

元音-单元音-前元音-[æ]

文件格式: .wav

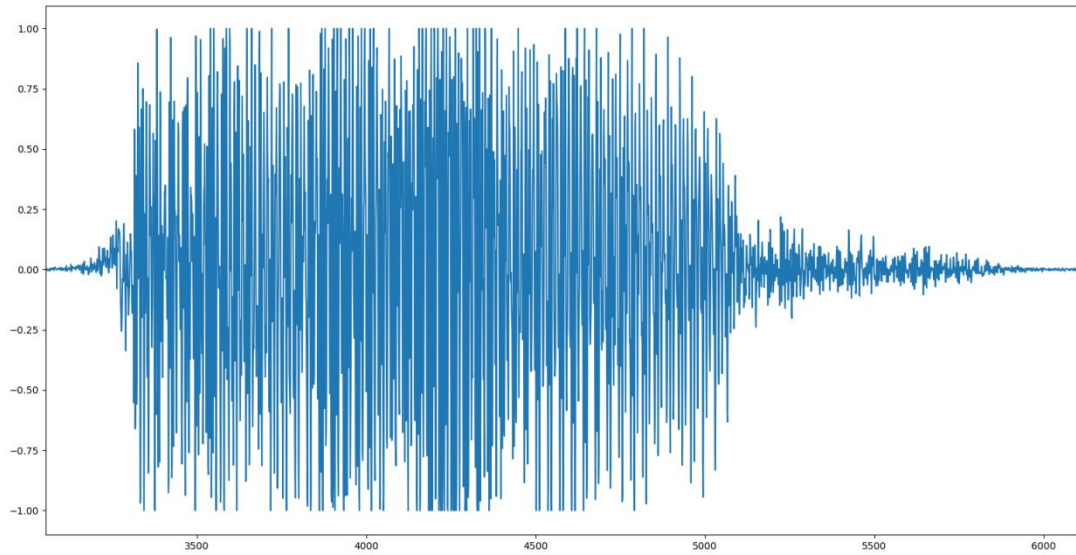
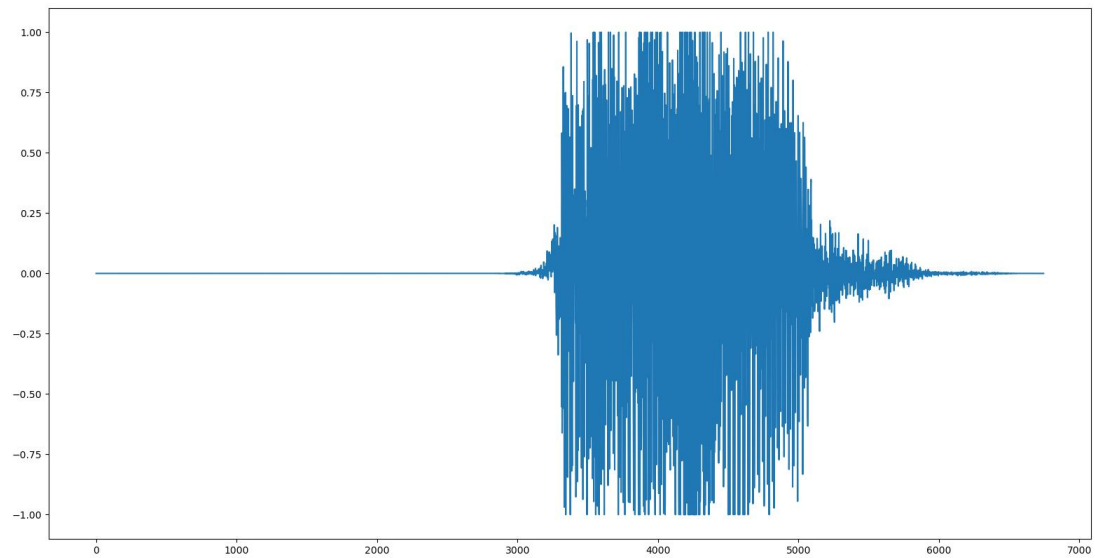
波形图

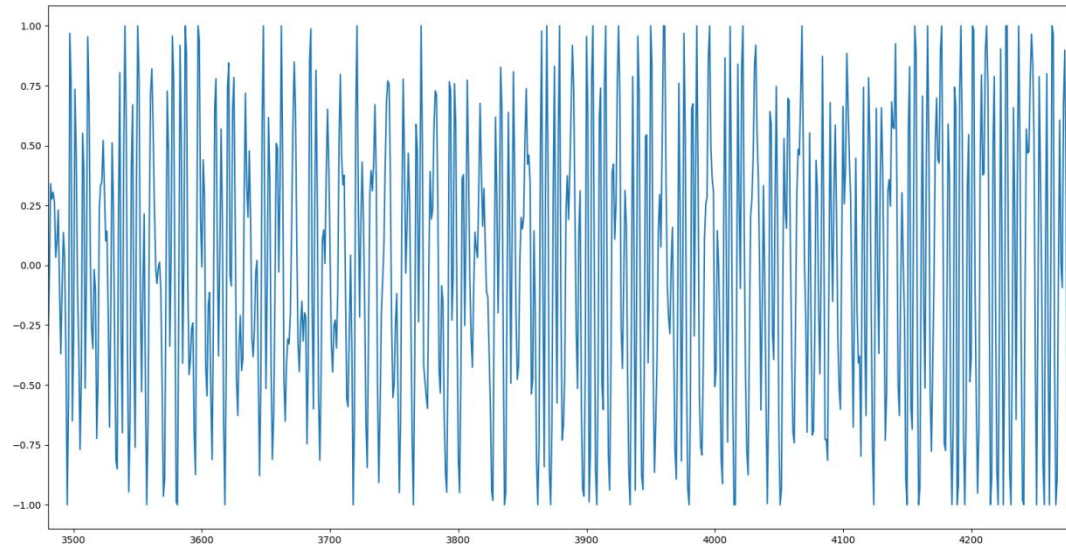
```
Str='D:\单元音\前元音\[æ]_1.WAV'
```

```
sig, samplerate = sf.read(Str)
```

```
f = plt.figure()
```

```
plt.plot(sig)
```





Sliding Window Embedding

Dimension=2

Delay=37, skip=1

```
point_Cloud=timedelay.TimeDelayEmbedding(dim=2, delay=100, skip=1)
```

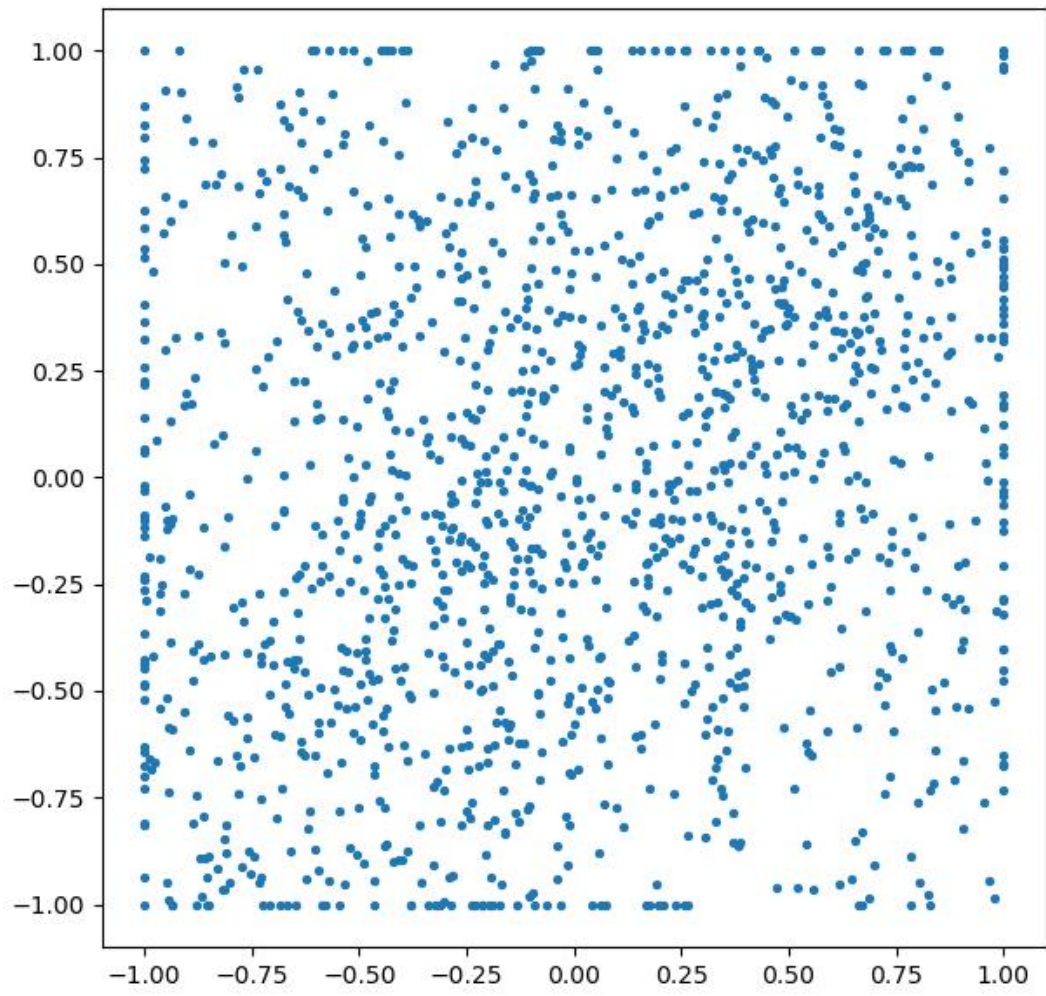
```
Points=point_Cloud(sig[3500:5000])
```

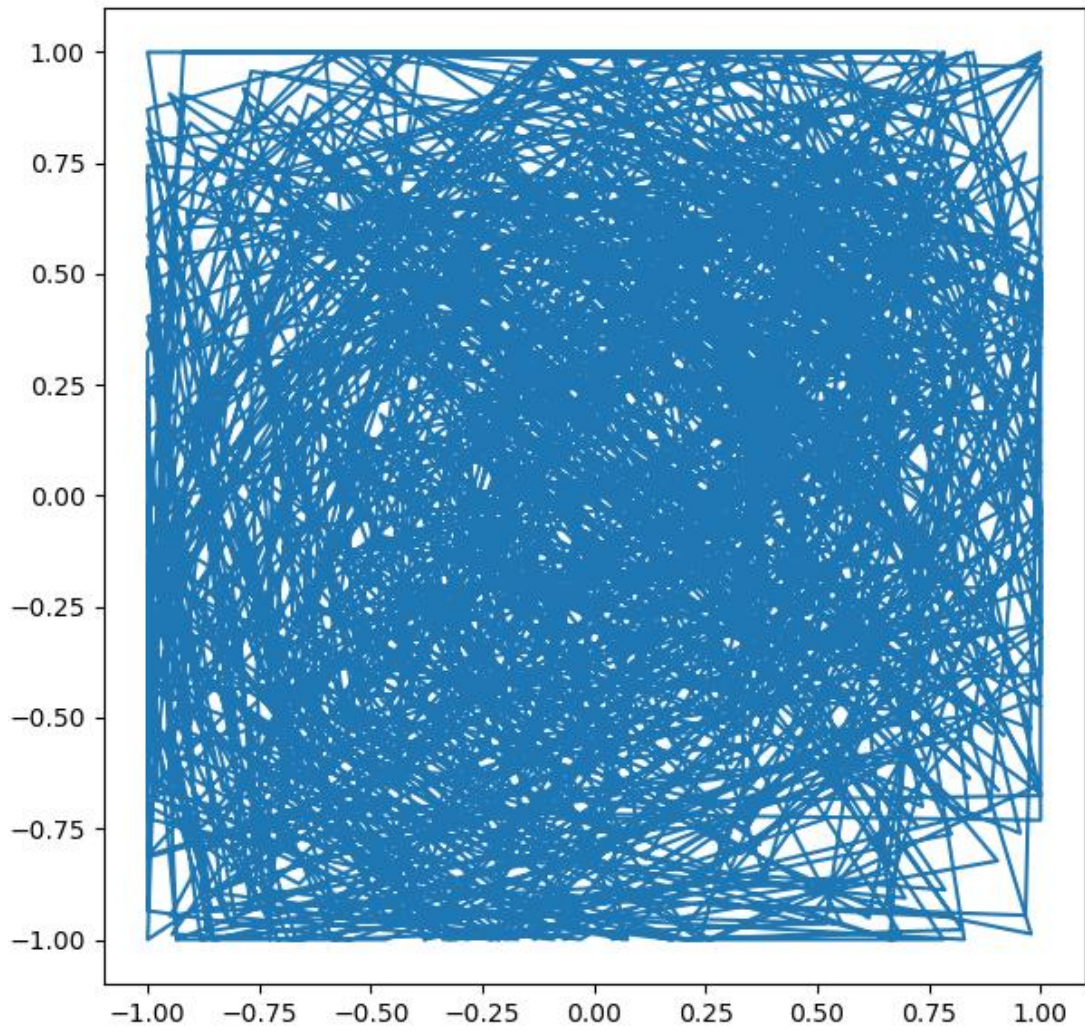
```
plt.figure(figsize=(7,7))
```

```
plt.scatter(Points[:,0], Points[:,1],s=8)
```

```
plt.figure(figsize=(7,7))
```

```
plt.plot(Points[:,0], Points[:,1])
```





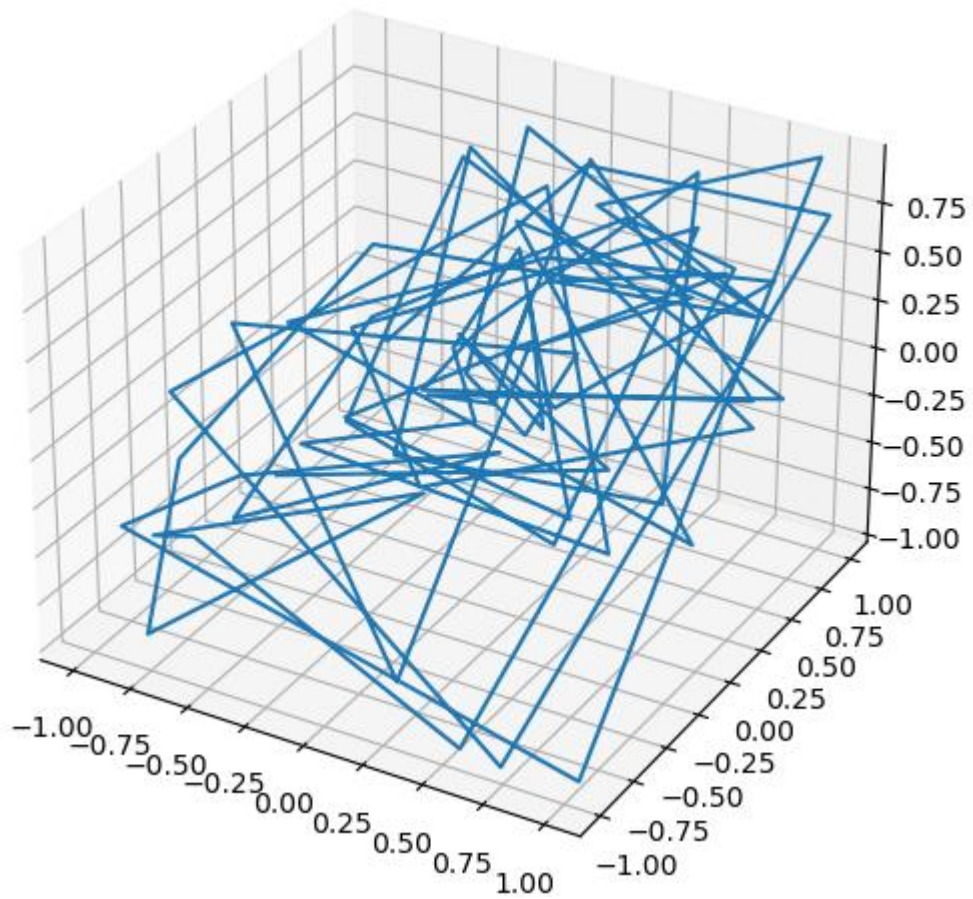
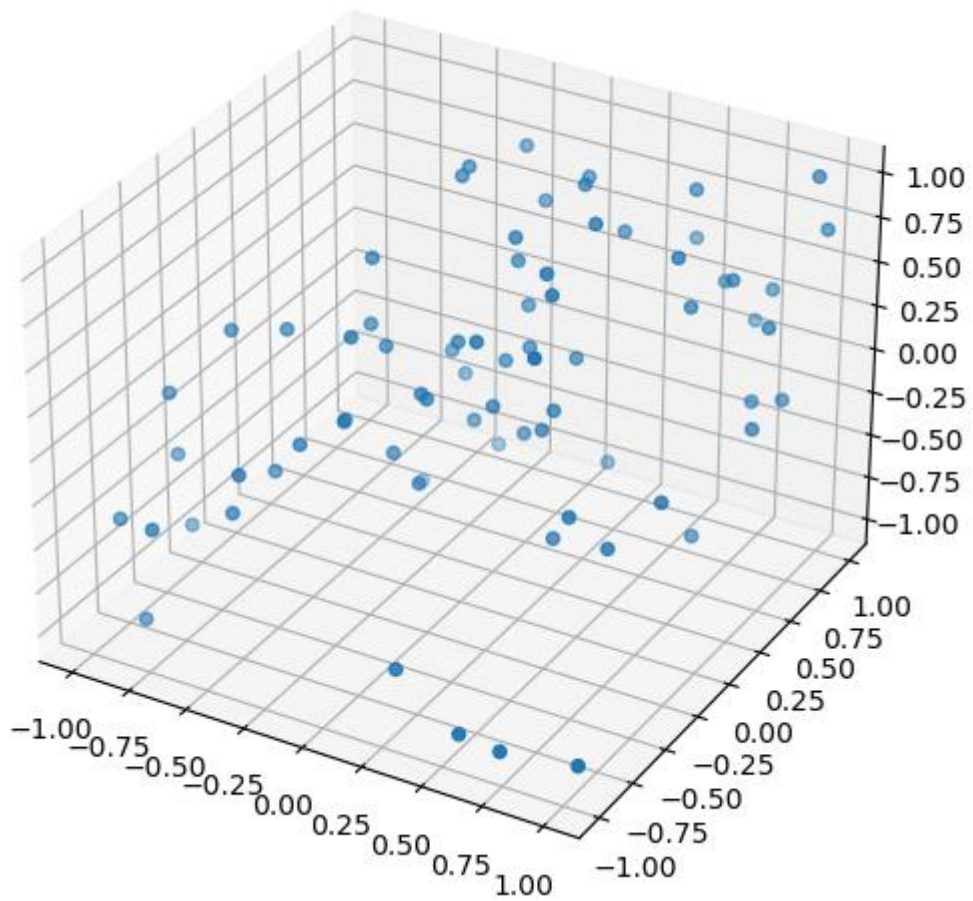
Dimension=3

Delay=10, skip=20

```
point_Cloud=timedelay.TimeDelayEmbedding(dim=3, delay=10, skip=20)  
Points=point_Cloud(sig[3500:5000])
```

```
fig=plt.figure()  
ax=Axes3D(fig)  
ax.scatter(Points[:,0],Points[:,1],Points[:,2])
```

```
fig=plt.figure()  
ax=Axes3D(fig)  
ax.plot(Points[:,0],Points[:,1],Points[:,2])
```

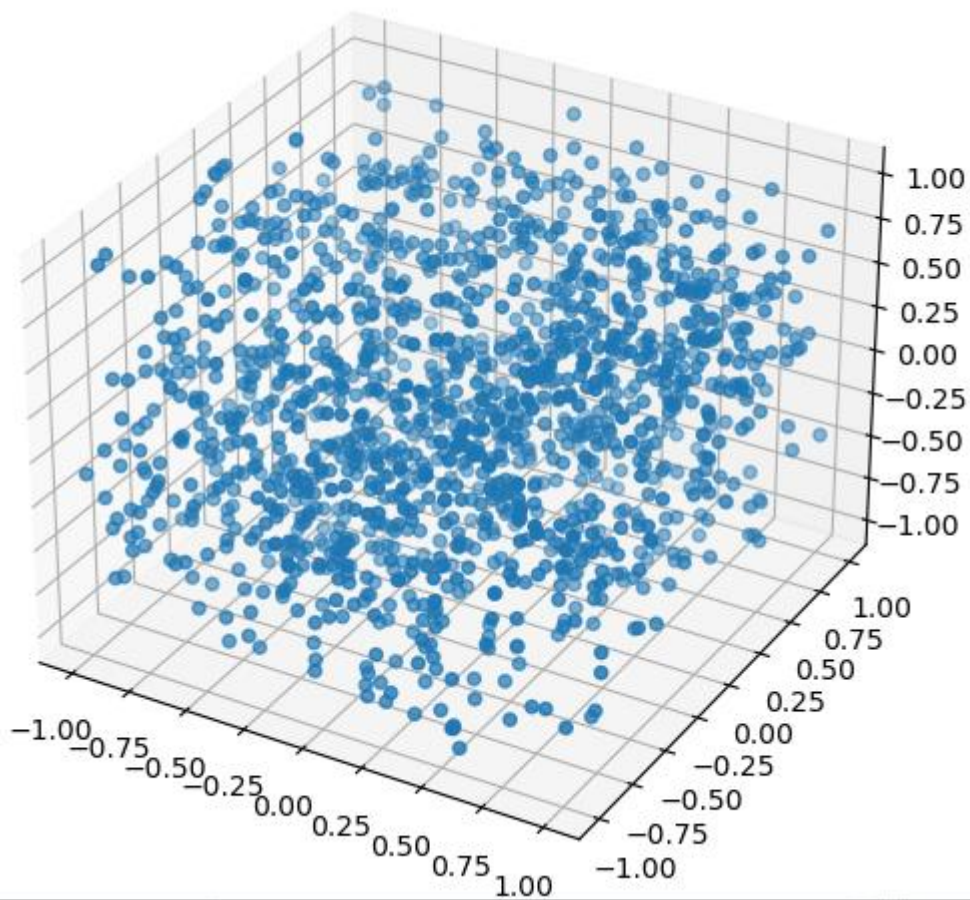



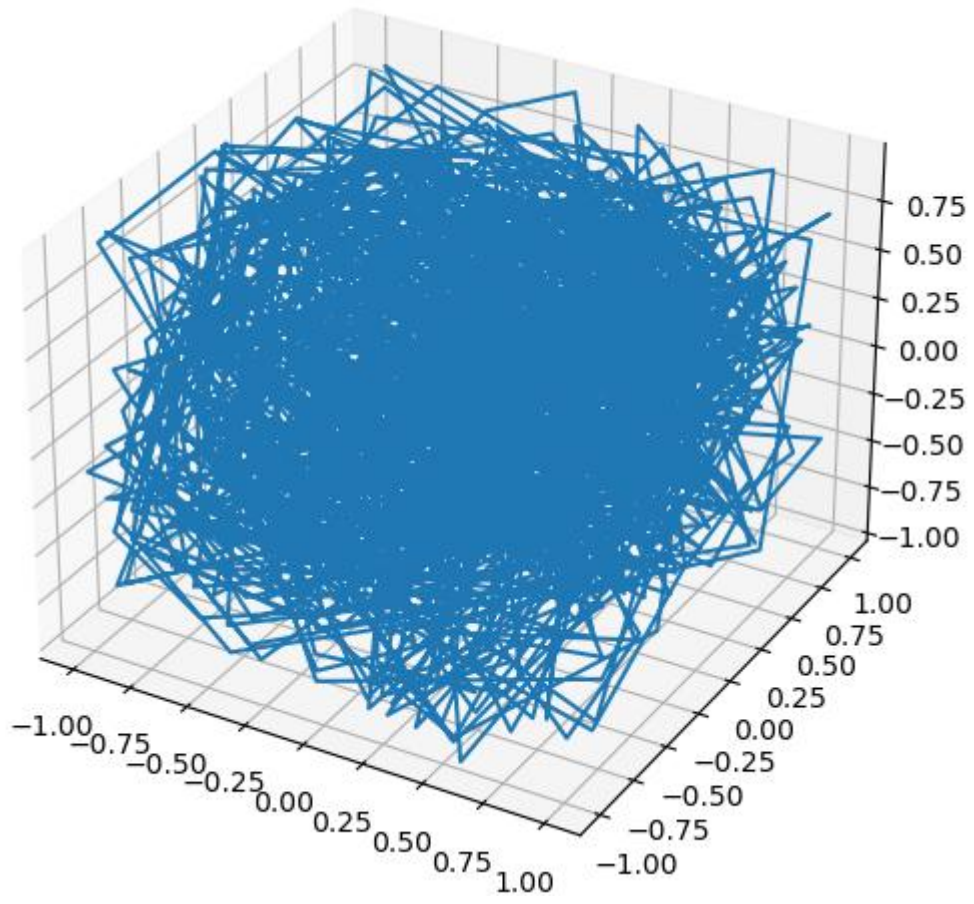
```
Delay=47, skip=1
```

```
point_Cloud=timedelay.TimeDelayEmbedding(dim=3, delay=10, skip=20)  
Points=point_Cloud(sig[3500:5000])
```

```
fig=plt.figure()  
ax=Axes3D(fig)  
ax.scatter(Points[:,0],Points[:,1],Points[:,2])
```

```
fig=plt.figure()  
ax=Axes3D(fig)  
ax.plot(Points[:,0],Points[:,1],Points[:,2])
```





辅音-爆破音-浊辅音-[b]

文件格式: .wav

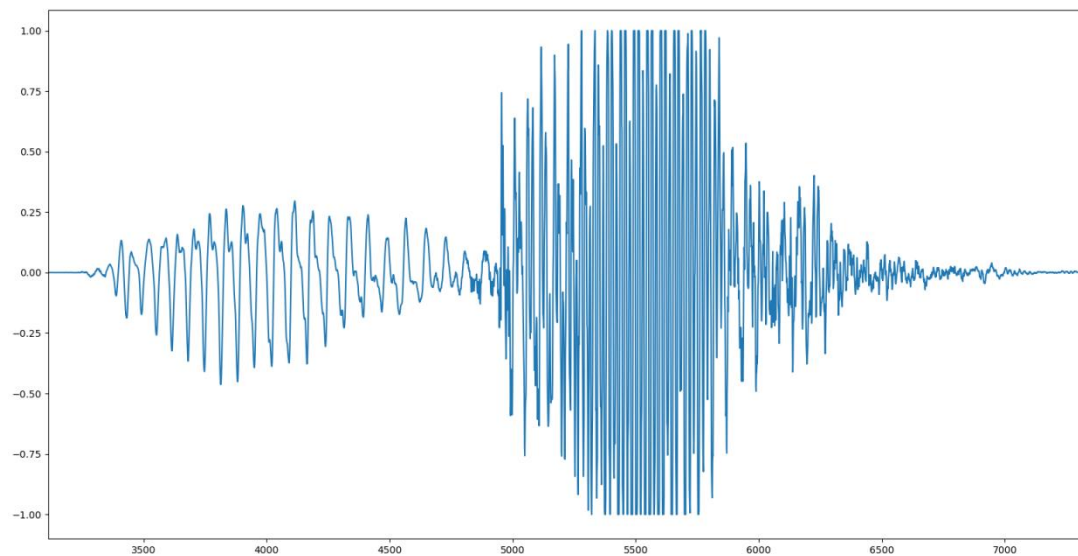
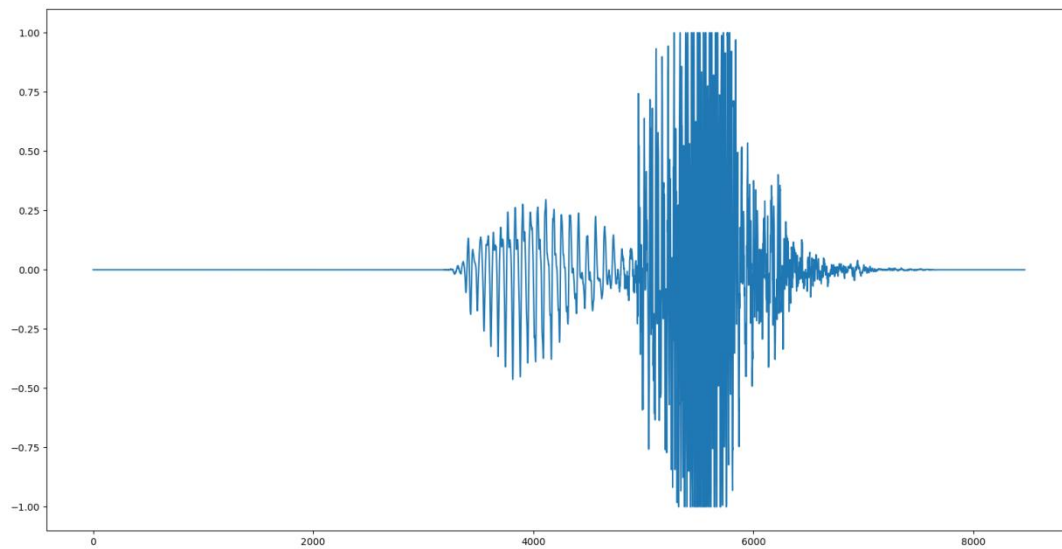
波形图

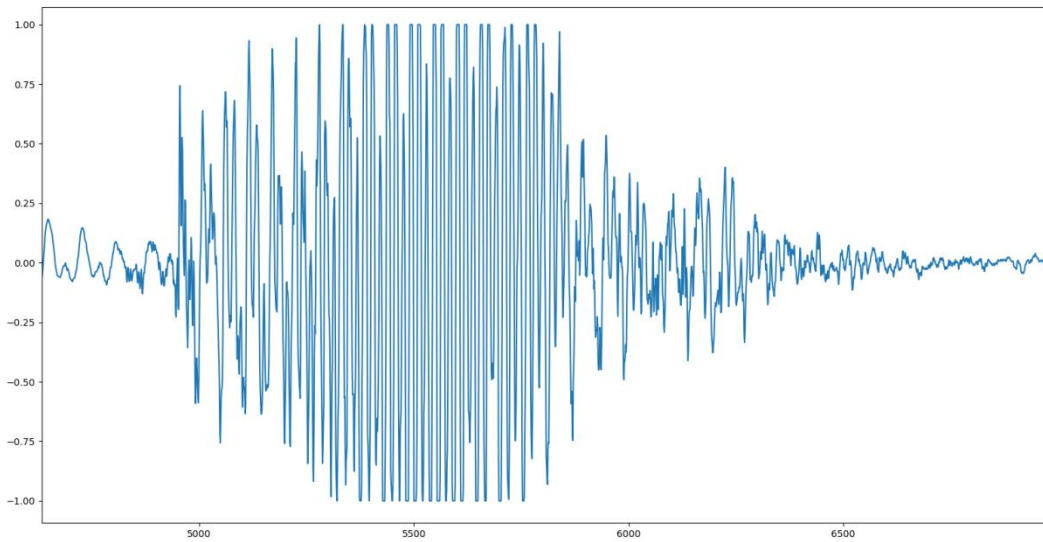
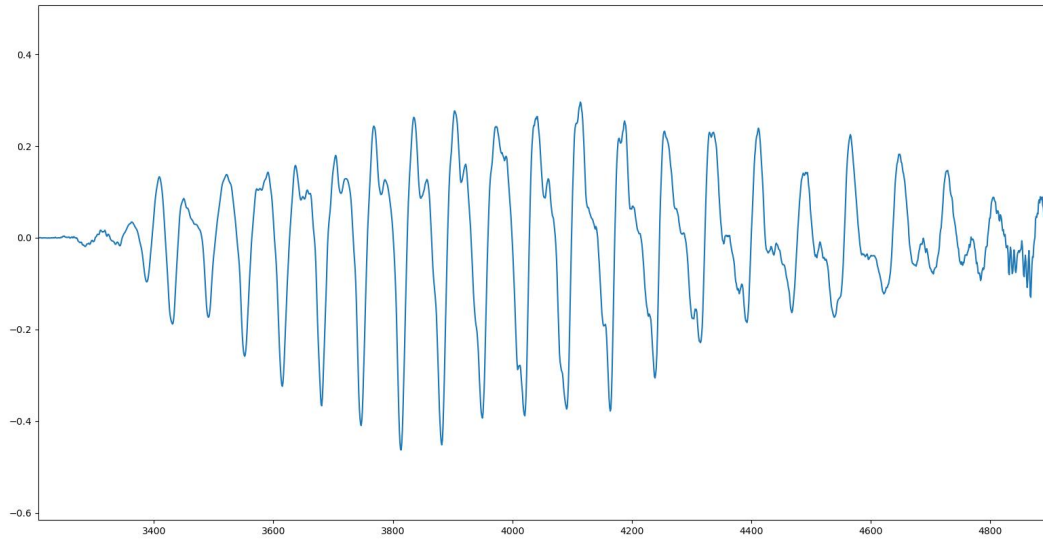
```
Str='D:\单元音\[b]_1.WAV'
```

```
sig, samplerate = sf.read(Str)
```

```
f = plt.figure()
```

```
plt.plot(sig)
```





Sliding window embedding

Dimension=2

Delay=20, skip=1

```
point_Cloud=timedelay.TimeDelayEmbedding(dim=2, delay=20, skip=1)
```

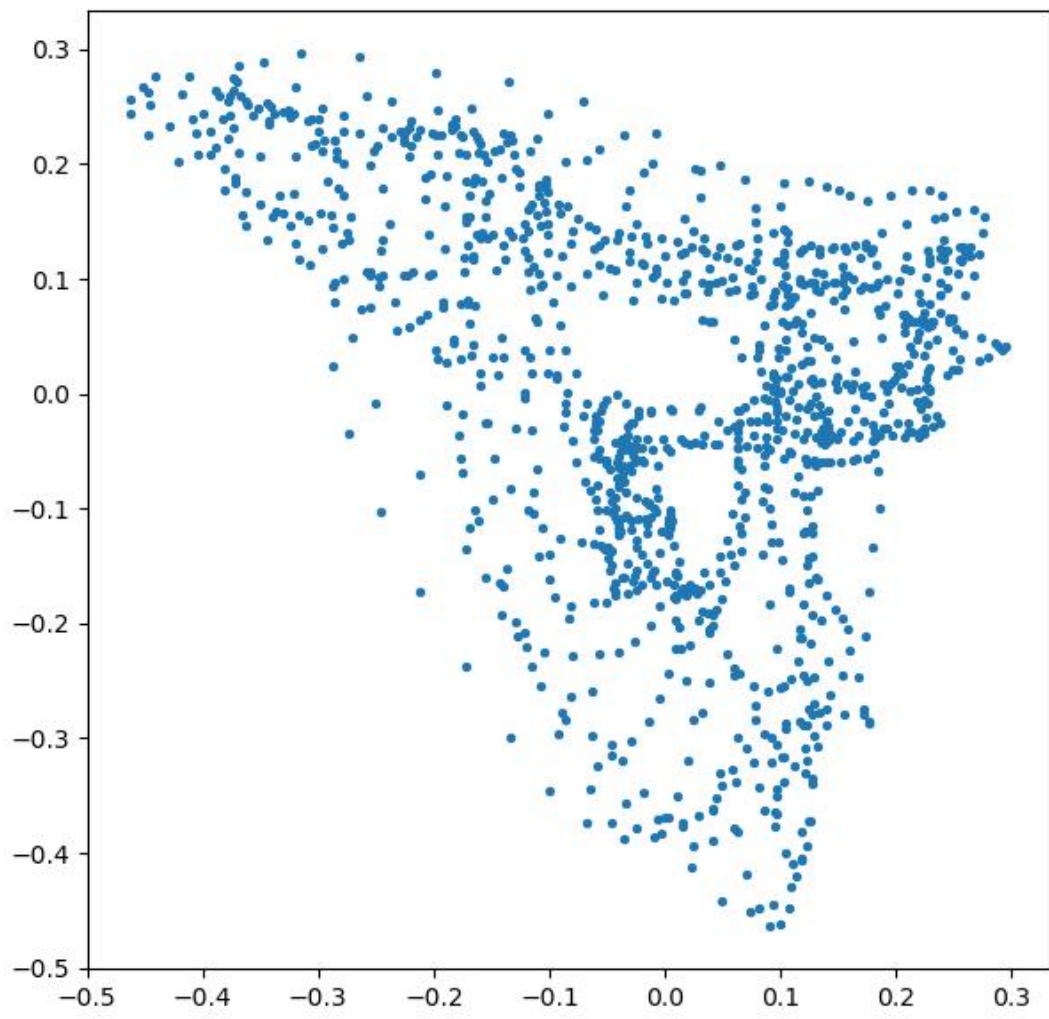
```
Points=point_Cloud(sig[3500:4700])
```

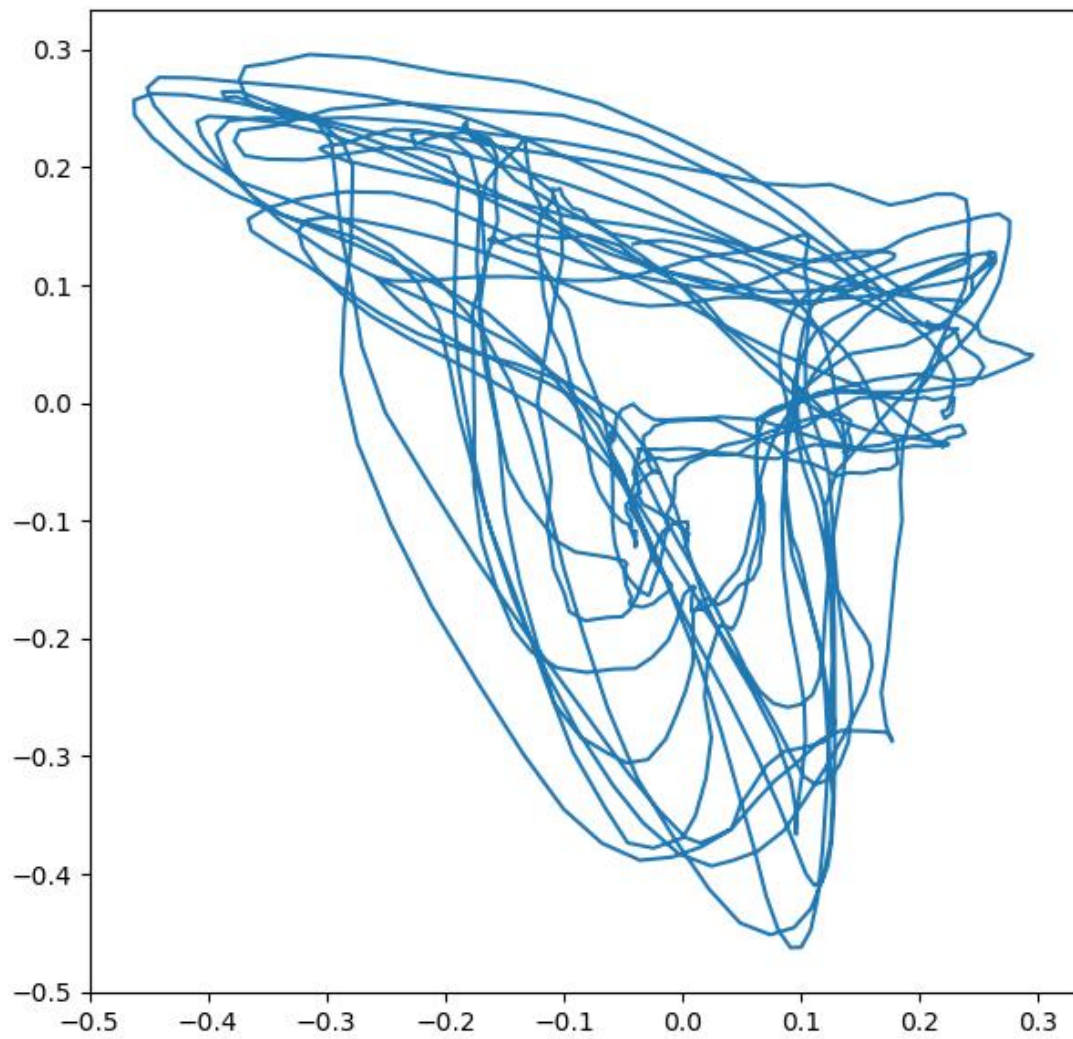
```
plt.figure(figsize=(7,7))
```

```
plt.scatter(Points[:,0], Points[:,1],s=8)
```

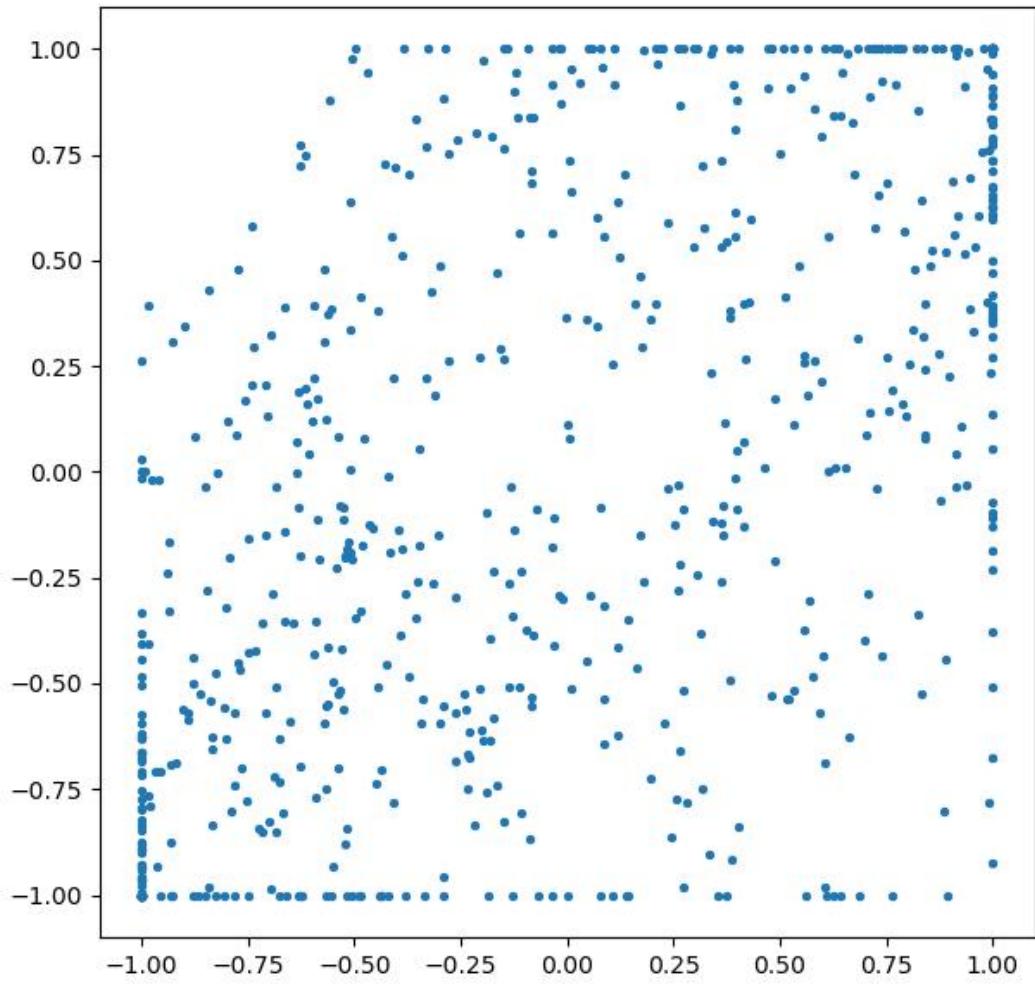
```
plt.figure(figsize=(7,7))
```

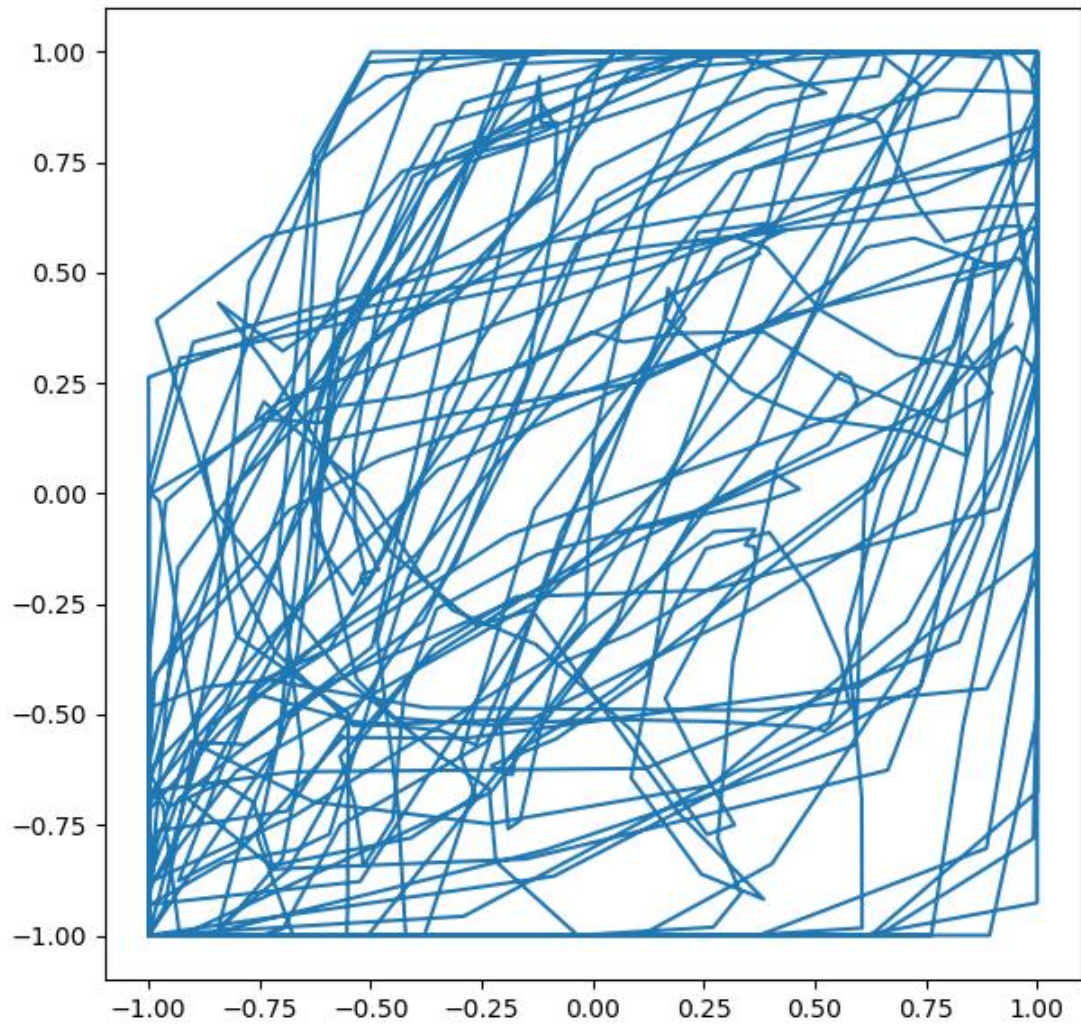
```
plt.plot(Points[:,0],Points[:,1])
```





```
point_Cloud=timedelay.TimeDelayEmbedding(dim=2, delay=20, skip=1)
Points=point_Cloud(sig[5100:5800])
plt.figure(figsize=(7,7))
plt.scatter(Points[:,0], Points[:,1],s=8)
plt.figure(figsize=(7,7))
plt.plot(Points[:,0],Points[:,1])
```





Dimension=3

Delay=10, skip=1

```
point_Cloud=timedelay.TimeDelayEmbedding(dim=3, delay=10, skip=1)
```

```
Points=point_Cloud(sig[3500:4700])
```

```
fig=plt.figure()
```

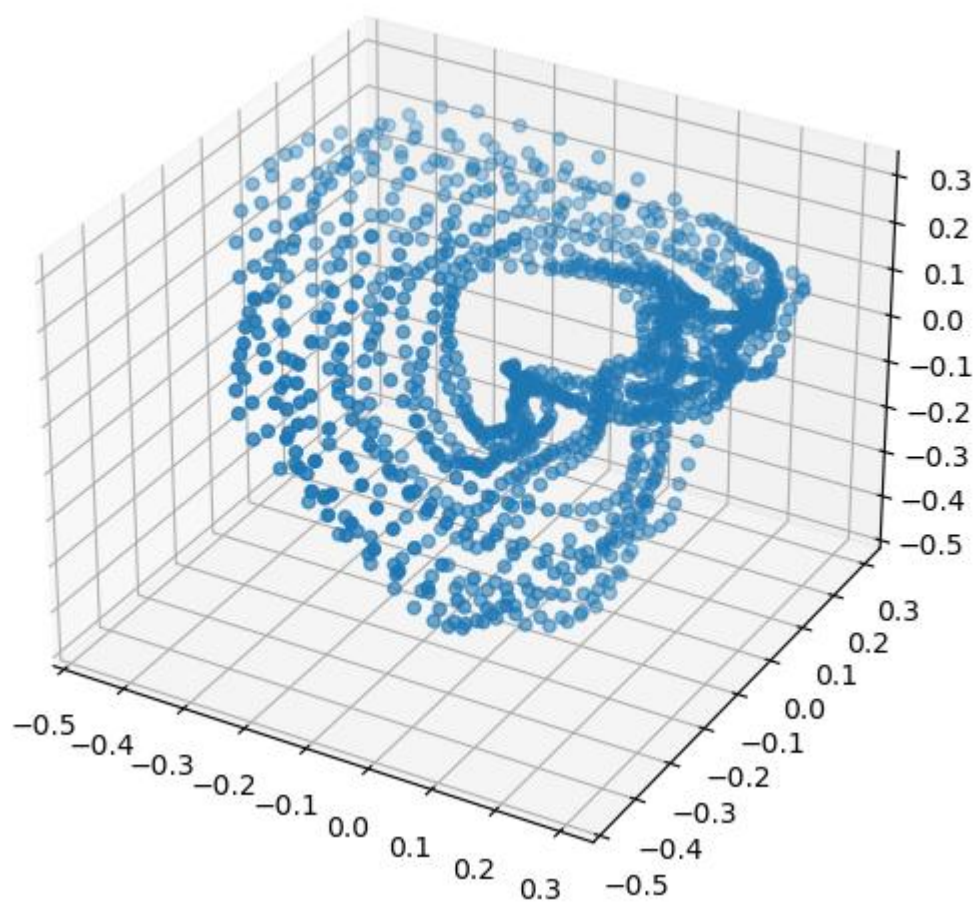
```
ax=Axes3D(fig)
```

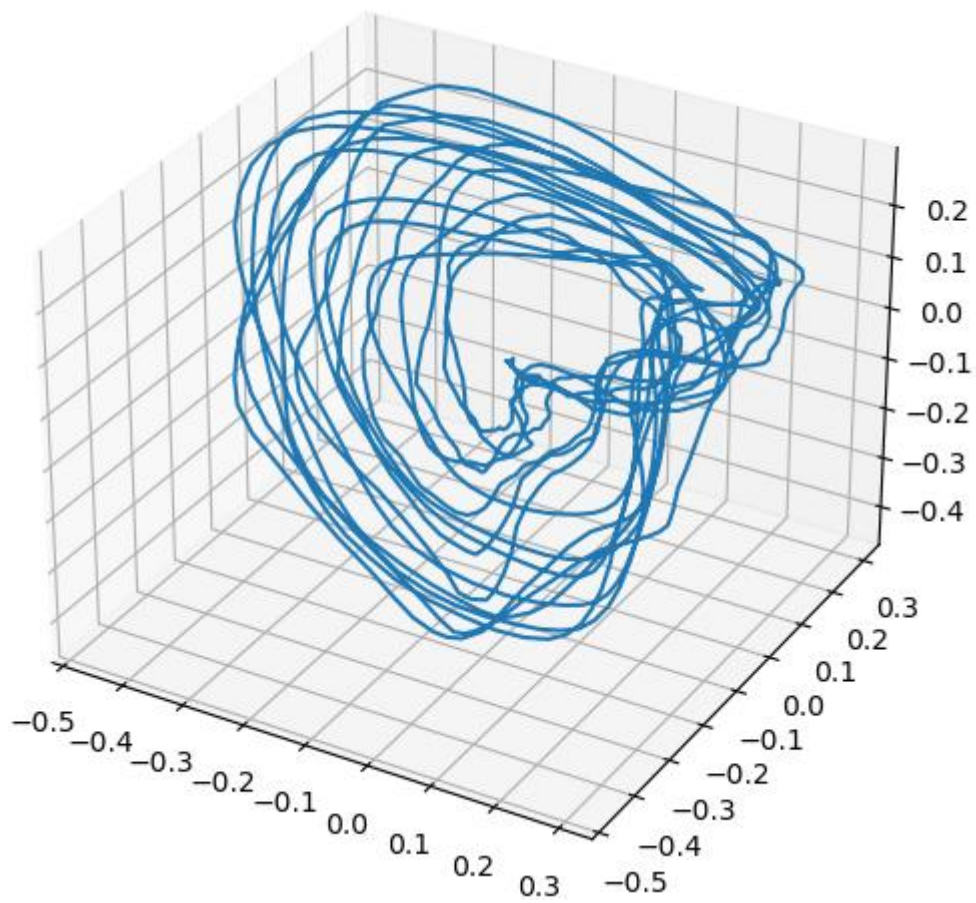
```
ax.scatter(Points[:,0],Points[:,1],Points[:,2])
```

```
fig=plt.figure()
```

```
ax=Axes3D(fig)
```

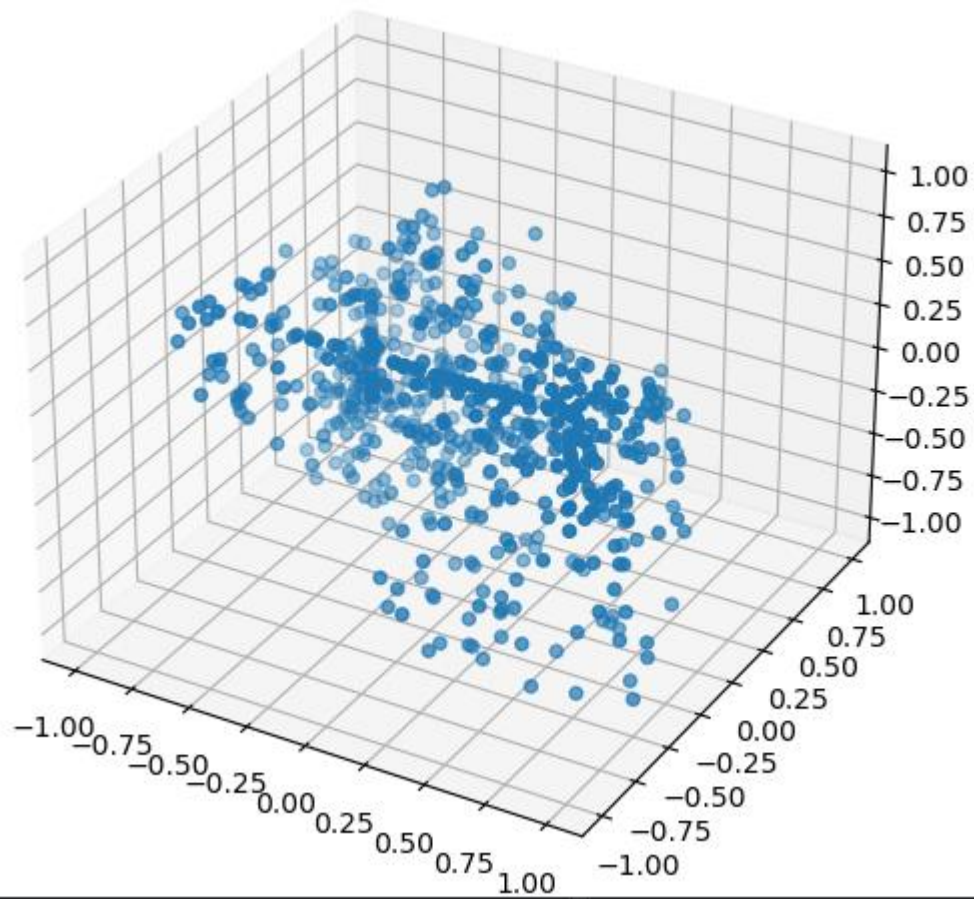
```
ax.plot(Points[:,0],Points[:,1],Points[:,2])
```

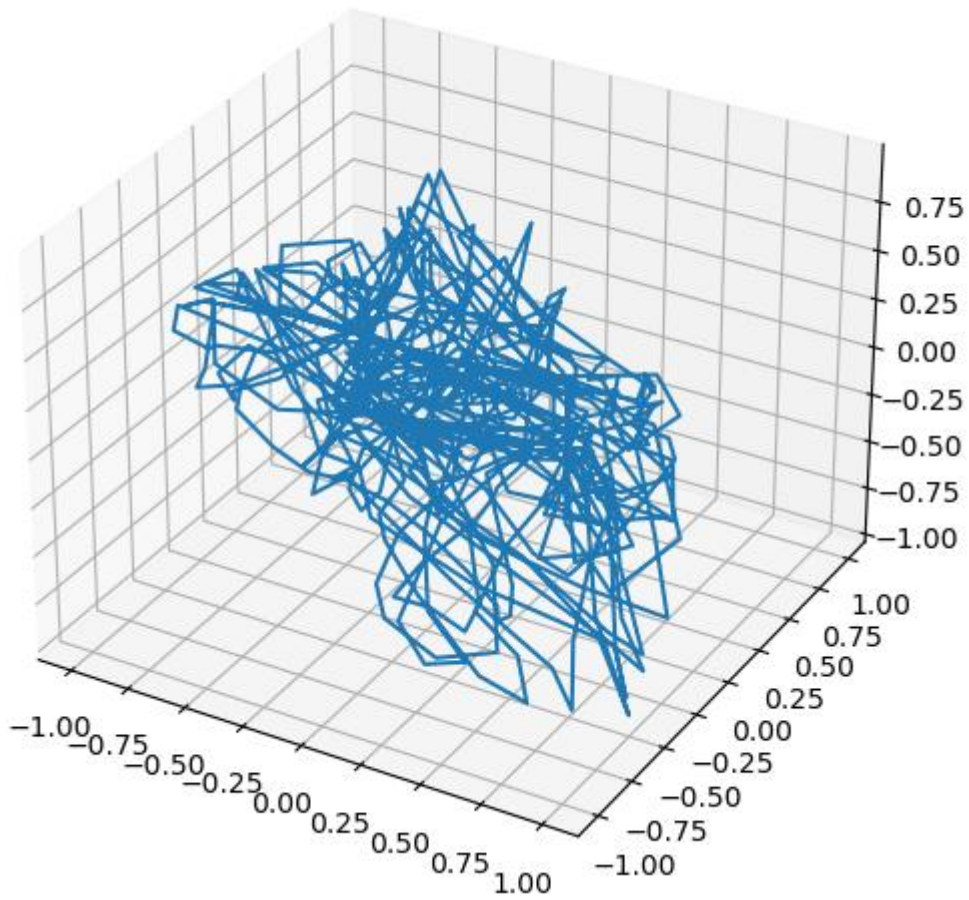




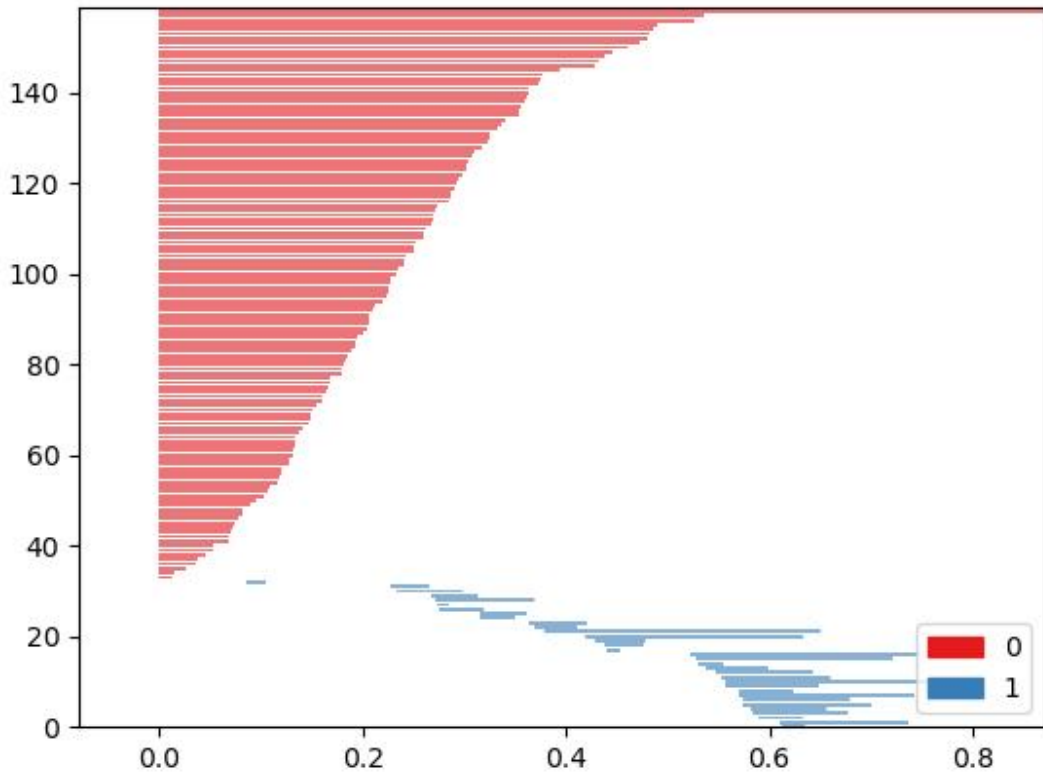
```
point_Cloud=timedelay.TimeDelayEmbedding(dim=3, delay=10, skip=1)
Points=point_Cloud(sig[5100:5800])
fig=plt.figure()
ax=Axes3D(fig)
ax.scatter(Points[:,0],Points[:,1],Points[:,2])
```

```
fig=plt.figure()
ax=Axes3D(fig)
ax.plot(Points[:,0],Points[:,1],Points[:,2])
```

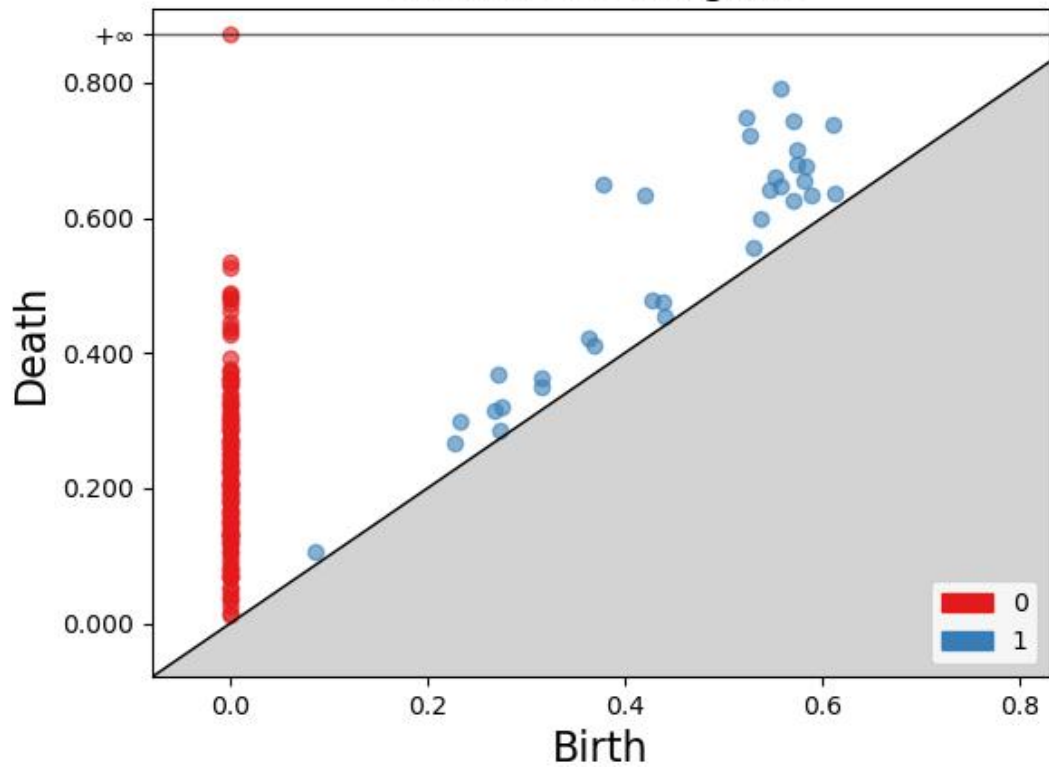




Persistence barcode



Persistence diagram



```

import soundfile as sf
import gudhi
from gudhi.point_cloud import timedelay
from scipy.io import wavfile
import numpy as np
import pylab as plt
import statsmodels.tsa.api as sm
from mpl_toolkits.mplot3d import Axes3D

```

```

def Read_Audio(Str):
    sig, samplerate = sf.read(Str)

    f1 = plt.figure(1)
    plt.plot(sig)

    return sig, samplerate

```

```

def Persistence_Homology(Data, Type_Complex='VR', Max_edge_length=1,
Max_dimension=3):
    if Type_Complex=='VR':
        rips_complex = gudhi.RipsComplex(points=Data,
max_edge_length=Max_edge_length)
        simplex_tree = rips_complex.create_simplex_tree(max_dimension=Max_dimension)
        diag = simplex_tree.persistence(min_persistence=0.01)
        gudhi.plot_persistence_barcode(diag, legend=True)
        gudhi.plot_persistence_diagram(diag, legend=True)
        plt.show()

```

```

def MainFunction(Delay=1, Skip=1, Dim=1, Max_edge_length=1, Max_dimension=1):
    Str='D:\单元音\前元音\[æ]_1.WAV'
    sig, samplerate=Read_Audio(Str)
    point_Cloud=timedelay.TimeDelayEmbedding(dim=Dim, delay=Delay, skip=Skip)
    Points=point_Cloud(sig[3500:5000])
    Persistence_Homology(Points, 'VR', Max_edge_length, Max_dimension)

```