Euclidean and Manhattan Distance Calculations

In this short mini project you will see examples and comparisons of distance measures. Specifically, you'll visually compare the Euclidean distance to the Manhattan distance measures. The application of distance measures has a multitude of uses in data science and is the foundation of many algorithms you'll be using such as Prinical Components Analysis.

by: Sonjoy Das, PhD

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.cm as cm
%matplotlib inline
plt.style.use('ggplot')

# 3D cluster plot
from mpl_toolkits import mplot3d
```

In [2]:

```
# Load Course Numerical Dataset
df = pd.read_csv('data/distance_dataset.csv',index_col=0)
df.head()
```

Out[2]:

	X	Y	Z	ClusterID
0	5.135779	4.167542	5.787635	4
1	4.280721	5.770909	6.091044	4
2	8.329098	7.540436	3.247239	2
3	5.470224	5.069249	5.768313	4
4	2.381797	2.402374	3.879101	1

In [3]:

```
df
```

Out[3]:

	X	Y	Z	ClusterID
0	5.135779	4.167542	5.787635	4
1	4.280721	5.770909	6.091044	4
2	8.329098	7.540436	3.247239	2
3	5.470224	5.069249	5.768313	4
4	2.381797	2.402374	3.879101	1
...	...	...	...	...
1995	4.616245	4.019561	5.522939	4
1996	4.753185	5.065076	8.074947	3
1997	2.000186	2.351911	6.779311	1
1998	4.735917	5.642677	4.855780	4

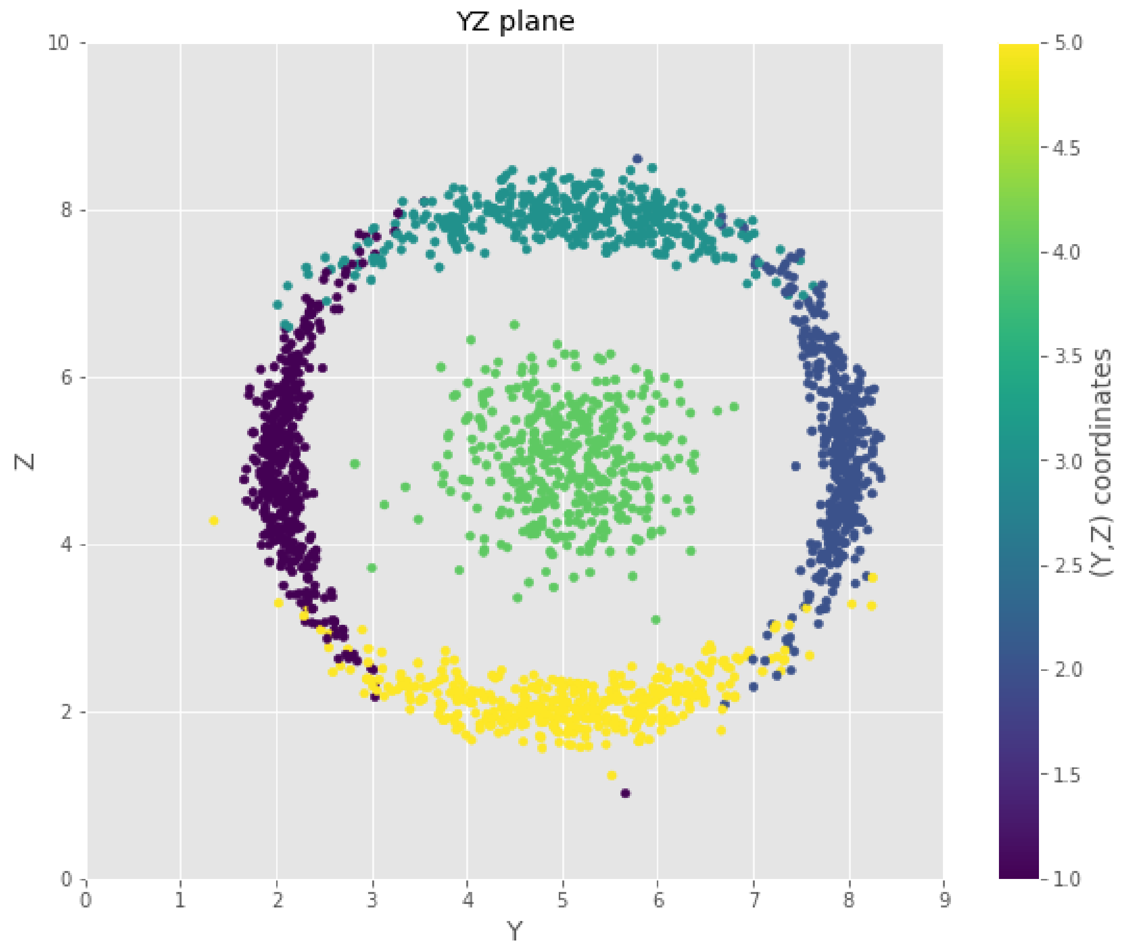
	X	Y	Z	ClusterID
1999	4.955436	5.270550	7.844768	3

2000 rows × 4 columns

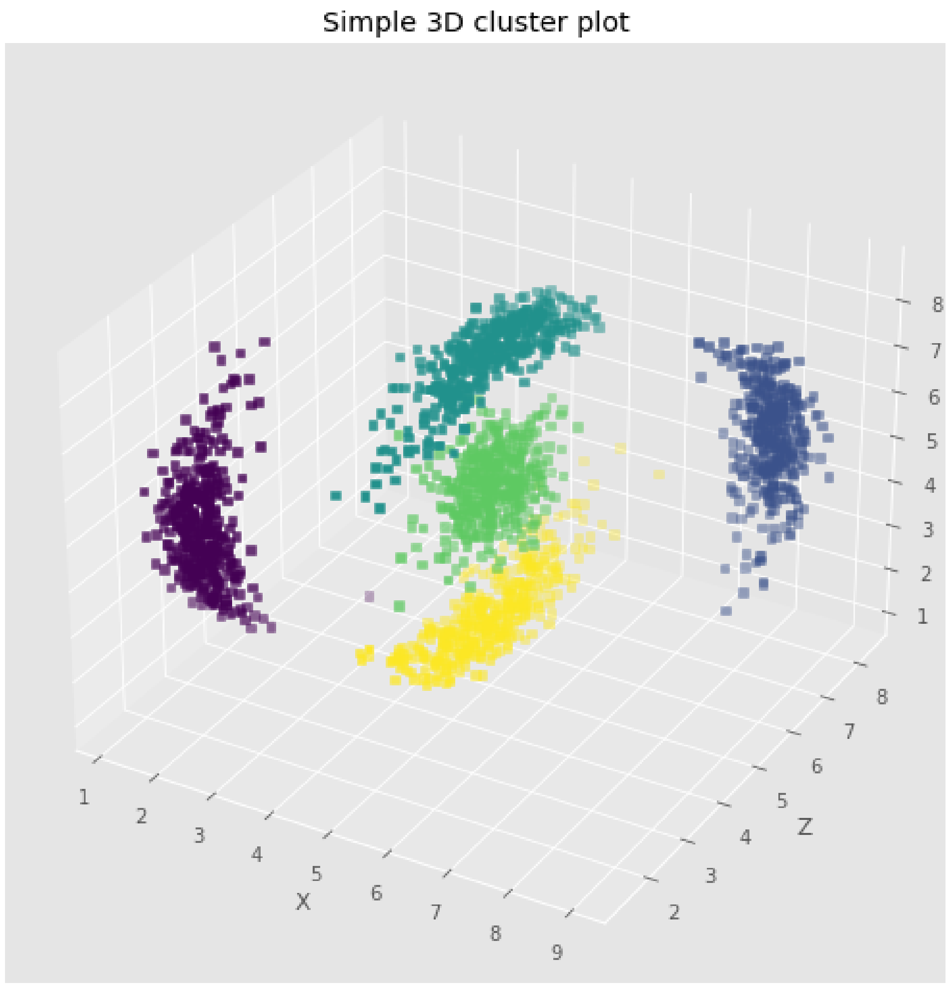
In [4]:
```
df.ClusterID.value_counts()
```

Out[4]:
```
1    400
2    400
3    400
4    400
5    400
Name: ClusterID, dtype: int64
```

In [5]:
```
# Cluster plot on YZ plane
figYZ = plt.figure(figsize=[10,8])

plt.scatter(df['Y'],df['Z'],c=df.ClusterID,s=20)
plt.ylim([0,10])
plt.xlim([0,9])
plt.xlabel('Y', size=14)
plt.ylabel('Z', size=14)
plt.title('YZ plane')
cb = plt.colorbar()
cb.set_label('(Y,Z) coordinates', size=14)
plt.show()
```



In [6]:
```
# 3D cluster plot
fig = plt.figure(figsize = (10, 9))
ax = plt.axes(projection ="3d")
ax.scatter3D(df['X'],df['Y'],df['Z'], marker="s", c=df.ClusterID, s=20)
plt.title("Simple 3D cluster plot")
plt.xlabel('X')
plt.ylabel('Y')
plt.ylabel('Z')
plt.show()
```



Euclidean Distance

Let's visualize the difference between the Euclidean and Manhattan distance.

We are using Pandas to load our dataset .CSV file and use Numpy to compute the **Euclidean distance** to the point (Y=5, Z=5) that we choose as reference. On the left here we show the dataset projected onto the YZ plane and color coded per the Euclidean distance we just computed. As we are used to, points that lie at the same Euclidean distance define a regular 2D circle of radius that distance.

Note that the **SciPy library** comes with optimized functions written in C to compute distances (in the scipy.spatial.distance module) that are much faster than our (naive) implementation.

In [7]:
```
# In the Y-Z plane, we compute the distance to ref point (5,5)
distEuclid = np.sqrt((df.Z - 5)**2 + (df.Y - 5)**2)
```

Create a distance to reference point (3,3) matrix similar to the above example.

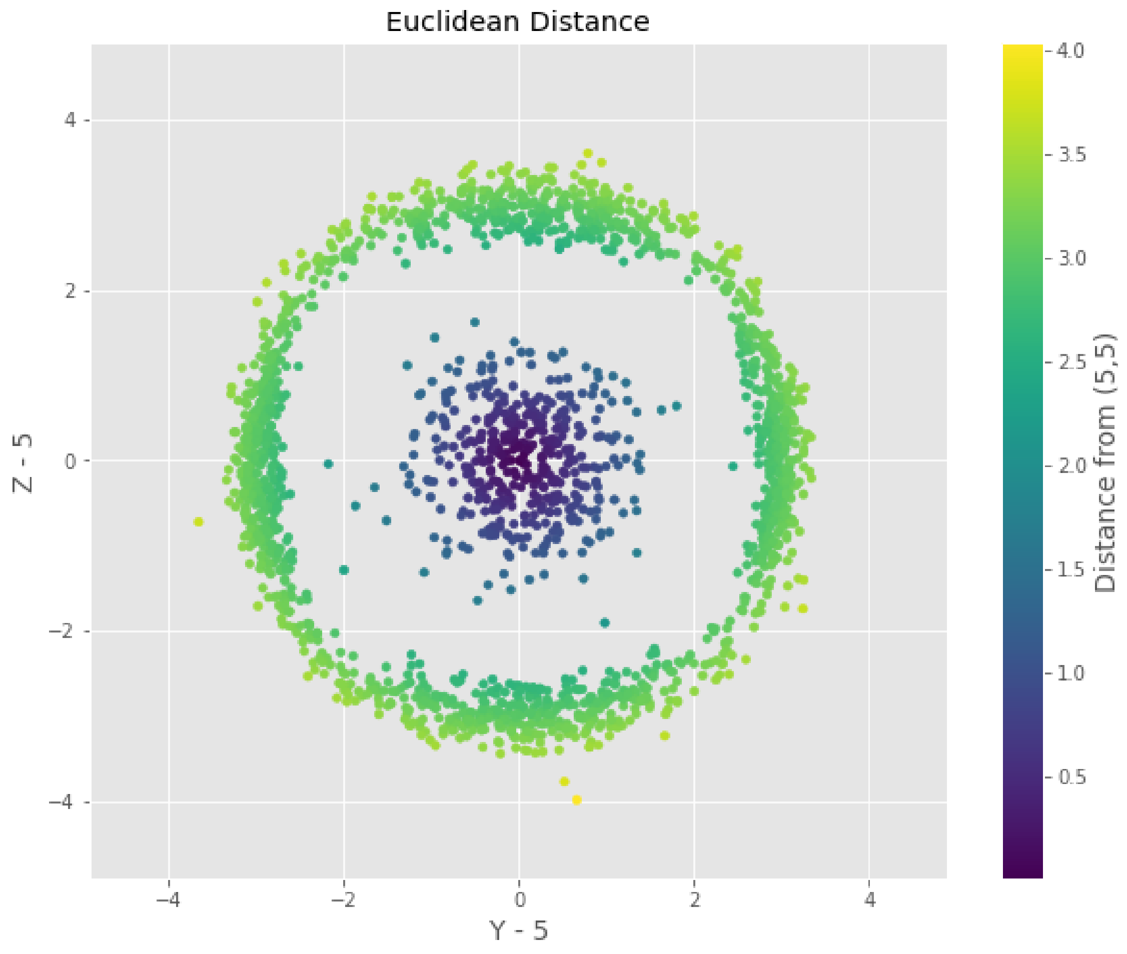
In [8]:

```
distEuclid_YZ33 = np.sqrt((df.Z - 3)**2 + (df.Y - 3)**2)
```

Replace the value set to 'c' in the plotting cell below with your own distance matrix and review the result to deepen your understanding of Euclidean distances.

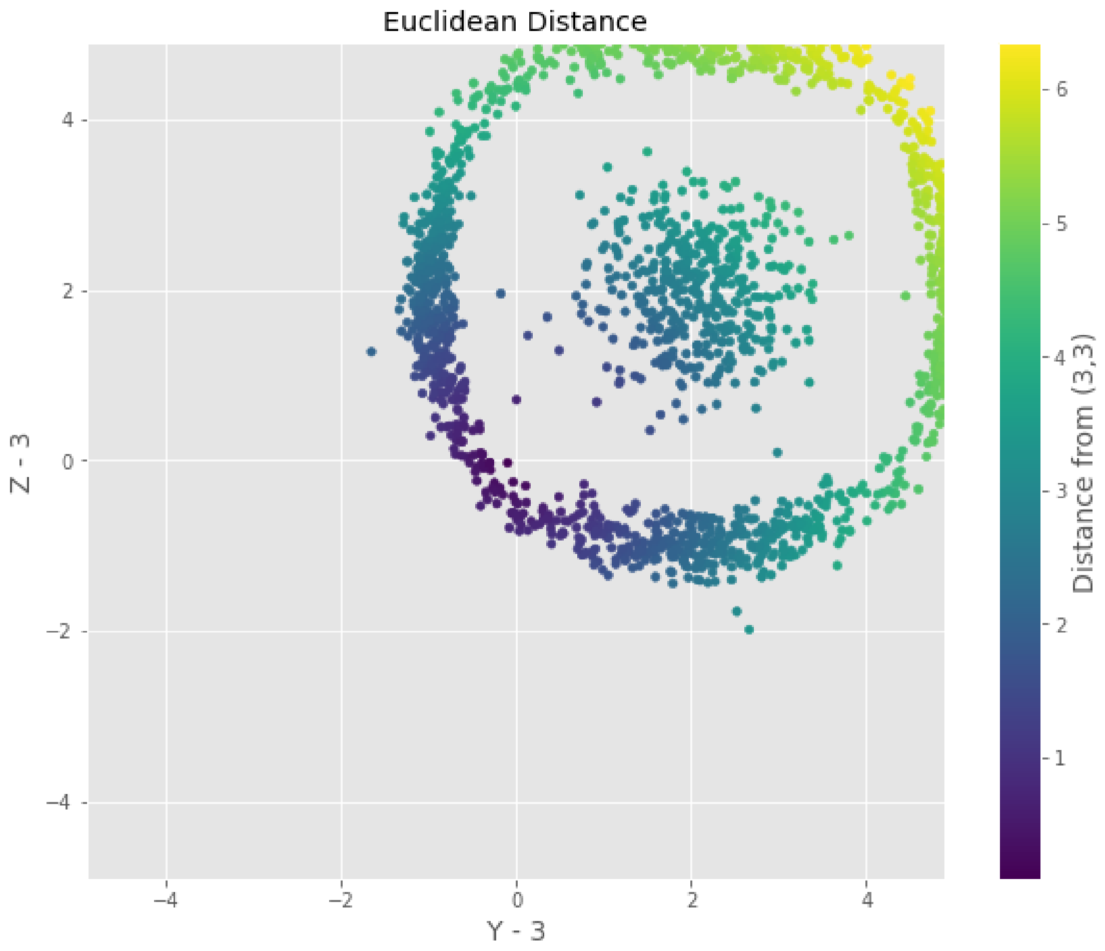
In [9]:

```
figEuclid = plt.figure(figsize=[10,8])

plt.scatter(df.Y - 5, df.Z-5, c=distEuclid, s=20)
plt.ylim([-4.9,4.9])
plt.xlim([-4.9,4.9])
plt.xlabel('Y - 5', size=14)
plt.ylabel('Z - 5', size=14)
plt.title('Euclidean Distance')
cb = plt.colorbar()
cb.set_label('Distance from (5,5)', size=14)

#figEuclid.savefig('plots/Euclidean.png')
```



In [10]:

```
figEuclid_YZ33 = plt.figure(figsize=[10,8])

plt.scatter(df.Y - 3, df.Z-3, c=distEuclid_YZ33, s=20)
plt.ylim([-4.9,4.9])
plt.xlim([-4.9,4.9])
plt.xlabel('Y - 3', size=14)
plt.ylabel('Z - 3', size=14)
plt.title('Euclidean Distance')
cb = plt.colorbar()
cb.set_label('Distance from (3,3)', size=14)
```



Manhattan Distance

Manhattan distance is simply the sum of absolute differences between the points coordinates. This distance is also known as the taxicab or city block distance as it measure distances along the coorinate axis which creates "paths" that look like a cab's route on a grid-style city map.

We display the dataset projected on the XZ plane here color coded per the Manhattan distance to the (X=5, Z=5) reference point. We can see that points laying at the same distance define a circle that looks like a Euclidean square.

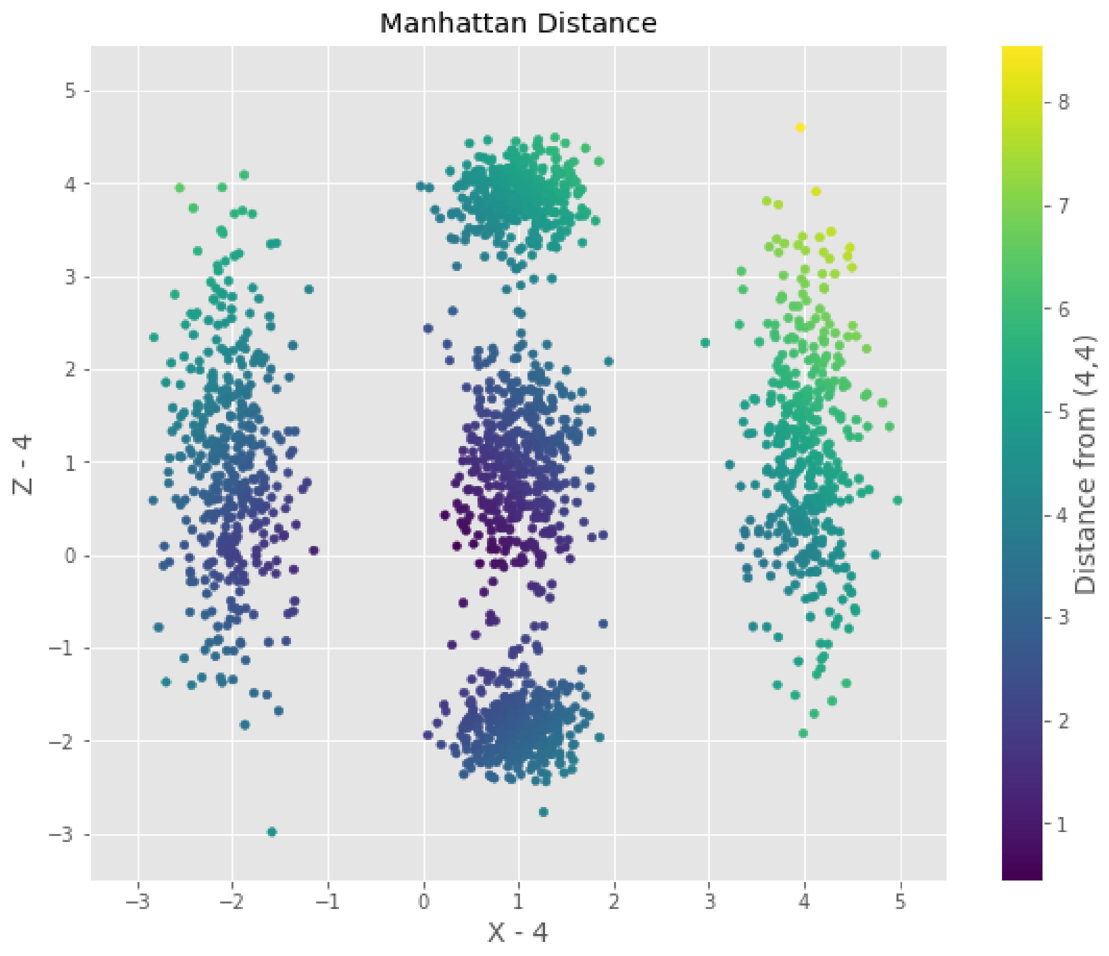
In [11]:

```
# In the X-Z plane, we compute the distance to ref point (5,5)
distManhattan = np.abs(df.X - 5) + np.abs(df.Z - 5)
```

Create a Manhattan distance to reference point (4,4) matrix similar to the above example and replace the value for 'c' in the plotting cell to view the result.

In [12]:

```
distManhattan_XZ44 = np.abs(df.X - 4) + np.abs(df.Z - 4)

figEuclid_XZ44 = plt.figure(figsize=[10,8])
plt.xlim([-3.5,5.5])
plt.ylim([-3.5,5.5])
plt.scatter(df.X - 4, df.Z-4, c=distManhattan_XZ44, s=20)
plt.xlabel('X - 4', size=14)
plt.ylabel('Z - 4', size=14)
plt.title('Manhattan Distance')
cb = plt.colorbar()
cb.set_label('Distance from (4,4)', size=14)
```



Now let's create distributions of these distance metrics and compare them. We leverage the scipy dist function to create these matrices similar to how you manually created them earlier in the exercise.

In [13]:

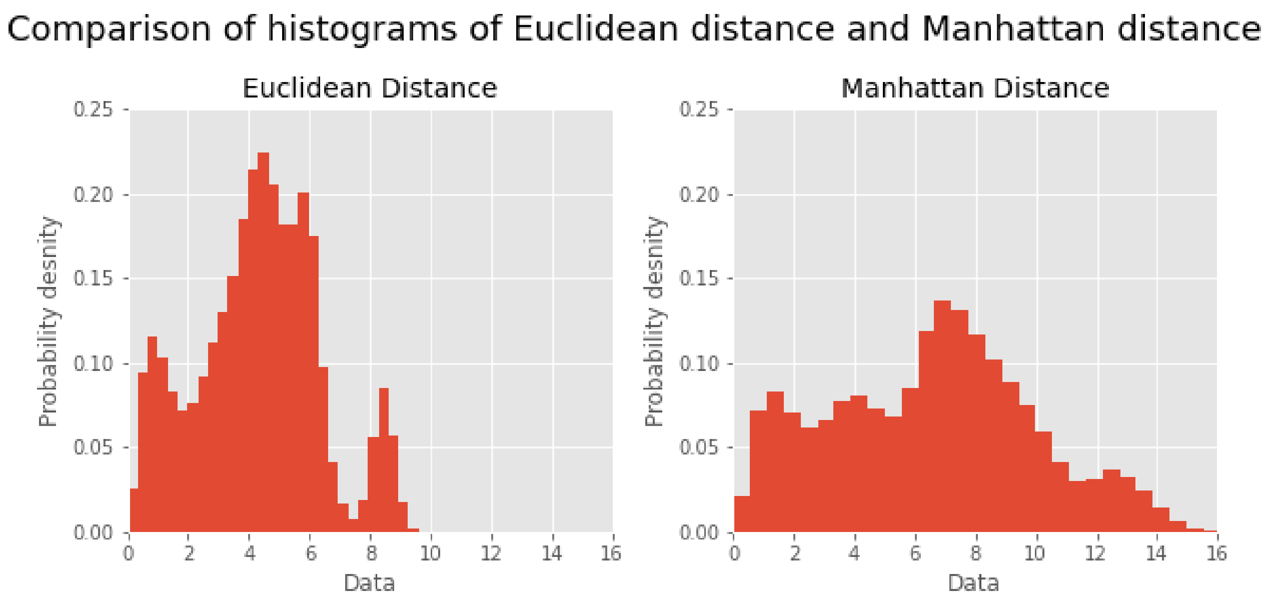
```
import scipy.spatial.distance as dist

mat = df[['X','Y','Z']].to_numpy()
DistEuclid = dist.pdist(mat,'euclidean')
DistManhattan = dist.pdist(mat, 'cityblock')
```

Plot histograms of each distance matrix for comparison.

In [14]:

```
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(9, 4.5), tight_layout=True)
fig.suptitle('Comparison of histograms of Euclidean distance and Manhattan distance'


ax1.hist(DistEuclid, density=True, bins=30)  # density=False would make counts
ax1.set_xlim([0, 16])
ax1.set_ylim([0, 0.25])
ax1.set_title('Euclidean Distance',size=14)
ax1.set_xlabel('Data')
ax1.set_ylabel('Probability desnity');
```

```python
ax2.hist(DistManhattan, density=True, bins=30)  # density=False would make counts
ax2.set_xlim([0, 16])
ax2.set_ylim([0, 0.25])
ax2.set_title('Manhattan Distance',size=14)
ax2.set_xlabel('Data')
ax2.set_ylabel('Probability desnity');
```



The comparison of histograms shows that the Euclidean distance is narrower with high probability density values, while the Manhattan distance is relatively spread over a slightly broader support and has, therefore, low values of probability density. Both the densities are normalized to 1, i.e., area under the curve is 1. The narrower support as shown in Euclidean distance and the slightly broader support for Manhattan distance are not surprising because the Euclidean distance represents the hypotenuse of a right traingle while the Manhattan distance is simply the sum of the length of other two sides of a right traingle.