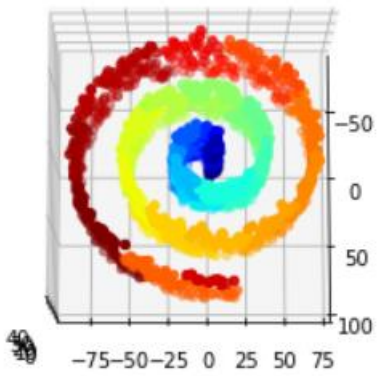
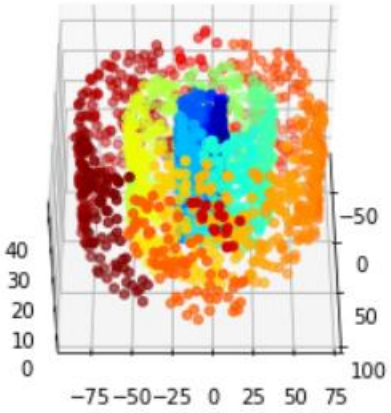
6-1、6-3

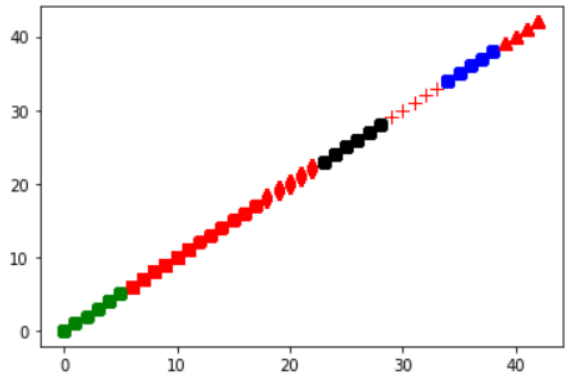
In [6]:
```
partition = community.best_partition(G)
size = float(len(set(partition.values())))
print("community:", size)
mod = community.modularity(partition,G)
print("modularity:", mod)
```

```
community: 43.0
modularity: 0.9373174232366818
```





In [17]:
```
label_pred = kmeans.labels_ #獲取標籤
centroids = kmeans.cluster_centers_ #獲取中心
inertia = kmeans.inertia_ # 總和
mark = ['or', 'ob', 'og', 'ok', '^r', '+r', 'sr', 'dr']
color = 0
j =int(0)
for i in label_pred:
    plt.plot([label[j:j+1,0]], [label[j:j+1,0]], mark[i], markersize = 8)
    j +=1
plt.show()
```



6-2、6-4

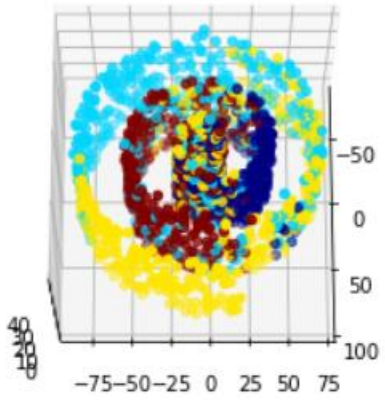
```
#Use pre-defined linkage (Edges.csv) to constructure whole network
G = nx.Graph()
for i in range(0, len(XYZ_E)):
    e = ( str(int(XYZ_E[i,0])), str(int(XYZ_E[i,1])), XYZ_E[i-660690,2] )
    G.add_weighted_edges_from([(e)])
```

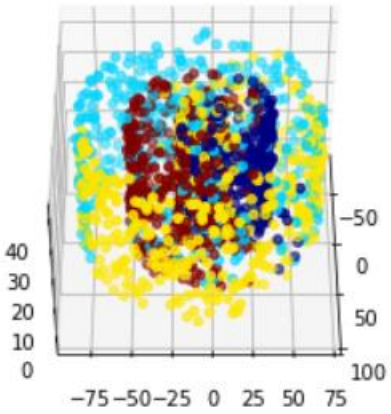
In [11]:
```
e #最大原子距離
```

Out[11]:
```
('1998', '1999', 124.42298697809942)
```

In [12]:
```
partition = community.best_partition(G)
size = float(len(set(partition.values())))
print("community:", size)
mod = community.modularity(partition,G)
print("modularity:", mod)
```

```
community: 4.0
modularity: 0.04691323089218934
```


−50
0
50
100
40
30
20
10
0
−75 −50 −25 0 25 50 75


−50
0
50
100
40
30
20
10
0
−75 −50 −25 0 25 50 75


kmeans(database_1999000 with best_partition)
database_1999000 with best_partition
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0 0.5 1.0 1.5 2.0 2.5 3.0