Untitled

October 7, 2024

[1]:
```
import gudhi as gd
import numpy as np
import matplotlib.pyplot as plt
```

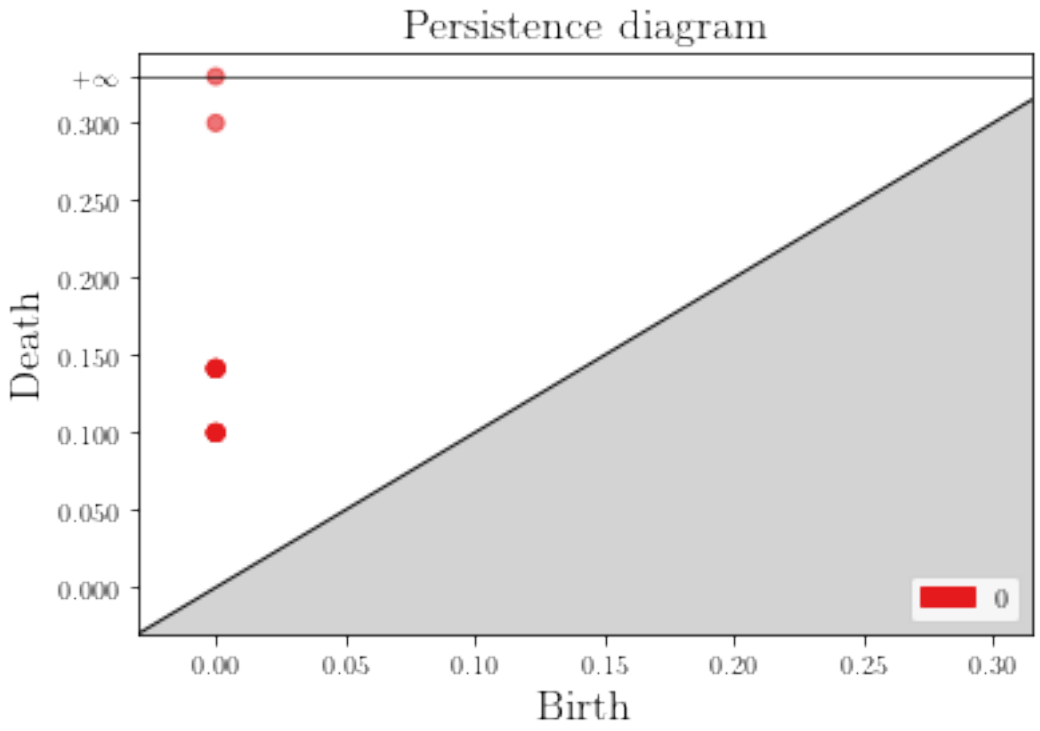
1 Ex 4

[2]:
```
V0 = [[-0.3, i/10.0] for i in range(11)]
V0.extend([i/10.0, i/10.0] for i in range(11))
```

[3]:
```
VR = gd.RipsComplex(points=V0)
ST = VR.create_simplex_tree(max_dimension=1)
ST.collapse_edges(nb_iterations=10)
ST.expansion(2)
diag = ST.persistence()
```

[4]:
```
gd.plot_persistence_diagram(diag)
```

[4]:
```
<AxesSubplot:title={'center':'Persistence diagram'}, xlabel='Birth',
ylabel='Death'>
```



2 Ex 5

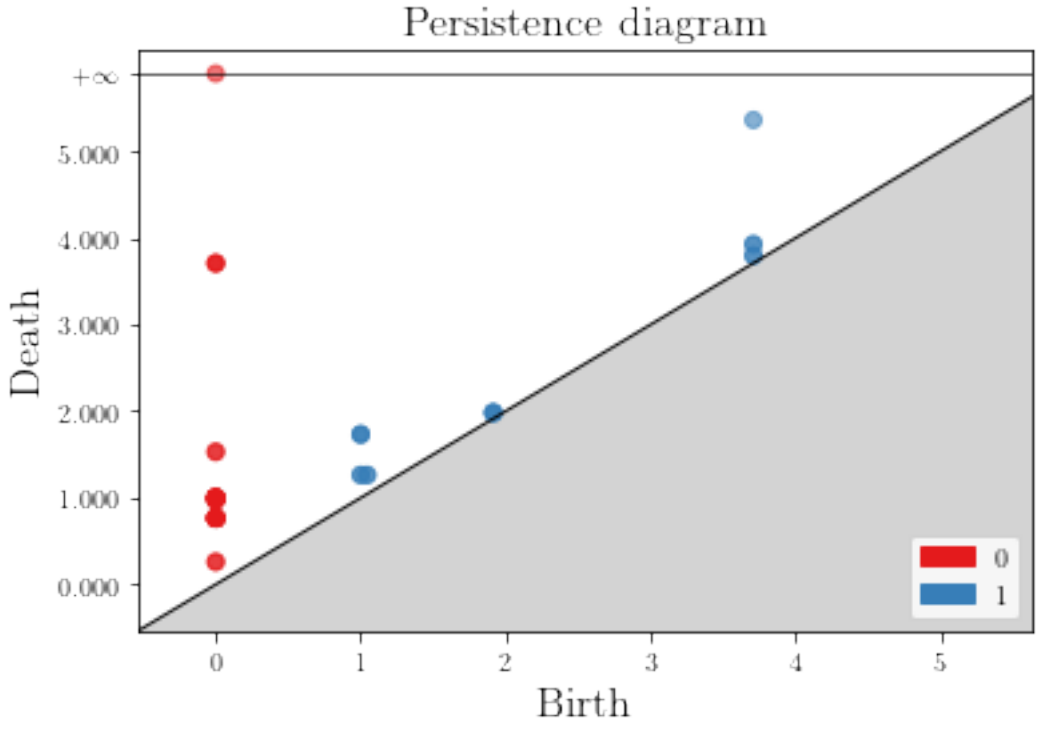
[5]:
```
T2 = [
    [(3 + np.cos(i))*np.cos(j),
    (3 + np.cos(i))*np.sin(j),
    np.sin(i)]
    for i in np.linspace(0, 2*np.pi, num=6, endpoint=False)
    for j in np.linspace(-np.pi/4, np.pi/4, num=3, endpoint=True)
]
T2.extend(
    [(3 + np.cos(i))*np.cos(j),
    (3 + np.cos(i))*np.sin(j),
    np.sin(i)]
    for i in np.linspace(-np.pi/4, np.pi/4, num=3, endpoint=True)
    for j in np.linspace(0, 2*np.pi, num=6, endpoint=False)
)
T2 = np.array(T2)
```

[6]:
```
VR = gd.RipsComplex(points=T2)
ST = VR.create_simplex_tree(max_dimension=1)
ST.collapse_edges(nb_iterations=10)
ST.expansion(2)
```

```
diag = ST.persistence()
```

[7]:
```
gd.plot_persistence_diagram(diag)
```

[7]:
```
<AxesSubplot:title={'center':'Persistence diagram'}, xlabel='Birth',
ylabel='Death'>
```



3 Ex 6

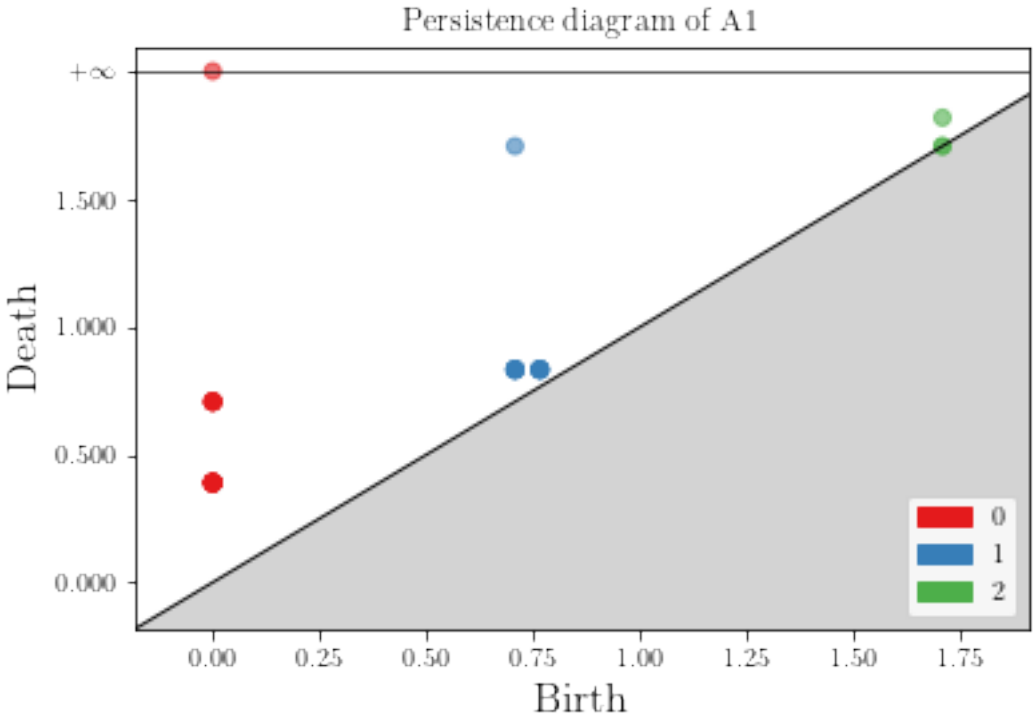
[8]:
```
D1 = [
    [0,0,1]
]
D1.extend(
    [np.cos(3*np.pi/8)*np.cos(j),
     np.cos(3*np.pi/8)*np.sin(j),
     np.sin(3*np.pi/8)]
    for j in np.linspace(-np.pi, np.pi, num=6, endpoint=False)
)

D2 = [
    [0,1,0]
]
```

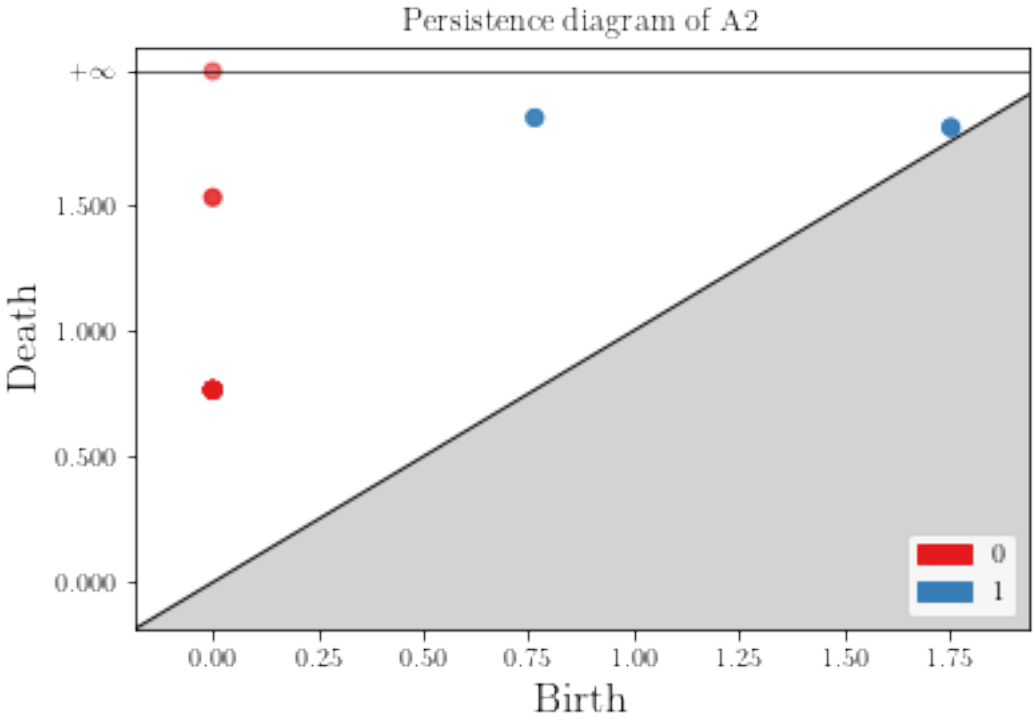
```
D2.extend(
    [np.cos(i*np.pi/9) * np.sin(j),
     np.sin(i*np.pi/9),
     np.cos(i*np.pi/9)*np.cos(j)]
    for i in [3,4]
    for j in np.linspace(-np.pi, np.pi, num=6, endpoint=False)
)
A1 = [
    [np.cos(i)*np.cos(j),
     np.cos(i)*np.sin(j),
     np.sin(i)]
    for i in np.linspace(-np.pi/8, np.pi/8, num=3, endpoint=True)
    for j in np.linspace(-np.pi, np.pi, num=8, endpoint=False)
]
A2 = [
    [(3+np.cos(i))*np.cos(j),
     (3+np.cos(i))*np.sin(j),
     np.sin(i)]
    for i in np.linspace(-np.pi, np.pi, num=8, endpoint=False)
    for j in np.linspace(-np.pi/4, np.pi/4, num=3, endpoint=True)
]
```

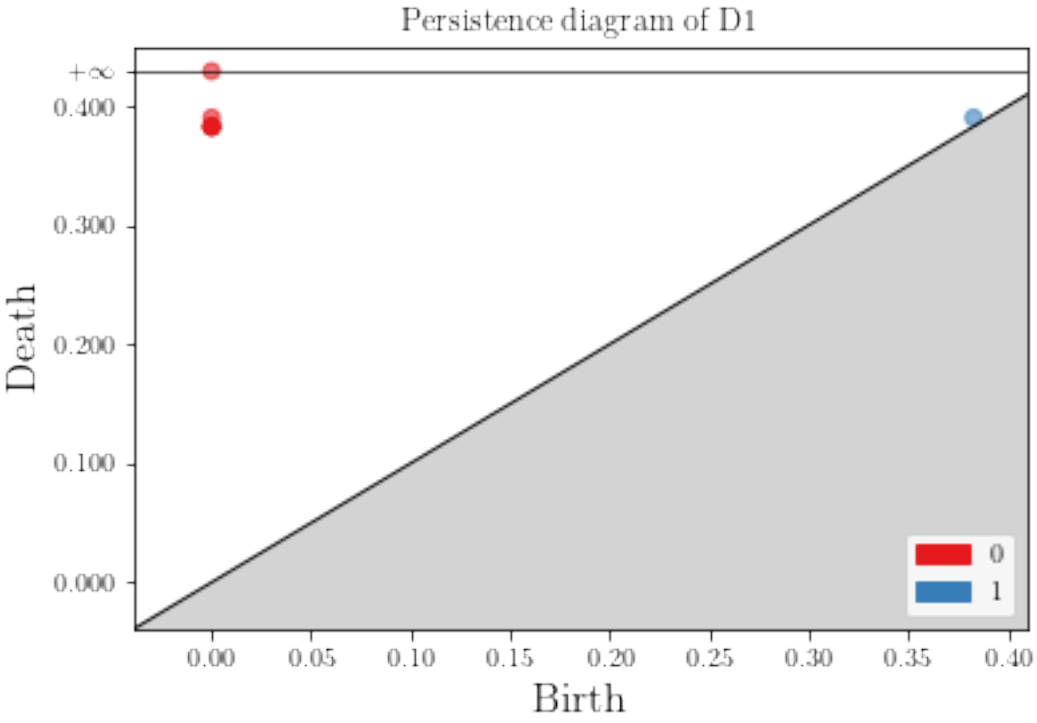
[9]:
```
def compute_everything(points, maxdim=3):
    VR = gd.RipsComplex(points=points)
    ST = VR.create_simplex_tree(max_dimension=1)
    ST.collapse_edges(nb_iterations=10)
    ST.expansion(maxdim)
    diag = ST.persistence()

    return VR, ST, diag
```

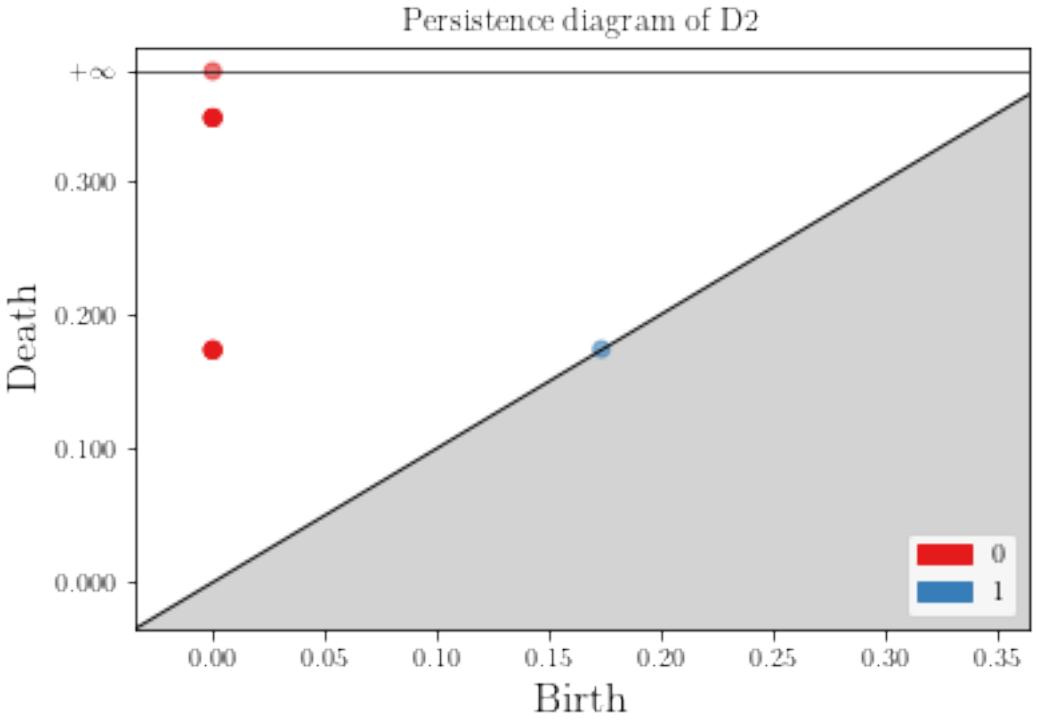
[10]:
```
VR_D1, ST_D1, diag_D1 = compute_everything(D1)
VR_D2, ST_D2, diag_D2 = compute_everything(D2)
VR_A1, ST_A1, diag_A1 = compute_everything(A1)
VR_A2, ST_A2, diag_A2 = compute_everything(A2)
```

[11]:
```
for d, label in [(diag_A1, "A1"), (diag_A2, "A2"), (diag_D1, "D1"), (diag_D2,␣
 ↪"D2")]:
    gd.plot_persistence_diagram(d)
    plt.title(f"Persistence diagram of {label}")
```


Persistence diagram of A1
+∞
1.500
1.000
0.500
0.000
Death
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Birth
0
1
2


Persistence diagram of A2
+∞
1.500
1.000
0.500
0.000
Death
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Birth
0
1


Persistence diagram of D1
+∞
0.400
0.300
0.200
0.100
0.000
Death
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Birth
0
1


Persistence diagram of D2
+∞
0.300
0.200
0.100
0.000
Death
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Birth
0
1

4 Ex 7, 8 , 9 & 10

[12]:
```
def extract_diagram(diagrams, dimension):
    return [x for d,x in diagrams if d==dimension]
```

[20]:
```

```

[20]:
```
array([[0.76536686, 0.83272867],
       [0.76536686, 0.83272867],
       [0.70710678, 0.83272867],
       [0.70710678, 0.83272867],
       [0.76536686, 0.83272867],
       [0.76536686, 0.83272867],
       [0.76536686, 0.83272867],
       [0.70710678, 0.83272867],
       [0.70710678, 0.83272867],
       [0.70710678, 0.83272867],
       [0.76536686, 0.83272867],
       [0.76536686, 0.83272867],
       [0.70710678, 0.83272867],
       [0.70710678, 0.83272867],
       [0.76536686, 0.83272867],
       [0.70710678, 0.83272867],
       [0.70710678, 1.70710678]])
```

[22]:
```
# D1 & D2
gd.bottleneck_distance(ST_D1.persistence_intervals_in_dimension(1),
                       ST_D2.persistence_intervals_in_dimension(1))
```

[22]:
```
0.003748605833583274
```

[21]:
```
# A1 & A2
gd.bottleneck_distance(ST_A1.persistence_intervals_in_dimension(1),
                       ST_A2.persistence_intervals_in_dimension(1))
```

[21]:
```
0.5411961001461969
```

[23]:
```
# D1 & A1
gd.bottleneck_distance(ST_D1.persistence_intervals_in_dimension(1),
                       ST_A1.persistence_intervals_in_dimension(1))
```

[23]:
```
0.4999999999999999
```

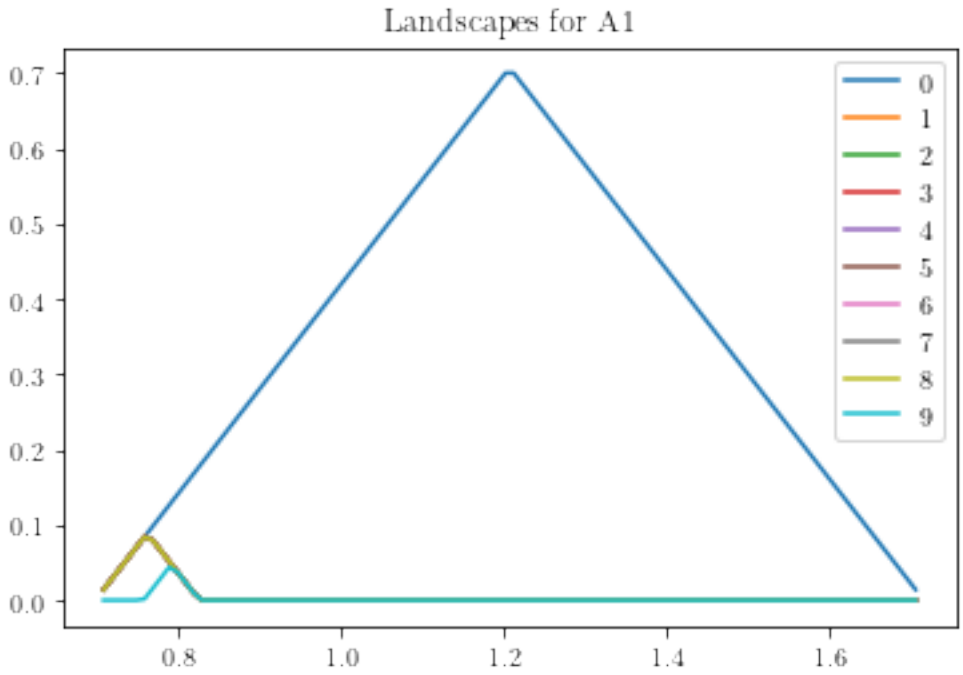
[25]:
```
# D1 & A2
gd.bottleneck_distance(ST_D1.persistence_intervals_in_dimension(1),
                       ST_A2.persistence_intervals_in_dimension(1))
```

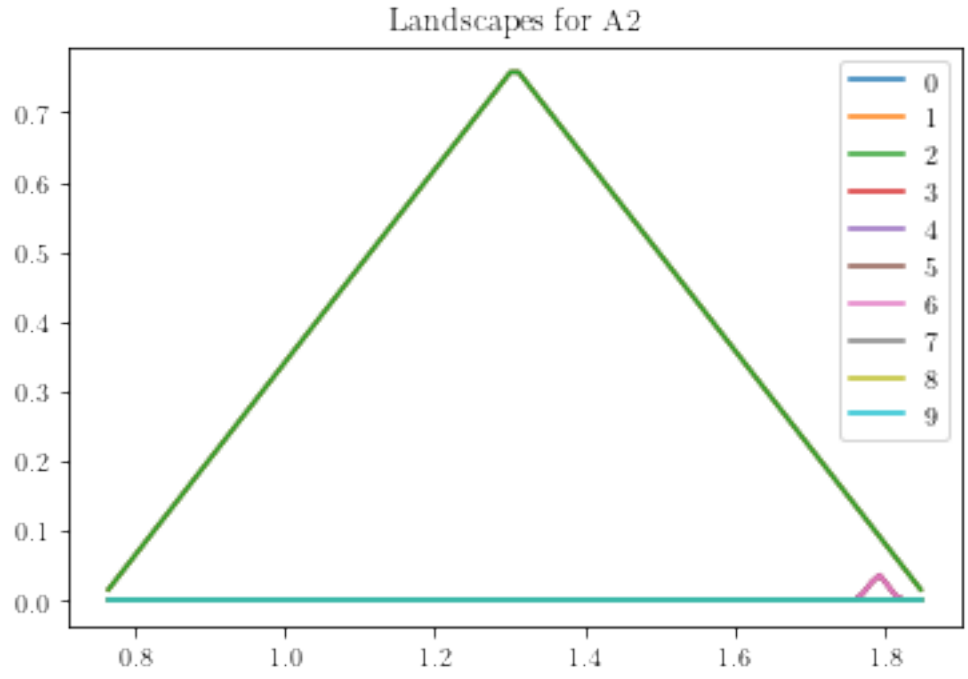
[25]: 0.5411961001461969

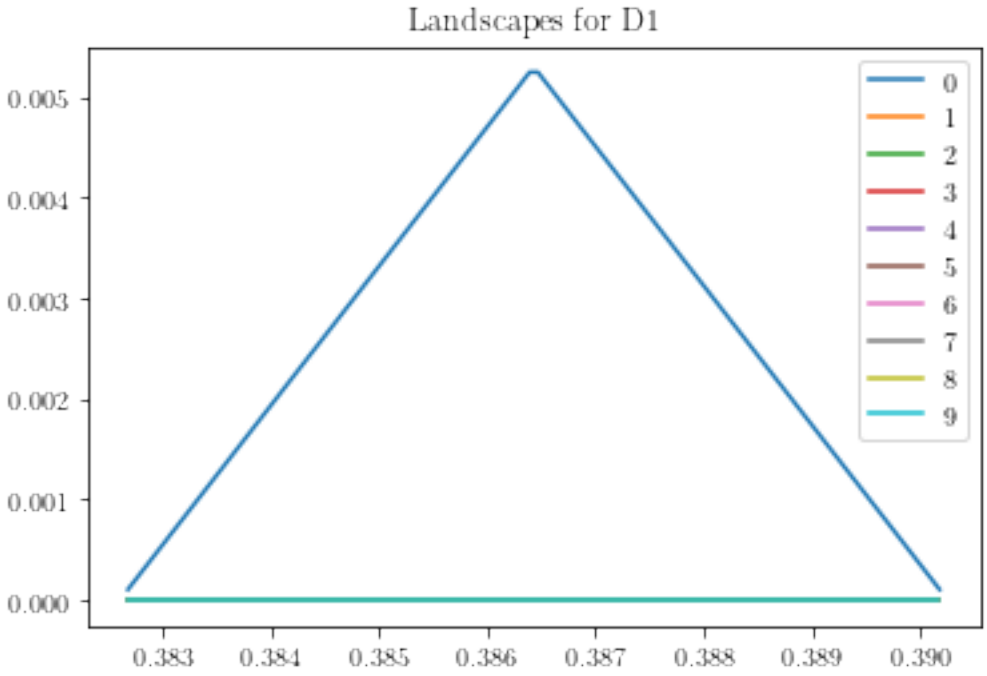
5 Ex 12

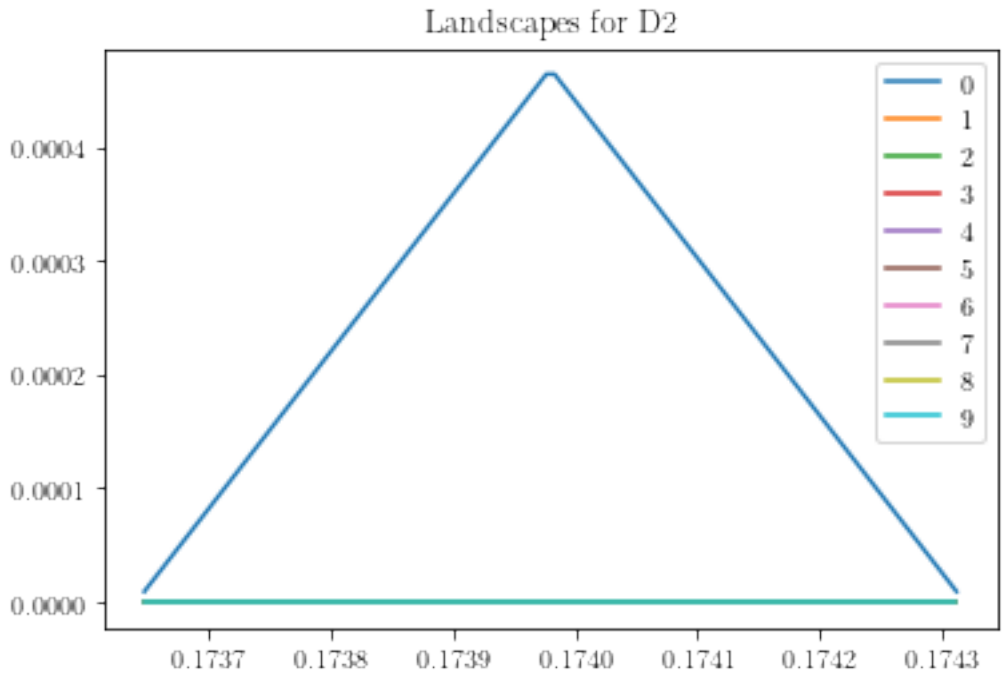
[19]:
```
from gudhi.representations import Landscape
```

[66]:
```
for d, label in [(ST_A1, "A1"), (ST_A2, "A2"), (ST_D1, "D1"), (ST_D2, "D2")]:
    N = 10
    res = 100
    LS = Landscape(resolution=res, num_landscapes=N)
    l = LS.fit_transform([d.persistence_intervals_in_dimension(1)])[0]
    for i in range(N):
        plt.plot(np.linspace(*LS.sample_range_fixed_, res), l[res*i:res*(i+1)],␣
↪label=i)
    plt.legend()
    plt.title(f"Landscapes for {label}")
    plt.show()
```




Landscapes for A2
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.8
1.0
1.2
1.4
1.6
1.8
0
1
2
3
4
5
6
7
8
9


Landscapes for D1
0.005
0.004
0.003
0.002
0.001
0.000
0.383
0.384
0.385
0.386
0.387
0.388
0.389
0.390
0
1
2
3
4
5
6
7
8
9


Landscapes for D2
0.0004
0.0003
0.0002
0.0001
0.0000
0.1737
0.1738
0.1739
0.1740
0.1741
0.1742
0.1743
0
1
2
3
4
5
6
7
8
9

[ ]: