# Euclidean Distance

Manuel Aguilar

May 2018

## 1 Jupyter Notebook Settings

```
In [ ]: # Set ip to '*' to bind on all interfaces (ips) for the public server
        c.NotebookApp.ip = '*'
        c.NotebookApp.password = u'sha1:ed1f40ad4ace:160461b6ca05a48d350811736936c5ff6a387539'
        c.NotebookApp.open_browser = False
        c.NotebookApp.base_url = '/epcc/'
        # It is a good idea to set a known, fixed port for server access
        c.NotebookApp.port = 9999
        c.NotebookApp.iopub_data_rate_limit=10e6
```

## 2 Python Objects

### Location.py

```
In [30]: import math
         import os
         import hashlib
         from pandas import DataFrame

         class Location(object):

             def __init__(self, name, northDeg, northMin, northSec, eastDeg, eastMin, eastSec):
                 """
                 :param name:
                 :param northDeg:
                 :param northMin:
                 :param northSec:
                 :param eastDeg:
                 :param eastMin:
                 :param eastSec:
                 :return: Location Object
                 """
                 self.name = name
                 self.northDeg = northDeg
                 self.northMin = northMin
                 self.northSec = northSec
```

```
        self.eastDeg = eastDeg
        self.eastMin = eastMin
        self.eastSec = eastSec

    def UUID(self,num=31):
        salt=os.urandom(num).hex()
        e=str(salt+self.name).encode("UTF-8")
        return hashlib.sha512(e).hexdigest()

    def latitudeDMS(self):
        return tuple([self.northDeg, self.northMin, self.northSec])

    def longitudeDMS(self):
        return tuple([self.eastDeg, self.eastMin, self.eastSec])

    def DMS(self):
        return tuple([ self.latitudeDMS(), self.longitudeDMS()])

    # Convert DMS -> DD
    # Formula := dd= degree + (min/60) + (second/3600)

    def latitudeDecimalDegree(self):
        f= round(float(float(self.northDeg)+float(self.northMin/60)+float(self.northSec
        return f

    def longitudeDecimalDegree(self):
        f= round(float(float(self.eastDeg)+float(self.eastMin/60)+float(self.eastSec/36
        return -f

    def DD(self):
        return tuple([self.latitudeDecimalDegree(),self.longitudeDecimalDegree()])

    def display(self):
        print("""
            UUID: {uuid}
            Name: {name}
            DMS_lat: {dms_lat}
            DMS_lon: {dms_lon}
            DD_lat: {dd_lat}
            DD_lon: {dd_lon}
            DMS: {dms}
            DD: {dd}
        """.format(uuid=self.UUID(),
                   name=self.name,
                   dms_lat=self.latitudeDMS(),
                   dms_lon=self.longitudeDMS(),
                   dd_lat=self.latitudeDecimalDegree(),
                   dd_lon=self.longitudeDecimalDegree(),
                   dms=self.DMS(),
                   dd=self.DD()))

    def __repr__(self):

        s="""        UUID: {uuid}
                    Name: {name}
                    DMS_lat: {dms_lat}
                    DMS_lon: {dms_lon}
                    DD_lat: {dd_lat}
                    DD_lon: {dd_lon}
                    DMS: {dms}
                    DD: {dd}
                """.format(uuid=self.UUID(),
                           name=self.name,
                           dms_lat=self.latitudeDMS(),
                           dms_lon=self.longitudeDMS(),
                           dd_lat=self.latitudeDecimalDegree(),
                           dd_lon=self.longitudeDecimalDegree(),
                           dms=self.DMS(),
                           dd=self.DD())
        return s
```

### Record.py

```
In [31]: import hashlib,os

         class Record(object):

             def __init__(self,name,address,city,zip_code,web,program_name):
                 self.name=name
                 self.address=address
                 self.city=city
                 self.zip_code=zip_code
                 self.web=web
                 self.program_name=program_name

             def UUID(self,num=31):
                 salt=os.urandom(num).hex()
                 e=str(salt+self.name).encode("UTF")
                 return hashlib.sha512(e).hexdigest()

             def display(self):
                 s = """
                         UUID: {id}
                         Name: {name}
                         Address: {address}
                         City: {city}
```

```
                Zip Code: {zipCode}
                Web: {web}
                Program Name: {pr_name}
                """.format(id=self.UUID(),
                           name=self.name,
                           address=self.address,
                           city=self.city,
                           zipCode=self.zip_code,
                           web=self.web,
                           pr_name=self.program_name)
        return s

    def __repr__(self):
        s="""
        UUID: {id}
        Name: {name}
        Address: {address}
        City: {city}
        Zip Code: {zipCode}
        Web: {web}
        Program Name: {pr_name}
        """.format(id=self.UUID(),
                   name=self.name,
                   address=self.address,
                   city=self.city,
                   zipCode=self.zip_code,
                   web=self.web,
                   pr_name=self.program_name)
        return s
```

### Algorithms.py

```
In [32]: from numpy.linalg import norm as euclidean
         from numpy import array as array

         class Algorithms(object):

             def __init__(self):
                 pass

             def euclideanDistance(self,a):
                 c=[]
                 res=[]
                 z=0
                 while(len(a)>z):
                     analysis=a.copy()
                     del analysis[z]
                     for i in analysis:
```

```
                res.append(euclidean(array(a[z])-array(i)))
            c.append(tuple( [ a[z],analysis[res.index(min(res))]]))
            res.clear()
            z+=1
        return c

    def find(self,target,arr):
        """
        :param target, expecting tuple
        :param list values
        :return shortest Euclidean Distance
        """
        t=tuple(target)
        analysis=list(arr)
        del analysis[analysis.index(t)]
        c=[]
        for i in analysis:
            c.append(euclidean(array(t)-array(i)))
        return analysis[c.index(min(c))]

    def binarySearch(self,arr,k):
        """
        O(log * n)
        :param arr: Sorted Array
        :param k: Key for Selection
        :return: Index
        """
        lo=0
        hi=len(arr)-1
        while(hi>=lo):
            mid=(hi+lo)//2
            if(arr[mid] == k):
                return mid
            elif(arr[mid]>k):
                hi-=1
            elif(arr[mid]<k):
                lo-=1
            else:
                return None
```

## 3 Euclidean Distance Implementation

$$\psi = \sqrt{\sum_{i=0}^{n}(P_i - Q_i)^2} = C$$

```
In [33]: import pickle as p
         from Model.classes.Algorithms import Algorithms
         with open("DATA/NameCoordinateMAP.dat","rb") as f:
             dictionary=p.load(f)
         with open("DATA/CoordinatesNameMAP.dat","rb") as f:
             names=p.load(f)
```

```
In [34]: dictionary
```

```
Out[34]: {'Complete Care Community Hospital': (31.801000000000002, -106.51002777777778),
          'Del Sol Medical Center': (31.75686111111111, -106.3505),
          'El Paso LTAC Hospital': (31.78377777777778, -106.47425),
          'El Paso Speciality Hospital': (31.77872222222222, -106.4775),
          'Hospital Providence of El Paso': (31.770555555555553, -106.50044444444444),
          'Hospital of Providence East Campus': (31.7905, -106.2645),
          'Kindred Hospital El Paso': (31.77872222222222, -106.47725),
          'Las Palmas Rehabilitation Center': (31.78902777777778, -106.50886111111112),
          'Palmas Medical Center': (31.770083333333332, -106.49905555555556),
          'University Medical Center Foundation': (31.78852777777778,
           -106.43511111111111),
          'William Beaumont Army Medical Center': (31.821527777777778,
           -106.46300000000001)}
```

```
In [35]: c=Algorithms().euclideanDistance(list(dictionary.values()))
         [print(i,"\t\t\t\t",j) for (i,j) in c]
         print("\n")
         [print(names[i],"\t\t\t\t",names[j]) for (i,j) in c]
```

```
(31.770555555555553, -106.50044444444444)                                  (31.770083333333332,
(31.770083333333332, -106.49905555555556)                                  (31.770555555555553,
(31.77872222222222, -106.47725)                                  (31.77872222222222, -106.4775)
(31.77872222222222, -106.4775)                                  (31.77872222222222, -106.47725)
(31.78377777777778, -106.47425)                                  (31.77872222222222, -106.47725)
(31.78902777777778, -106.50886111111112)                                  (31.801000000000002, -
(31.801000000000002, -106.51002777777778)                                  (31.78902777777778, -
(31.821527777777778, -106.46300000000001)                                  (31.78377777777778, -
(31.78852777777778, -106.43511111111111)                                  (31.78377777777778, -1
(31.7905, -106.2645)                                  (31.75686111111111, -106.3505)
(31.75686111111111, -106.3505)                                  (31.78852777777778, -106.4351111


Hospital Providence of El Paso                                  Palmas Medical Center
Palmas Medical Center                                  Hospital Providence of El Paso
Kindred Hospital El Paso                                  El Paso Speciality Hospital
El Paso Speciality Hospital                                  Kindred Hospital El Paso
El Paso LTAC Hospital                                  Kindred Hospital El Paso
Las Palmas Rehabilitation Center                                  Complete Care Community Hospit
Complete Care Community Hospital                                  Las Palmas Rehabilitation Cent
William Beaumont Army Medical Center                                  El Paso LTAC Hospital
University Medical Center Foundation                                  El Paso LTAC Hospital
Hospital of Providence East Campus                                  Del Sol Medical Center
```

```
Del Sol Medical Center                                University Medical Center Foundation
```

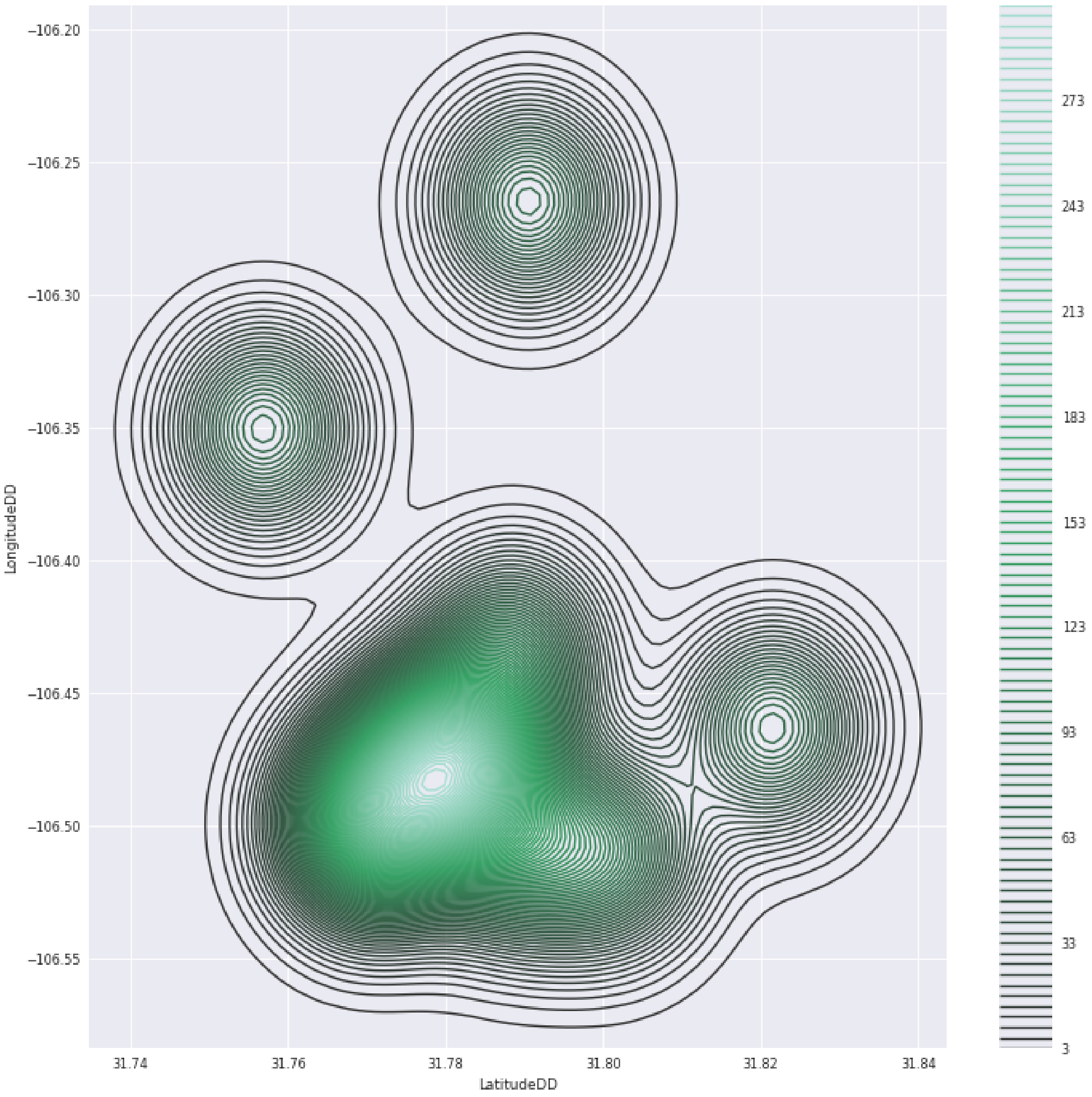
```
Out[35]: [None, None, None, None, None, None, None, None, None, None, None]
```

# 4 Visualization

```
In [36]: import matplotlib.pyplot as plt
         import seaborn as sns
         import pandas as pd
         %matplotlib inline
         with open("DATA/NameCoordinates.dat","rb") as f:
             d=pd.DataFrame(p.load(f))
```

```
In [37]: plt.figure(figsize=(12,12))
         sns.set_style("darkgrid"); sns.set_context("paper")
         sns.kdeplot(d["LatitudeDD"],d["LongitudeDD"],cbar=True,n_levels=100)
```

```
Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x7f839bb1f160>
```


−106.20
−106.25
−106.30
−106.35
−106.40
−106.45
−106.50
−106.55
LongitudeDD
31.74
31.76
31.78
31.80
31.82
31.84
LatitudeDD
273
243
213
183
153
123
93
63
33
3