List_5_tasks

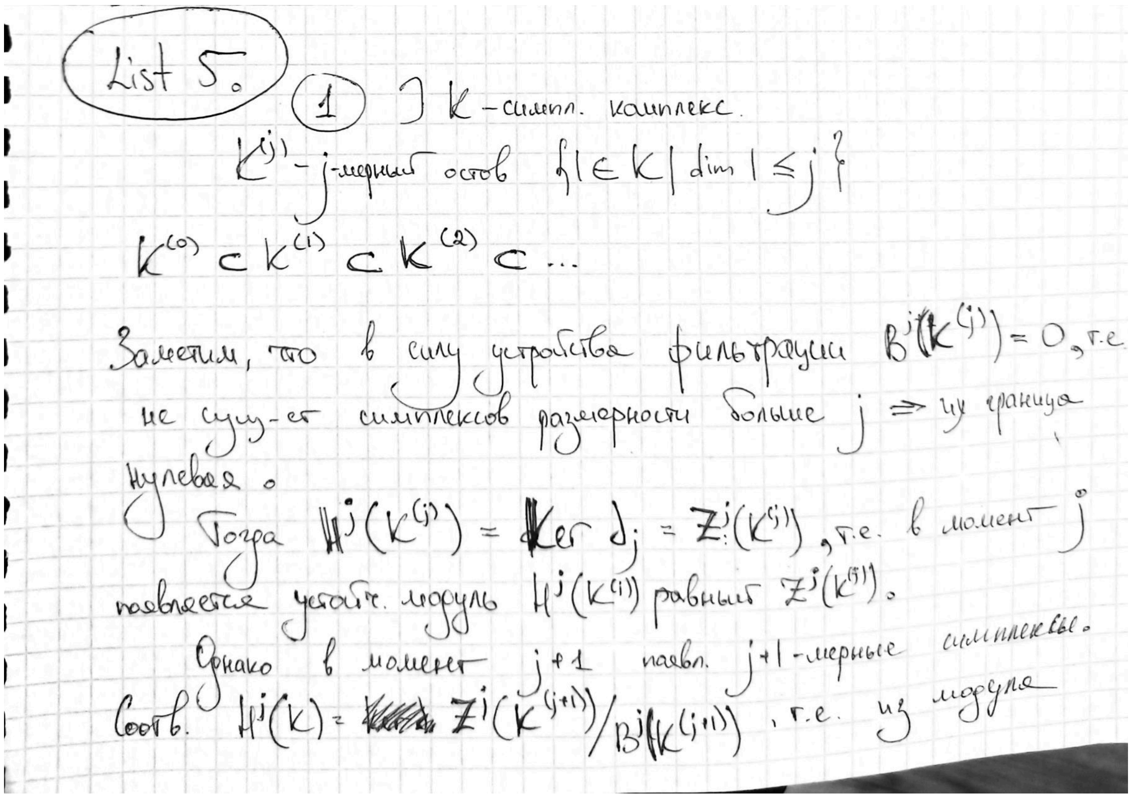
March 10, 2024

[112]:
```python
from PIL import Image
import gudhi as g
import numpy as np
```

0.0.1 Task 1

[2]:
```python
Image.open("1.jpg")
```

[2]:



List 5.

1] K – симпл. комплекс.

$K^{(j)}$ – j-мерный остов $\{I \in K \mid \dim I \leq j\}$

$K^{(0)} \subset K^{(1)} \subset K^{(2)} \subset \ldots$

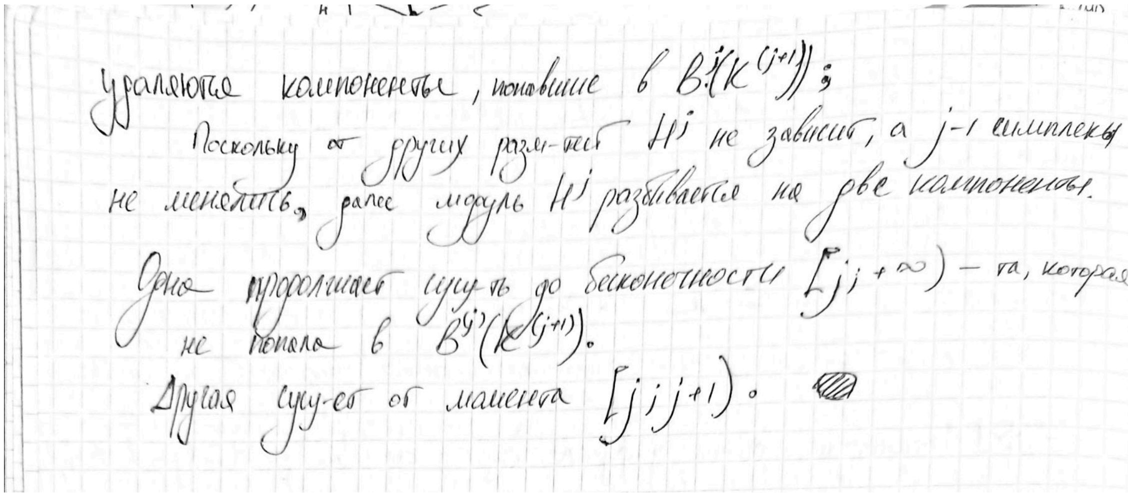
Заметим, что в силу устройства фильтрации $B^j(K^{(j)}) = 0$, т.е. не сущ-ет симплексов размерности больше j $\Rightarrow$ их граница нулевая.

Тогда $H^j(K^{(j)}) = \operatorname{Ker} \partial_j = Z^j(K^{(j)})$, т.е. в момент j появляется устойч. модуль $H^j(K^{(j)})$ равный $Z^j(K^{(j)})$.

Однако в момент $j+1$ появл. $j+1$-мерные симплексы. Соотв. $H^j(K) = Z^j(K^{(j+1)}) / B^j(K^{(j+1)})$, т.е. из модуля

[3]:
```python
Image.open("2.jpg")
```

[3]:



Удаляются компоненты, попавшие в $B^{j}(K^{(j+1)})$;

Поскольку от других разм-тей H^j не зависит, а $j-1$ симплексы не меняются, далее модуль H^j разбивается на две компоненты.

Одна продолжает сущ-ть до бесконечности $[j; +\infty)$ – та, которая не попала в $B^{(j)}(K^{(j+1)})$.

Другая сущ-ет от момента $[j; j+1)$.

```
[ ]:
```

0.0.2 Task 2

```
[94]: filtration = [
          ([1], 0),
          ([2], 0),
          ([3], 0),
          ([1, 2], 4),
          ([2, 3], 4),
          ([4], 4),
          ([1, 4], 5),
          ([3, 4], 5),
          ([1, 3], 7),
          ([2, 4], 10),
          ([1, 2, 3], 16),
          ([1, 2, 4], 16),
          ([1, 3, 4], 16),
          ([2, 3, 4], 20),
          ([1, 2, 3, 4], 23)
      ]
```

```
[95]: simplices, times = zip(*filtration)
```

```
[179]: simplices
```

```
[179]: ([1],
        [2],
        [3],
        [1, 2],
        [2, 3],
        [4],
        [1, 4],
        [3, 4],
        [1, 3],
        [2, 4],
        [1, 2, 3],
        [1, 2, 4],
        [1, 3, 4],
        [2, 3, 4],
        [1, 2, 3, 4])
```

[239]:
```
matrix = np.matrix(np.zeros((len(simplices), len(simplices))), dtype=np.byte)
```

[241]:
```
for i in range(len(simplices)):
    for j in range(i, len(simplices)):
        x = set(simplices[i])
        y = set(simplices[j])
        if len(y) == len(x) + 1 and x.issubset(y):
            matrix[i, j] = 1
```

[242]:
```
matrix
```

[242]:
```
matrix([[0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
        [0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], dtype=int8)
```

[246]:
```
for i in range(matrix.shape[0]):
    nonzero_col = np.flatnonzero(matrix[:, i])
    while len(nonzero_col) != 0:
        flag = False
        for j in range(i):
            if matrix[nonzero_col[-1], j] == matrix[nonzero_col[-1], i]:
                matrix[:, i] = np.abs(matrix[:, i] - matrix[:, j])
                nonzero_col = np.flatnonzero(matrix[:, i])
                break
            elif j == i-1:
                flag = True
        if flag:
            break
```

[247]:
```
matrix
```

[247]:
```
matrix([[0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], dtype=int8)
```

0.0.3 Homologies

[258]:
```
print(simplices[0], "inf")
for i in range(matrix.shape[1]):
    death = np.flatnonzero(matrix[:, i])
    if len(death) == 0:
        continue
    else:
        print(simplices[death[-1]], simplices[i], times[death[-1]], times[i])
```

```
[1] inf
[2] [1, 2] 0 4
[3] [2, 3] 0 4
[4] [1, 4] 4 5
[1, 3] [1, 2, 3] 7 16
[2, 4] [1, 2, 4] 10 16
[3, 4] [1, 3, 4] 5 16
[2, 3, 4] [1, 2, 3, 4] 20 23
```

[ ]:
```

```

0.0.4 Simplices birth and death

[254]:
```
for i in range(matrix.shape[0]):
    death = np.flatnonzero(matrix[i, :])
    if len(death) == 0:
        print(simplices[i], "inf")
    else:
        print(simplices[i], simplices[death[0]], times[i], times[death[0]])
```

```
[1] [1, 2] 0 4
[2] [1, 2] 0 4
[3] [2, 3] 0 4
[1, 2] [1, 2, 3] 4 16
[2, 3] [1, 2, 3] 4 16
[4] [1, 4] 4 5
[1, 4] [1, 2, 4] 5 16
[3, 4] [1, 3, 4] 5 16
[1, 3] [1, 2, 3] 7 16
[2, 4] [1, 2, 4] 10 16
[1, 2, 3] [1, 2, 3, 4] 16 23
[1, 2, 4] [1, 2, 3, 4] 16 23
[1, 3, 4] [1, 2, 3, 4] 16 23
[2, 3, 4] [1, 2, 3, 4] 20 23
[1, 2, 3, 4] inf
```

0.0.5 Task 3

[ ]:
```

```

[ ]:
```

```

0.0.6 Task 4

[51]:
```
tree = g.SimplexTree()
```

[52]:
```
simplices = [
    ([1], 0),
    ([2], 0),
    ([3], 0),
    ([1, 2], 2),
    ([4], 2.5),
    ([5], 3),
    ([2, 3], 3),
    ([3, 4], 3.7),
    ([1, 4], 4),
    ([1, 5], 4.3),
    ([4, 5], 5),
    ([3, 5], 7.9),
    ([6], 8),
    ([1, 6], 9),
    ([2, 6], 9.3),
    ([3, 6], 9),
    ([2, 5], 10.2),
    ([4, 6], 12)
]
```

[54]:
```
for simplex in simplices:
    simplex, filt_val = simplex
    tree.insert(simplex, filt_val)
```

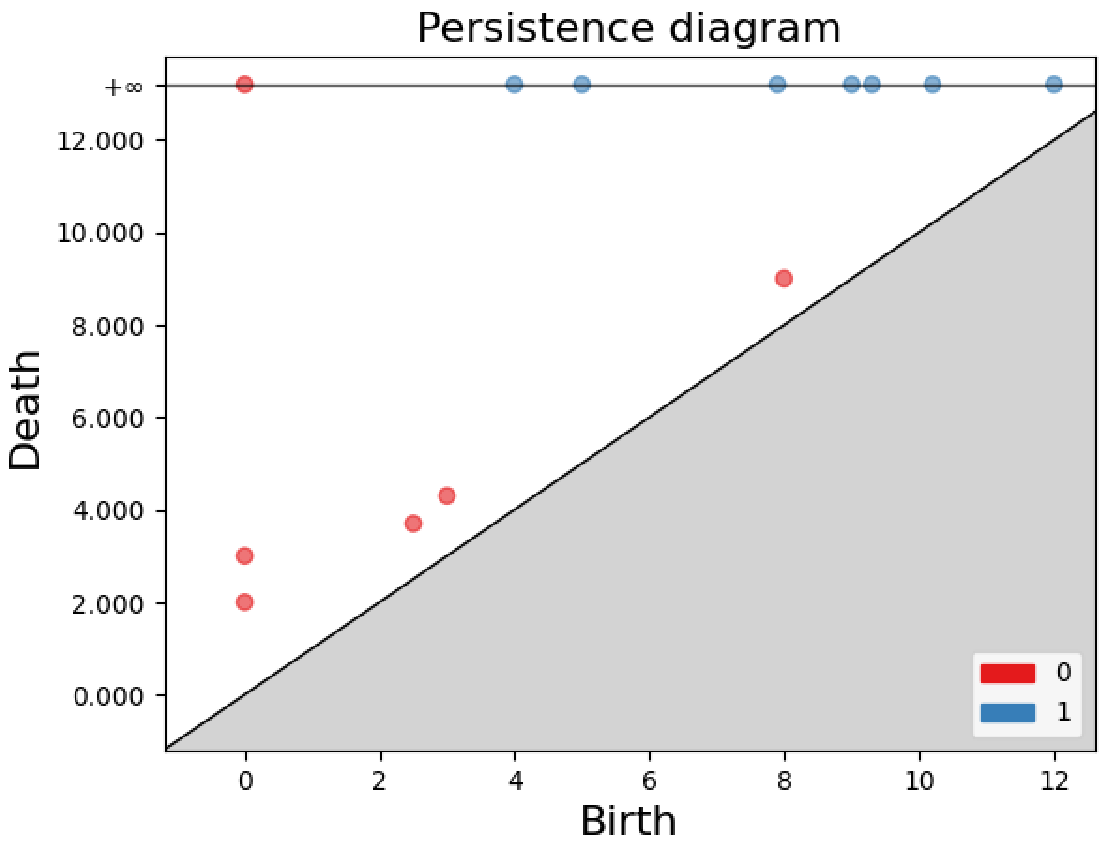
[ ]:

[91]:
```
tree.compute_persistence()
```

[84]:
```
persist_tree = tree.persistence(persistence_dim_max=True)
```

[85]:
```
g.persistence_graphical_tools.plot_persistence_diagram(persist_tree)
```

[85]:
```
<Axes: title={'center': 'Persistence diagram'}, xlabel='Birth', ylabel='Death'>
```


Persistence diagram
+∞
Death
Birth