Boundary Matrix

March 22, 2022

In this notebook, we compute the rank of the boundary matrix examples in the paper, as well as the reduced form of the full boundary matrix.

[71]:
```
from sage.all import GF, matrix, zero_matrix, copy

# We construct the boundary matrices from the paper
B1 = matrix(GF(2), [[1, 1, 0, 0, 0, 0, 0, 0, 0],
                    [1, 0, 1, 1, 1, 0, 0, 0, 0],
                    [0, 1, 1, 0, 0, 1, 1, 0, 0],
                    [0, 0, 0, 1, 0, 0, 0, 1, 0],
                    [0, 0, 0, 0, 1, 1, 0, 1, 1],
                    [0, 0, 0, 0, 0, 0, 1, 0, 1]])

B2 = matrix(GF(2), [[1, 0, 0],
                    [1, 0, 0],
                    [1, 0, 0],
                    [0, 1, 0],
                    [0, 1, 0],
                    [0, 0, 1],
                    [0, 0, 1],
                    [0, 1, 0],
                    [0, 0, 1]])

B = zero_matrix(GF(2), 6, 6).augment(B1).augment(zero_matrix(GF(2), 6, 3)).\
        stack(zero_matrix(GF(2), 9, 15).augment(B2)).stack(zero_matrix(GF(2),␣
 ↪3, 18))
```

[72]:
```
# We compute the ranks of the individual boundary matrices B1 and B2
print("Rank B1: " + str(B1.rank()))
print("Rank B2: " + str(B2.rank()))
```

```
Rank B1: 5
Rank B2: 3
```

[87]:
```
# Get the lowest nonzero entry of a column, return -1 if undefined
def low(j, B):
    lowj = -1
    for i in range(B.nrows())[::-1]:
```

```
        if B[i, j] != 0:
            lowj = i
            break
    return lowj

# Reduce a boundary matrix B
def reduce(B):
    for j in range(B.ncols()):
        repeat = True
        while(repeat):
            repeat = False
            for i in range(j):
                if low(i, B) == low(j, B):
                    for k in range(B.nrows()):
                        B[k, j] = B[k, j] + B[k, i]
                    if low(j, B) != -1:
                        repeat = True
    return(B)

# We compute the reduced form of the full boundary matrix B over GF(2)
reduce(copy(B))
```

[87]:
```
[0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 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 1 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 0 0 0 1 0 0 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 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 0 0 0 1 0 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 0 0 0 1 0]
[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 0 0 0 1]
[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 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

[ ]:

[ ]: