|  |
| --- |
|  |
| SSE 691: Intro to Data Science Project 2 |
|  |

|  |
| --- |
| Jonathan Alfred  9-17-2016 |

Contents

[Introduction 1](#_Toc461879411)

[Linear Algebra 1](#_Toc461879412)

[What is Linear Algebra? 1](#_Toc461879413)

[Linear Algebra in Data Science 1](#_Toc461879414)

[Important Concepts 2](#_Toc461879415)

[Vectors 2](#_Toc461879416)

[Matrices 2](#_Toc461879417)

[NumPy 2](#_Toc461879418)

[Indirect Logs 2](#_Toc461879419)

[References 2](#_Toc461879420)

[Image Credits 2](#_Toc461879421)

[Code 2](#_Toc461879422)

# Introduction

In this project, important concepts of linear algebra were explored. These concepts were applied to data science and important functionalities were both created and tested in python, utilizing lists as vectors and matrices. NumPy was also explored, as linear algebra concepts are already built into this library.

# Linear Algebra

This section gives an overview of linear algebra and how linear algebra applies to data science.

## What is Linear Algebra?

Linear algebra is the branch of mathematics that deals with vector spaces. Vector spaces are collections of vectors. These are essential lines with a magnitude and direction. These can be used to represent forces and motion, but can be used to represent less physical objects, such as a collection of data points representing a person. Vector operations, such vector addition and scalar multiplication (Figure 1), must meet several requirements, called axioms. These requirements are for things like Commutativity of addition, where the order of addition of vectors does not matter.

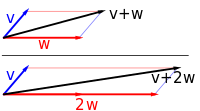


Figure 1 - Vector addition and scalar muliplication

The study of linear algebra first came to be around the late 17th and early 18th centuries. Several of the fundamental topics, such as matrices did not emerge until the 19th century. Linear algebra looks into interactions of vectors within vector space, looking to find common properties among all vector spaces and studying lines, planes and subplanes. Its applications are very diverse, ranging from analytical geometry to social sciences.

## Linear Algebra in Data Science

Linear algebra is a fundamental pillar of data science. Basic concepts, such as matrices and vectors, are used in practically everything. Mathematical modeling makes great use of vectors and matrices. An example of this would be a model of a rocket. Vectors would be used for the motion of the rocket, along with the derivations of its velocity and acceleration. These would be placed in a 3x3 matrix, used within a variety of different equations to produce a predicted path of a rocket along a path. An image of the predicted path can be seen in Figure 2.

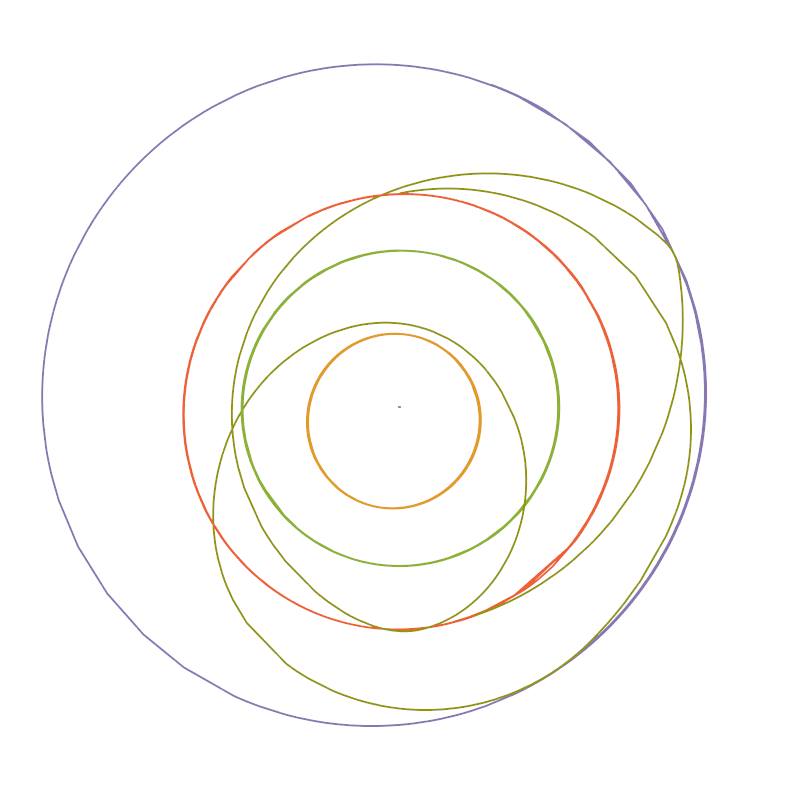


Figure 2 - Mathematica model illustrating the Rich Purnell Manuever see in the Martian. The orbits of Earth, Mercury, Venus, and Mars are all included.

Linear algebra is used in machine learning. Many different matrix operations are used in neural networks. Learning concepts make use of vectors to change the weights within the neural network. Errors seen in the output are corrected using different types of learning iterations to compute a correction to weights within the neural network, making use of vector manipulations and matrix operations.

# Important Concepts

This section goes over the two very key topics used in linear algebra: Vectors and Matrices

## Vectors

Chapter 4 goes over many of the different operations used with vectors. It showed just how the different operations were coded into python. Examples were provided to code these operations for lists, though it was sure to note that lists were not optimal for resources and that these operations were not efficient, but simply for to serve as an introduction to linear algebra techniques.

The operations that were described in the chapter are seen below:

* Vector Addition. Vectors are added component wise and returned as vector of the same dimensions as the added vectors. The two vectors being added together must be of the same dimension, or the addition will fail. Our code does not account for this.

def vector\_add(v,w):

return [v\_i + w\_i for v\_i, w\_i in zip (v,w)]

* Vector Subtraction. This is similar to vector addition, in that the vectors are manipulated component-wise, but are subtracted instead of added. As with addition, the vectors must be of the same dimensions or the subtraction will fail.

def vector\_subtract(v,w):

return [v\_i -w\_i for v\_i, w\_i in zip (v,w)]

* Vector Summation. This is vector addition over more than just two vectors. It returns one single vector with the same dimensions as the added vectors.

def vector\_sum(vectors):

return reduce(vector\_add, vectors)

* Scalar multiplication. A scalar number is multiplied by all components of the vector. Each component is multiplied and returned a vector of the same size as the original vector.

def scalar\_multiply(c, v):

return [c\*v\_i for v\_i in v]

* Vector Mean. This takes a list of vectors, component wise adds them, and then takes the average of the vectors, returning that to the user.

def vector\_mean(vectors):

n = len(vectors)

return scalar\_multiply(1/n, vector\_sum(vectors))

* Dot Product. The dot product is the component wise multiplication of two vectors. After the multiplication, all the components are multiplied together.

def dot(v, w):

return sum(v\_i\*w\_i for v\_i, w\_i in zip(v, w))

* Sum of Squares. Sum of squares is the dot product of a vector with itself. This squares all components of the vector, before summing the values up.

def sum\_of\_squares(v):

return dot(v, v)

* Magnitude. The square root of the sum of squares is the magnitude of a vector. This is the length of the vector as an absolute value.

def magnitude(v):

return math.sqrt(sum\_of\_squares(v))

* Distance. This refers to the distance between the two ends of a vector. There were two methodologies used, both returning the same value.

def squared\_distance(v, w):

return sum\_of\_squares(vector\_subtract(v,w))

def distance\_v1(v, w):

return math.sqrt(squared\_distance(v,w))

def distance\_v2(v, w):

return magnitude(vector\_subtract(v,w))

Each of these functions were run with example code and produced the following output.

Vector Addition: [1, 4] + [3, 2] = [4, 6]

Vector Subtraction: [-1, 3] - [2, -2] = [-3, 5]

Vector Sum:

Vector List -> [[1, 4], [3, 2], [-1, 3], [2, -2], [-3, -4]]

Answer -> [2, 3]

Scalar Multiply: [2, -2] \* 4 = [8, -8]

Vector Mean:

Vector List -> [[1, 4], [3, 2], [-1, 3], [2, -2], [-3, -4]]

Answer -> [0.4, 0.6000000000000001]

Dot Product: [-3, -4] . [1, 4] = -19

Sum of Squares: for [3, 2] -> 13

Magnitude: for [-1, 3] -> 3.1622776601683795

Distance Squared: for [2, -2] and [-3, -4] -> 29

Distance: for [1, 4] and [3, 2]

Version 1 -> 2.8284271247461903

Version 2 -> 2.8284271247461903

## Matrices

# NumPy

# Discussion

# Indirect Logs

# References

"Linear Algebra". *Wikipedia*. N.p., 2016. Web. 17 Sept. 2016.

"Vector Space". *Wikipedia*. N.p., 2016. Web. 17 Sept. 2016.

# Image Credits

# Code