero rather than a small positive number.

For example, we usually want to avoid division by zero or taking the logarithm of zero

Overflow occurs when numbers with large magnitude are approximated as ∞ or $-\infty$. Further arithmetic will usually change these infinite values into not-a-number values.

Softmax function

The softmax function is often used to predict the probabilities associated with a multinoulli distribution. The softmax function is defined to be

$$\text{softmax}(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)}$$

Consider what happens when all of the x_i are equal to some constant c . Analytically, we can see that all of the outputs should be equal to $1/n$. Numerically, this may not occur when c has large magnitude. If c is very negative, then $\exp(c)$ will underflow. When c is very large and positive, $\exp(c)$ will overflow, again resulting in the expression as a whole being undefined.

<https://www.youtube.com/watch?v=8ah-qhvaQqU>

Poor Conditioning

Conditioning refers to how rapidly a function changes with respect to small changes in its inputs.

Functions that change rapidly when their inputs are perturbed slightly can be problematic for scientific computation because rounding errors in the inputs can result in large changes in the output.

Consider the function $f(x) = A^{-1}x$. When $A \in \mathbb{R}^{n \times n}$ has an eigenvalue decomposition, its condition number is

$$\max_{i,j} \left| \frac{\lambda_i}{\lambda_j} \right|$$

This is the ratio of the magnitude of the largest and smallest eigenvalue. When this number is large, matrix inversion is particularly sensitive to error in the input. This sensitivity is an intrinsic property of the matrix itself, not the result of rounding error during matrix inversion. Poorly conditioned matrices amplify pre-existing errors when we multiply by the true matrix inverse. In practice, the error will be compounded further by numerical errors in the inversion process itself.

Gradient-Based Optimization

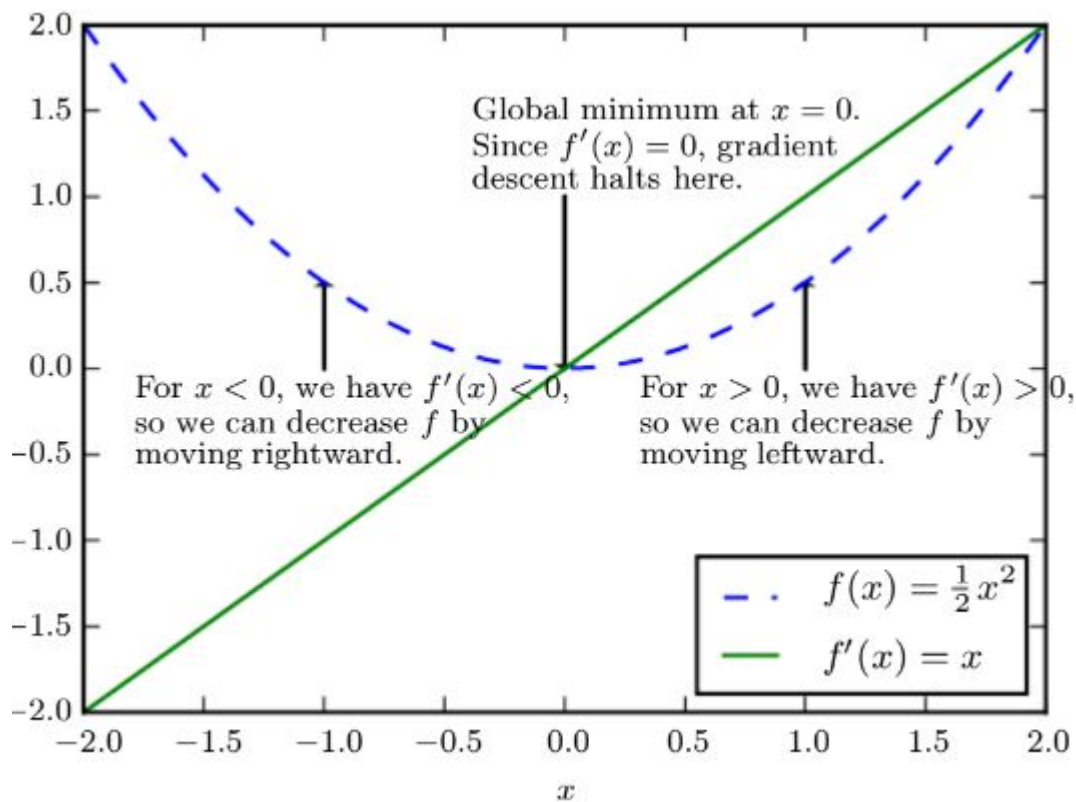
Optimization refers to the task of either minimizing or maximizing some function $f(x)$ by altering x .

We usually phrase most optimization problems in terms of minimizing $f(x)$. Maximization may be accomplished via a minimization algorithm by minimizing $-f(x)$. The function we want to minimize or maximize is called the **objective function or criterion**. When we are minimizing it, we may also call it the cost function, loss function, or error function.

We often denote the value that minimizes or maximizes a function with a superscript $*$. For example, we might say $x^* = \arg \min f(x)$.

<https://youtu.be/gzrQvzYEvYc?si=hFWCxxw6rvTawuzmC> (must watch)

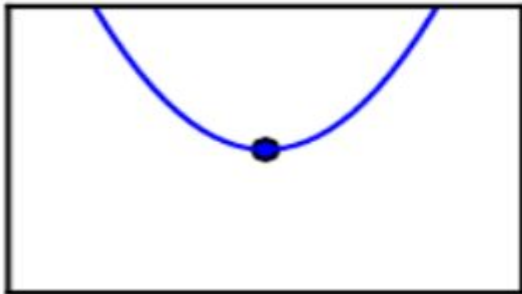
<https://youtu.be/D4zMKh3krPc?si=hl7D2UpFnUYGa1Vz> (detailed video)



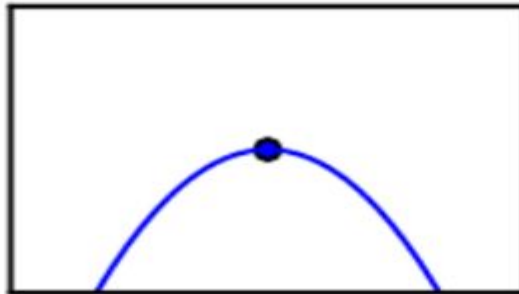
Local Minima, Maxima and Saddle point

When $f'(x) = 0$, the derivative provides no information about which direction to move. Points where $f'(x) = 0$ are known as **critical points** or **stationary points**. A **local minimum** is a point where $f(x)$ is lower than at all neighboring points, so it is no longer possible to decrease $f(x)$ by making infinitesimal steps. A **local maxima** is a point where $f(x)$ is higher than at all neighboring points, so it is not possible to increase $f(x)$ by making infinitesimal steps. Some critical points are neither maxima nor minima. These are known as **saddle points**.

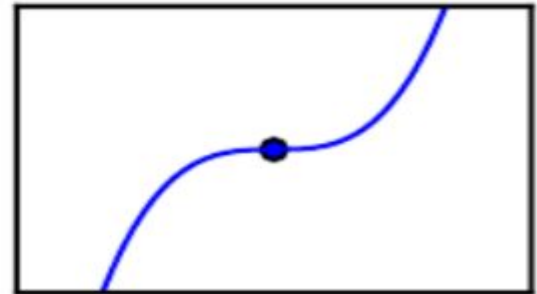
Minimum



Maximum



Saddle point



Constrained Optimization

- Sometimes we wish not only to maximize or minimize a function $f(x)$ over all possible values of x . Instead we may wish to find the maximal or minimal value of $f(x)$ for values of x in some set S . This is known as constrained optimization.

Example:

- Designing the fastest vehicle with a constraint on a fuel efficiency.
- Calculate weight of chair, weight must be positive

All constraints can be converted to two types of constraints

- Equality - Minimize $f(x_1, x_2, x_3)$ subject to $x_1 + x_2 + x_3 = 1$
- Inequality - Minimize $f(x_1, x_2, x_3)$ subject to $x_1 + x_2 + x_3 < 1$

<https://www.youtube.com/watch?v=Dn1vmANCvvs>

Canonical form of constraint optimization

Points x that lie within the set S are called **feasible points** in constrained optimization terminology.

The Karush–Kuhn–Tucker (KKT) approach provides a very general solution to constrained optimization. With the KKT approach, we introduce a new function called the **generalized Lagrangian** or generalized Lagrange function.

To define the Lagrangian, we first need to describe S in terms of equations and inequalities. We want a description of S in terms of m functions $g(i)$ and n functions $h(j)$ so that

$$S = \{x \mid \forall i, g(i)(x) = 0 \text{ and } \forall j, h(j)(x) \leq 0\}.$$

The equations involving $g(i)$ are called the equality constraints and the inequalities involving $h(j)$ are called inequality constraints.

Example, $x_1 + x_2 + x_3 - 1 = 0 \rightarrow$ equality constraint

$x_1 + x_2 + x_3 - 1 < 0 \rightarrow$ inequality constraint

Generalized Lagrange Function

The constrained optimization problem requires us to minimize the function while ensuring that the point discovered belongs to the feasible set.

Generalized Lagrange Function:
$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = f(\mathbf{x}) + \sum_i \lambda_i g^i(\mathbf{x}) + \sum_j \alpha_j h^{(j)}(\mathbf{x})$$

Unit 01

Ends...