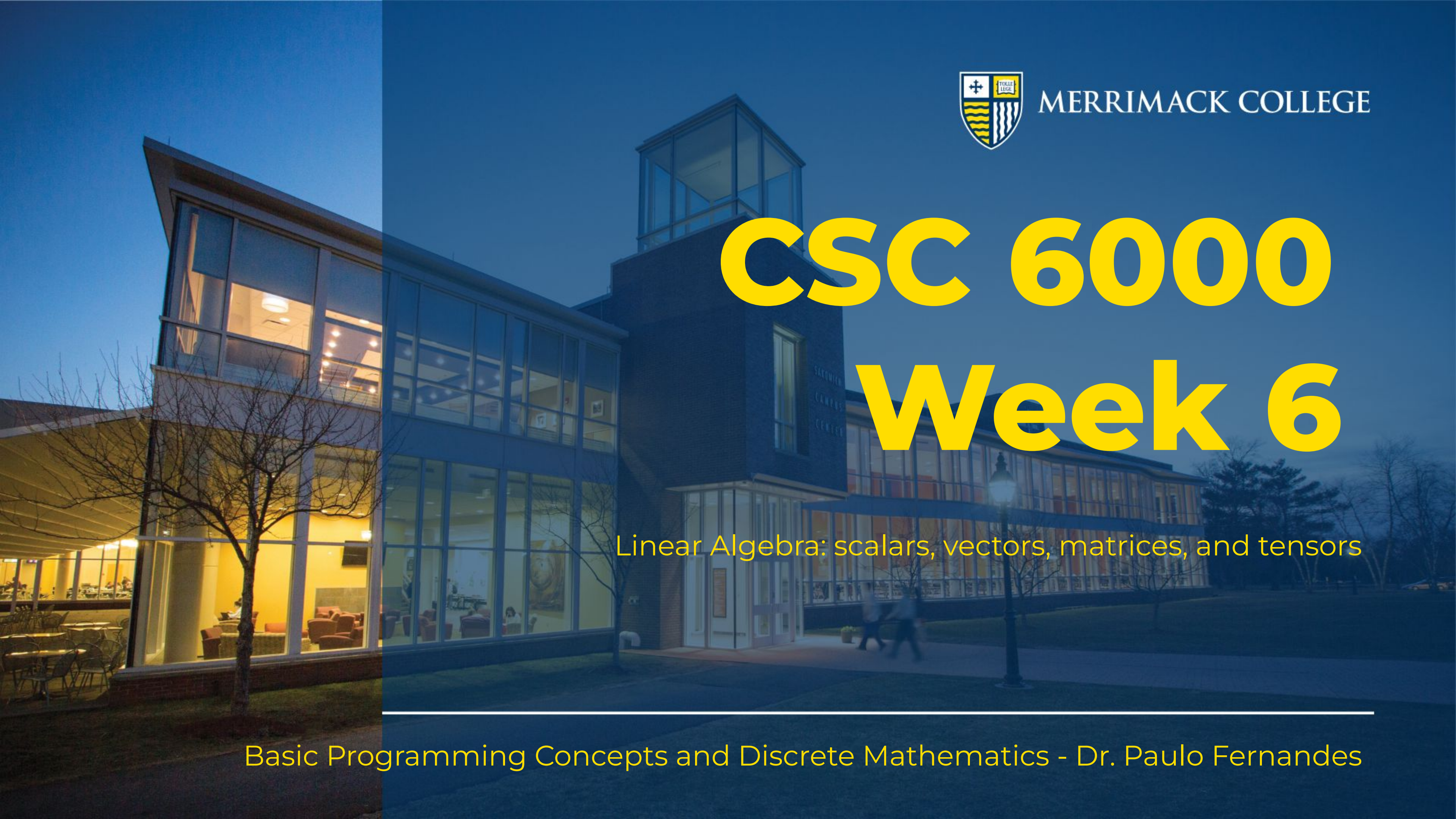


MERRIMACK COLLEGE

CSC 6000 Week 6

Linear Algebra: scalars, vectors, matrices, and tensors

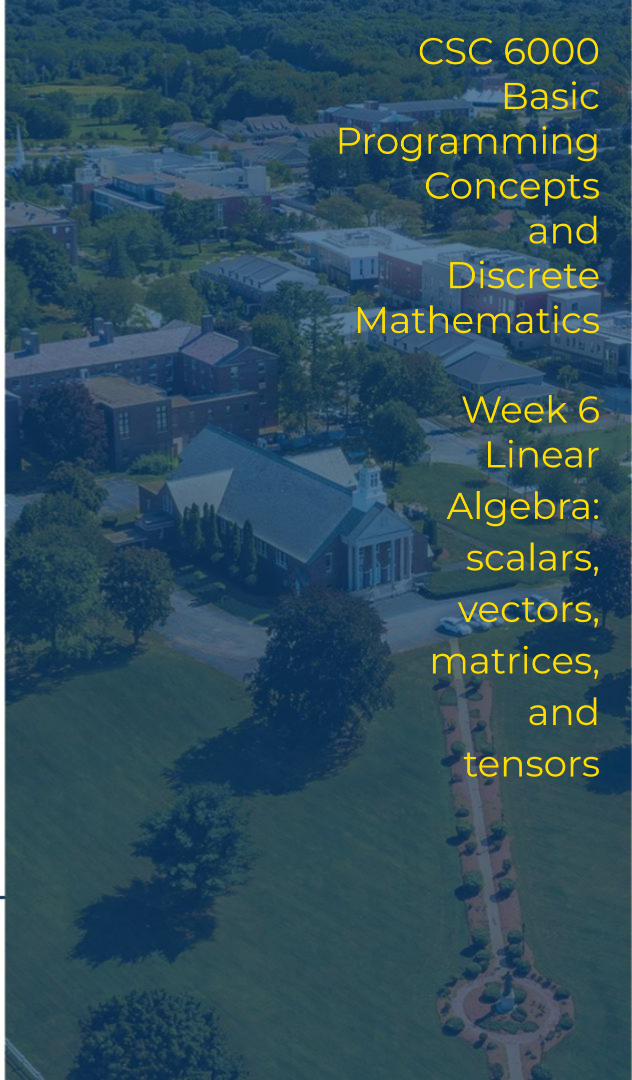
Basic Programming Concepts and Discrete Mathematics - Dr. Paulo Fernandes

Presentation Agenda

Week 6

- Linear Algebra
    a. Scalars and Vectors
    b. Matrices and Tensors
- Arithmetic Operations
    a. Scaling
    b. Transposition
    c. Addition and Subtraction
    d. Multiplication
    e. Inverse
- This Week's tasks

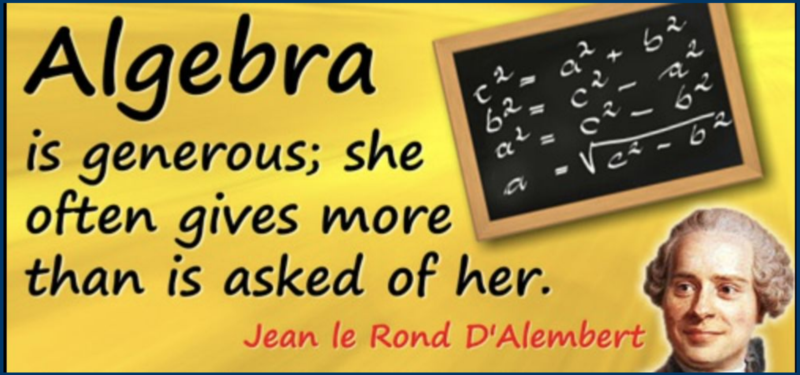
MERRIMACK COLLEGE



CSC 6000 Basic Programming Concepts and Discrete Mathematics

Week 6 Linear Algebra: scalars, vectors, matrices, and tensors

Linear Algebra



If you put numbers in line and you do algebraic operations with them, you are doing linear algebra!

It may sound funny, but that is pretty much what linear algebra is.

The main elements of linear algebra are the ways you can put elements in line:

- **scalars and vectors**,
- matrices and tensors.

What is Algebra?

Persian mathematician **Muḥammad ibn Mūsā al-Khwārizmī** is sometimes mentioned as the father of algebra. Algebra existed before him, but not with this name, nor with the basic rules.

The word Algebra (from the Arabic word الجبر (al-jabr)) means ***reunion of broken parts***.

Basically, it means an equation (the broken parts are the two sides of the equation) with variables. To find the proper value of the variables is to mend the broken parts.

While Arithmetic deals only with numbers, Algebra also deals with variables, thus ***solving equations***.

Wikipedia: Algebra.

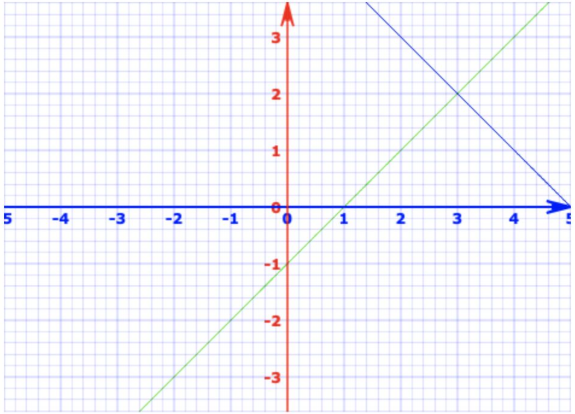
What is Linear Algebra?

If you line up the equations, as in a linear equation system, you are doing a specific kind of algebra that tries to find out the solution of several variables by correlated equations.

For example, if we have two variables ***x*** and ***y*** and we know:

- $x + y = 5$
- $x - y = 1$

It is possible to discover the values of both ***x*** and ***y***.

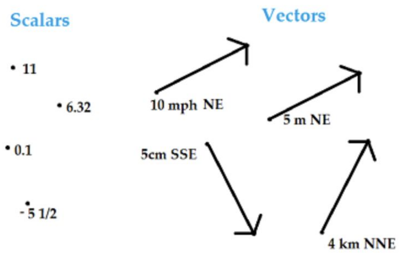


Can you identify the values of ***x*** and ***y*** that satisfy each equation alone, and the values that satisfy both equations?

MERRIMACK COLLEGE

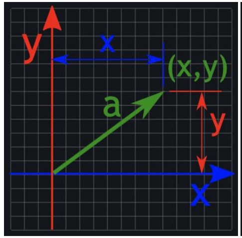
Wikipedia: Linear Algebra.

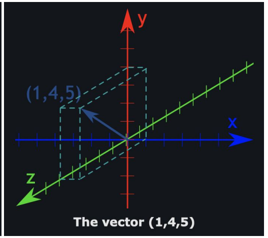
Scalars and Vectors



The origins of Linear Algebra were with systems of linear equations and matrices. However, modern linear algebra (what we call modern dates back to the XIX Century) is probably better defined with the idea of Vector Spaces.

The numbers we all know, standing alone, are what is called a ***scalar***. It is used to express a quantity that is scaling a certain unit: the number one.





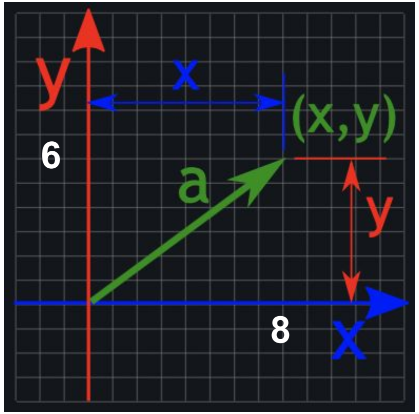
As such, the scalar 4 means a quantity that is four times whatever unit we are calling 1.

A ***vector*** is a group of ***d*** scalars disposed in an orderly fashion, as [8,6] or [1,4,5]. Conversely, from a Computer Science point of view, an array of numbers.

Text: Scalars and Vectors.

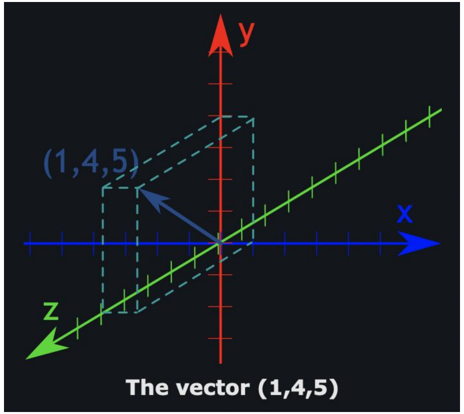
Vectors and Cartesian Display

A vector with ***d*** elements is sometimes referred as a ***d***-dimensional because it can be represented in a ***d***-dimensional space.



For example, the vector **[8, 6]** can be drawn in a two-dimensional space (a Cartesian plane: ***d = 2***) as the green vector on the left-hand side.

Another exemple, on the right-hand side, is the vector **[1, 4, 5]** that can be drawn in a three-dimensional space (a Cartesian space: ***d = 3***).



Despite living in a 3-dimensional world, it is possible to handle vectors with ***d > 3***.

Displaying vectors graphically is not really required.

Vectors Elements - Indexing

When we are using a small vector, we may often refer to each of its elements with different names.

For vectors with just two elements, usually thinking about a Cartesian display, it is common to refer to the elements as ***x*** and ***y***. Similarly, for vectors with three elements the letters ***x***, ***y***, and ***z*** are often used.

However, when the number of elements grows, or just for convenience, it is normal to name the vectors and refer to its elements by indices.

Usually mathematical notation uses a capital letter for the vector and lower case letters with indices from ***1*** to ***d*** to the elements.

Computationally, we often change the indices to go from ***0*** to ***d-1***.

$V = (x, y)$

$V = (x, y, z)$

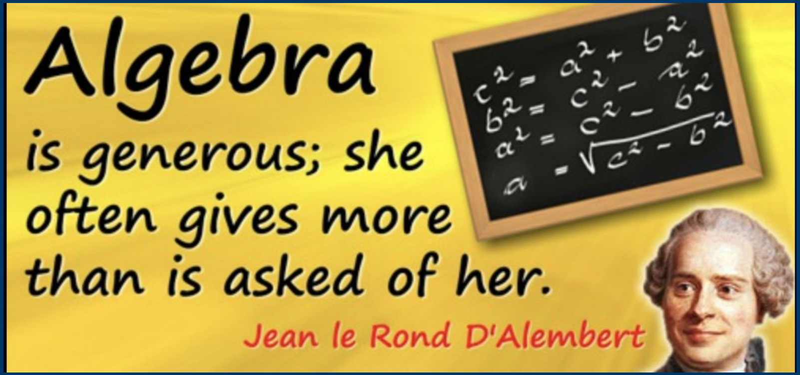
$V = [v_1\ v_2\ v_3\ v_4]$

$V = [v_0\ v_1\ v_2\ v_3]$

MERRIMACK COLLEGE

In Python: ***V = [1, 4, 5]*** and ***V[0] = 1***, ***V[1] = 4***, ***V[2] = 5***

Linear Algebra



If you put numbers in line and you do algebraic operations with them, you are doing linear algebra!

It may sound funny, but that is pretty much what linear algebra is.

The main elements of linear algebra are the ways you can put elements in line:

- scalars and vectors,
- **matrices and tensors**.

Matrices

Here things might get messy… While we may refer to vectors as being depicted in ***d***-dimensional spaces, the fact remains that a vector is always a uni-dimensional organization of scalars.

In fact, a scalar has 0 dimensions, as it is a number alone.

45 56 2 1 88

A vector has 1 dimension as the elements are disposed in a single list of scalars.

[8, 6] [7, 0, -3, 91, 14] [1, 4, 5] [8, 6, 7, 5, 3, 0, 9]

Logically, it is possible to imagine a structure with 2 dimensions. We call these matrices.

$$\begin{bmatrix} 0 & 2 & 0 \\ 0 & 5 & 7 \\ 1 & 2 & 9 \end{bmatrix} \quad \begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 18 & 22 \end{bmatrix} \quad \begin{bmatrix} 12 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 5 & 2 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 5 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 0 & -2 & -1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 \end{bmatrix}$$

Wikipedia: Matrices.

Matrices Sizes

The matrix **elements** are indexed by row (***i***) and column (***j***), e.g., using CS style where the first row (***i = 0***) and second column (***j = 1***), an element of matrix ***M*** is referred by $m_{0,1}$.

While a vector is defined by the number of elements ***d***, the size of a matrix is defined by the number of rows ***r*** and the number of columns ***c***. The matrix size is often referred as ***r****x****c***.

- For the matrix ***A*** example, ***r = 3*** and ***c = 2*** which allows us to refer to the matrix ***A*** as having dimensions ***3****x****2***.

$$A = \begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix}$$

In fact, vectors can also be seen as matrices, which leads to two kind of vectors in a matrix representation:

- a vector disposed in a matrix row
  - like the vector ***R*** that corresponds to a matrix ***1****x****3***,
- or a vector disposed in a matrix column
  - like the vector ***C*** that corresponds to a matrix ***2****x****1***.

$$R = \begin{bmatrix} 1 & 4 & 5 \end{bmatrix}$$

$$C = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

In Python: ***A = [[11, 14], [7, 18], [0, 22]]*** and ***A[0][0] = 11***, ***A[0][1] = 14***, ***A[1][0] = 7***

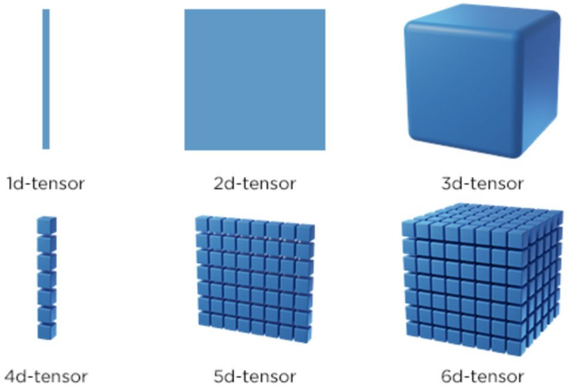
Matrix Types

The matrix **diagonal** are the elements with the same index for row (***i***) and column (***j***), i.e., ***i = j***.

- Row matrix - a vector disposed in a matrix row; $\begin{bmatrix} 4 & -8 & 5 \end{bmatrix}$
- Column matrix - a vector disposed in a matrix column; $\begin{bmatrix} 4 \\ -6 \\ 7 \end{bmatrix}$
- Null Matrices - all elements equal to zero; $\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$
- Square Matrices - the same number of rows and columns; $\begin{bmatrix} \boxed{4} & -8 & 5 \\ -6 & \boxed{3} & 1 \\ 7 & 2 & \boxed{2} \end{bmatrix}$
- Diagonal Matrices - all elements equal to zero, except the diagonal elements; $\begin{bmatrix} 4 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 2 \end{bmatrix}$
- Identity Matrices - all elements equal to zero, except the diagonal elements are all equal to 1; $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Upper Matrices - all elements below the diagonal are equal to zero; $\begin{bmatrix} 4 & -8 & 5 \\ 0 & 3 & 1 \\ 0 & 0 & 2 \end{bmatrix}$
- Lower Matrices - all elements above the diagonal are equal to zero. $\begin{bmatrix} 4 & 0 & 0 \\ -6 & 3 & 0 \\ 7 & 2 & 2 \end{bmatrix}$

Text: [Matrix Types](#).

Matrices and Tensors



Scalars have 0 dimensions:

45 **56** **2** **1** **88**

Vectors have 1 dimension:

[8, 6] **[7, 0, -3, 91, 14]** **[1, 4, 5]** **[8, 6, 7, 5, 3, 0, 9]**

Matrices have 2 dimensions:

$$\begin{bmatrix} 4 & -8 & 5 & 11 & 14 \\ -6 & 3 & -1 & 3 & 1 \\ 7 & 2 & 0 & 2 & 22 \\ 1 & 4 & 5 & -9 & 10 \\ 56 & 24 & 23 & 31 & 0 \end{bmatrix} \quad \begin{bmatrix} -7 & 3 & 0 \\ 2 & -5 & 74 \end{bmatrix} \quad \begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix}$$

Tensors have 3 or more dimensions:

$$\left[\begin{array}{cc|cc|cc} -11 & 5 & 4 & 0 & 2 & 0 \\ 7 & -13 & 0 & 4 & 0 & 2 \\ \hline 8 & 0 & -17 & 5 & 4 & 0 \\ 0 & 8 & 7 & -19 & 4 & 0 \\ \hline 1 & 0 & 6 & 0 & -12 & 5 \\ 0 & 1 & 0 & 6 & 7 & -14 \end{array}\right]$$

This tensor has 4 dimensions: 3x3x2x2

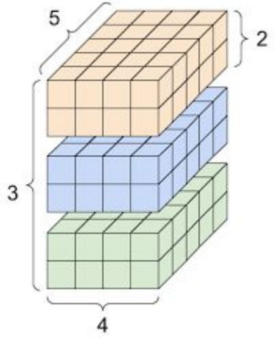
MERRIMACK COLLEGE

Wikipedia: Tensor.

Tensors

This is an example of a 3x5x2x4, which is four dimensional:

- 3 in the first dimension (3 blocks - line of matrices);
- 5 in the second dimension (5 in depth - row of matrices);
- 2 in the third dimension (2 rows in the inner matrices);
- 4 in the fourth dimension (4 columns in the inner matrices).



$$\left[\begin{array}{cccc|cccc|cccc|cccc|cccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \\
5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 & 0 & 9 & 9 & 0 & 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \\
\hline
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
\hline
5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 & 0 & 9 & 9 & 0 & 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \\
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}\right]$$

The physical display is merely a visual help.

Arithmetic Operations

Anyone who understands algebraic notation, reads at a glance in an equation results reached arithmetically only with great labour and pains.

Antoine-Augustin Cournot

Algebra is arithmetic plus variables, thus we have to first establish the arithmetic operations over matrices, as matrices can be easily generalized into vectors and tensors.

The arithmetic operations of linear algebra we will see are:

- Scaling
- Transposition
- Addition and Subtraction
- Multiplication
- Inverse

Scaling Matrices

The operation of scaling matrices (or vectors and tensor) is simply made by multiplying all elements of the matrix by a scalar.

Actually, the name "scalar" for a single number comes from this basic operation.

For example:

$$\begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix} \times 10 = \begin{bmatrix} 110 & 140 \\ 70 & 180 \\ 0 & 220 \end{bmatrix} \quad \begin{bmatrix} 8 \\ 6 \end{bmatrix} \times 0.5 = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

```python
def scale(A, s):
    As = []
    for i in range(len(A)):
        As.append([])
        for j in range(len(A[0])):
            As[-1].append(A[i][j] * s)
    return As

A = [[11,14], [7,18], [0,22]]
print(scale(A, 10))
B = [[8], [6]]
print(scale(B, 0.5))
```

```
[[110, 140], [70, 180], [0, 220]]
[[4.0], [3.0]]
```

```python
A = [[11,14], [7,18], [0,22]]
print(A*2)
```

```
[[11, 14], [7, 18], [0, 22], [11, 14], [7, 18], [0, 22]]
```

Be aware, when you multiply a Python list you are not scaling, you are **replicating** the structure.

Transposition of Matrices

Given a matrix ***A*** of dimensions ***N****x****M*** defined as:

$$A = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,m-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \cdots & a_{n-1,m-1} \end{bmatrix}$$

Can be transposed into a matrix $\boldsymbol{A^T}$ of dimensions ***M****x****N*** defined with all element $\boldsymbol{a_{i,j}}$ repositioned as $\boldsymbol{a_{j,i}}$.

For example:

$$B = \begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix} \qquad C = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

$$B^T = \begin{bmatrix} 11 & 7 & 0 \\ 14 & 18 & 22 \end{bmatrix} \qquad C^T = \begin{bmatrix} 8 & 6 \end{bmatrix}$$

```python
def transpose(A):
    AT = []        # the transposed of A
    for j in range(len(A[0])):
        AT.append([]) # the j row of AT
        for i in range(len(A)):
            AT[-1].append(A[i][j])
    return AT

B = [[11,14], [7,18], [0,22]]
print(B)
print(transpose(B))

C = [[8], [6]]
print(C)
print(transpose(C))
```

```
[[11, 14], [7, 18], [0, 22]]
[[11, 7, 0], [14, 18, 22]]
[[8], [6]]
[[8, 6]]
```


Text: Transpose a matrix in Single line in Python.

Adding and Subtracting Matrices

Given a matrix ***A*** of dimensions ***N***x***M*** and a matrix ***B*** of the same dimensions, the matrix ***C***, also with the same dimensions, has its elements defined by:

- $c_{i,j} = a_{i,j} + b_{i,j}$

Similarly the subtraction of A minus B resulting in C has the elements of C defined by:

- $c_{i,j} = a_{i,j} - b_{i,j}$

For example:

$$\begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix} + \begin{bmatrix} 3 & 5 \\ 6 & -9 \\ 8 & -15 \end{bmatrix} = \begin{bmatrix} 14 & 19 \\ 13 & 9 \\ 8 & 7 \end{bmatrix}$$

$$\begin{bmatrix} 11 & 14 \\ 7 & 18 \\ 0 & 22 \end{bmatrix} - \begin{bmatrix} 3 & 5 \\ 6 & -9 \\ 8 & -15 \end{bmatrix} = \begin{bmatrix} 8 & 9 \\ 1 & 27 \\ -8 & 37 \end{bmatrix}$$

```python
def add(A, B, op="add"):
    C = []
    for i in range(len(A)):
        C.append([])
        for j in range(len(A[0])):
            if (op == "add"):
                C[-1].append(A[i][j] + B[i][j])
            else:
                C[-1].append(A[i][j] - B[i][j])
    return C

A = [[11, 14], [7, 18], [0, 22]]
B = [[3, 5], [6, -9], [8, -15]]
print("The sum is:", add(A,B))
print("The subtraction is:", add(A,B, "sub"))
```

```
The sum is: [[14, 19], [13, 9], [8, 7]]
The subtraction is: [[8, 9], [1, 27], [-8, 37]]
```

Note the optional parameter ***op*** that by default does a sum, otherwise it does a subtraction.

The null matrix is the neutral of sum and subtraction.

Matrix Multiplication (or product)

Given:

- a matrix ***A*** of dimensions ***N****x****P*** and
- a matrix ***B*** of dimensions ***P****x****M***,
- the matrix ***C*** of dimensions ***N****x****M*** is the multiplication of ***A*** and ***B*** if, and only if, its elements are defined by:
  - $c_{i,j} = a_{i,k} * b_{k,j}$ summed for all values of ***k*** from ***0*** to ***P-1***
- The order matters: ***A*** times ***B*** is not equal to ***B*** times ***A***

For example:

$$\begin{bmatrix} 10 & 20 \\ 30 & 40 \\ 50 & 60 \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} = \begin{bmatrix} 110 & 140 & 170 & 200 \\ 230 & 300 & 370 & 440 \\ 350 & 460 & 570 & 680 \end{bmatrix}$$

$$\begin{bmatrix} 10 \times 1 + 20 \times 5 & 10 \times 2 + 20 \times 6 & 10 \times 3 + 20 \times 7 & 10 \times 4 + 20 \times 8 \\ 30 \times 1 + 40 \times 5 & 30 \times 2 + 40 \times 6 & 30 \times 3 + 40 \times 7 & 30 \times 4 + 40 \times 8 \\ 50 \times 1 + 60 \times 5 & 50 \times 2 + 60 \times 6 & 50 \times 3 + 60 \times 7 & 50 \times 4 + 60 \times 8 \end{bmatrix}$$

3*x****2*** times ***2****x****4*** gives ***3****x****4***

$$c_{i,j} = \sum_{k=0}^{P-1} a_{i,k} \times b_{k,j}$$

N*x****P*** times ***P****x****M*** gives ***N****x****M***

The identity matrix is the neutral of the product.

Video: Matrix Multiplication.

Matrix Multiplication

Given:

- a matrix ***A*** of dimensions ***N***x***P*** and
- a matrix ***B*** of dimensions ***P***x***M***,
- the matrix ***C*** of dimensions ***N***x***M*** is the multiplication of ***A*** and ***B*** if, and only if, its elements are defined by:

$$c_{i,j} = \sum_{k=0}^{P-1} a_{i,k} \times b_{k,j}$$

```
1   def mult(A,B):
2       C = []
3       for i in range(len(A)):
4           C.append([])
5           for j in range(len(B[0])):
6               tmp = 0
7               for k in range(len(B)):
8                   tmp += A[i][k] * B[k][j]
9               C[-1].append(tmp)
10      return C
11
12  A = [[10,20], [30,40], [50,60]]
13  B = [[1,2,3,4], [5,6,7,8]]
14  print(mult(A,B))
```

```
[[110, 140, 170, 200], [230, 300, 370, 440], [350, 460, 570, 680]]
```

For example: $\begin{bmatrix} 10 & 20 \\ 30 & 40 \\ 50 & 60 \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} = \begin{bmatrix} 110 & 140 & 170 & 200 \\ 230 & 300 & 370 & 440 \\ 350 & 460 & 570 & 680 \end{bmatrix}$

Each element is the product of a row of ***A*** by a column of ***B***.

Video: Dot Product, multiplying two vectors (lines 6 to 8).

Matrix Inverse

There is no division of matrices, but you can inverse a matrix, just like you invert a scalar.

- e.g.: ***0.2*** is the inverse of ***5***.

We denote the inverse of the matrix $\boldsymbol{A}$ as $\boldsymbol{A}^{-1}$ just like we do with a scalar:

- e.g.: $\boldsymbol{0.2^{-1} = 5}$ and $\boldsymbol{5^{-1} = 0.2}$

The inverse operation is symmetric!

To find the inverse of a matrix is a complex operation. There are several ways to do it, but it is important to know that:

- some matrices do not have an inverse;
- some methods to invert a matrix are not applicable to some matrices that do have an inverse;
- To invert a matrix is as complex as solving a linear equation system. Brilliant mathematicians from both the past and now spend great effort to find ways to efficiently invert very large linear equation systems.

The product of $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$ is an Identity matrix.

Merrimack College

Text: [Inverse of a Matrix].

Matrix Inverse

Linear equation systems as matrices

A linear equation system composed of ***N*** equations using ***N*** variables can be represented as an algebraic expression of matrices and vectors.

For example, the system below has two equations and two variables (x_0 and x_1)

Can you solve this system?
What are the values of x_0 and x_1?

$$\begin{array}{lll} 3x_0 & +2x_1 & = 26 \\ 4x_0 & +5x_1 & = 51 \end{array}$$

Given a column vector of variables ***X***

- $X = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$

Given a square matrices ***A*** with the coefficients of the equations:

- $A = \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix}$

Given a column vector of the resulting constants ***B***:

- $B = \begin{bmatrix} 26 \\ 51 \end{bmatrix}$]

$$A \times X = B$$

Do the matrix multiplication of ***A*** times ***X*** equal to ***B***!

Matrix Inverse

$$A = \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix} \quad X = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} \quad B = \begin{bmatrix} 26 \\ 51 \end{bmatrix}$$

Linear equation systems as matrices

An elegant way to solve this system is:

- Compute the inverse of ***A***, ***A***$^{-1}$;
- Arithmetically multiply both sides of the equation by ***A***$^{-1}$;
- Since ***A*** times ***A***$^{-1}$ is the identity, the product of ***A***$^{-1}$ times ***B*** gives the values of ***X***.

$$\boxed{A \times X = B} \longleftrightarrow \boxed{\begin{array}{lll} 3x_0 & +2x_1 & = 26 \\ 4x_0 & +5x_1 & = 51 \end{array}}$$

$$\boxed{A^{-1} \times A \times X = A^{-1} \times B}$$

$$\boxed{X = A^{-1} \times B} \qquad A^{-1} = \begin{bmatrix} 5/7 & -2/7 \\ -4/7 & 3/7 \end{bmatrix}$$

x_0 = ***4*** and ***x_1*** = ***7***

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 5/7 & -2/7 \\ -4/7 & 3/7 \end{bmatrix} \times \begin{bmatrix} 26 \\ 51 \end{bmatrix} = \begin{bmatrix} 4 \\ 7 \end{bmatrix}$$

Text: Matrix Reshish (compute Inverse)

This Week's tasks

- Post discussion D#6
- Coding Project P#6
- Quiz Q#6

Tasks

- Post in the discussion one application of matrices.
- Coding Project #6, check if two matrices are one the inverse of the other.
- Quiz #6 about this week topics.

Post Discussion - Week 6 - D#6

Post in the discussion one practical application of matrices:

- Go over the Internet and try to find a practical application of matrices;
- There are several, so it should be fairly easy, but...
  - You have to check out your colleagues' posts first and you cannot post one that has already been posted by one of your colleagues!
  - So, you'd better complete it sooner rather than later, as this task will become harder as your colleagues post their own.

Once you've done it, reply to at least one of your colleagues about how unexpected (or not) their application was.

Your task:

- Post your discussion in the message board by this Monday;
- Reply posts of your colleagues in the message board by next Thursday.

MERRIMACK COLLEGE

This task counts towards the Discussions grade.

Sixth Coding Project - Week 6 - P#6

- Write a Python program that receives two matrices.
  - Your program should then verify if those two matrices are each other's inverses
    - You can multiply them to check it.
      - If the matrices cannot be multiplied, then they are not inverses to each other;
      - If their multiplication does not give an Identity matrix, they are not inverses to each other.

Example in the next slide

Your task:

- Go to Canvas, and submit your Python file (.py) within the deadline:
  - The deadline for this assignment is next Thursday.

This assignment counts towards the Projects grade.

Sixth Coding Project - Week 6 - P#6

- For example, if your program receives the following two matrices, it should print "the matrices are inverses to each other":

$$\begin{bmatrix} 2 & 1 & 4 \\ 1 & -2 & 2 \\ -1 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0.2 & -0.4 & -1 \\ 0.2 & -0.4 & 0 \\ 0.1 & 0.3 & 0.5 \end{bmatrix}$$

```
round(number,8)
```

- On the contrary, for these two matrices, it should print "the matrices are not inverses to each other":

$$\begin{bmatrix} 2 & -4 & -1 \\ 2 & -4 & 0 \\ 1 & 3 & 5 \end{bmatrix} \qquad \begin{bmatrix} 2 & -1.7 & 4 \\ 1 & -1.1 & 2 \\ -1 & 1 & 0 \end{bmatrix}$$

```
working with floats,
close to zero is
zero
```

MERRIMACK COLLEGE

You can test with any pair of matrices generated using the Reshish tool.

Sixth Quiz - Week 6 - Q#6

- The sixth quiz in this course covers the topics of Week 6;
- The quiz will be available this Saturday, and it is composed of 10 questions;
- The quiz should be taken on Canvas (Module 6), and it is not timed:
  - You can take as long as you want to answer it;
- The quiz is open book, open notes, and you can even use any language interpreter to answer it;
- Yet, the quiz is evaluated and you are allowed to submit it only once.

Your task:

- Go to Canvas, answer the quiz and submit it within the deadline:
  - The deadline for the quiz is next Thursday.

This quiz counts towards the Quizzes grade.

We are in Week 6 of CSC 6000

- **Post discussion D#6 by Monday, reply to colleagues by next Thursday;**
- **Do quiz Q#6 (available Saturday) by next Thursday;**
- **Develop coding project P#6 by next Thursday.**

Next Week - Basic Probabilities and their Computation