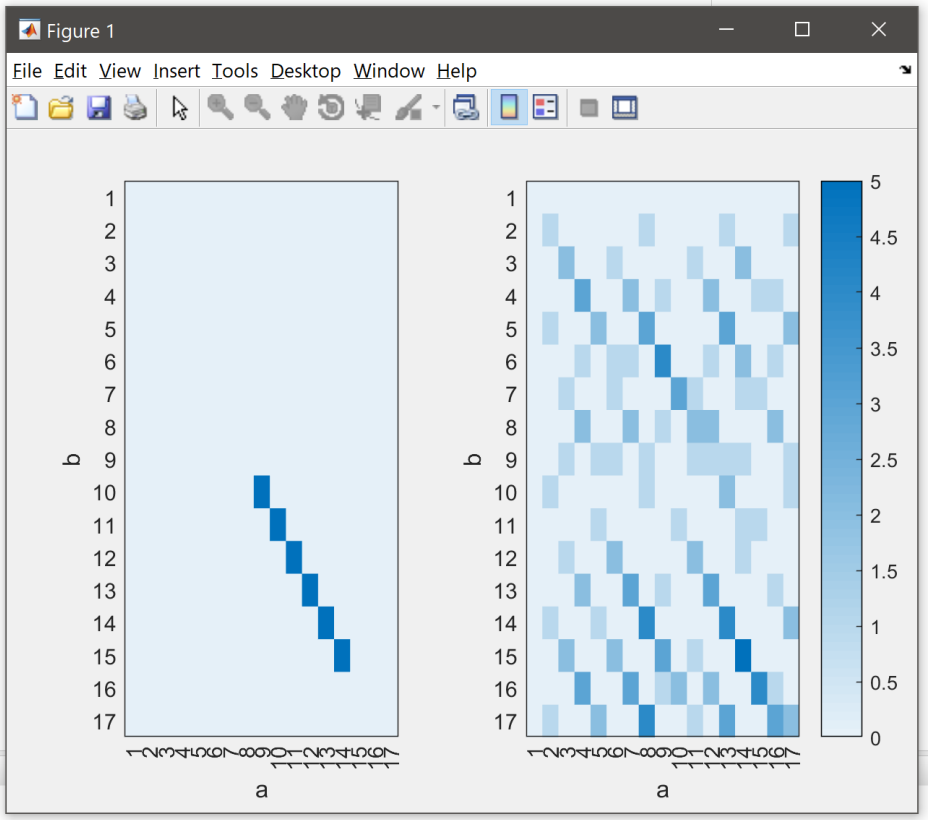
Zadanie 3

Wojciech Miśta, 236453

Algorytmy znajdują się w osobnych plikach w repozytorium.

Działanie:

```
Gap: -2
ID: a
ATCACTCTAGTCATCA
ID: b
ATCGTCACGTCATGCA


AGTCAT
CGTCAT

***
```




```
mmand Window
ans =

  1×2 table

    seq1       seq2
    _____      _____

    AGTCAT     CGTCAT
```

Analiza złożoności obliczeniowej:

findMatch.m

```
function point = findMatch(substitutionMatrix,nuclOne,nuclTwo)

vecRow = substitutionMatrix(1,:);
vecCol = substitutionMatrix(:,1);

x = char(vecRow) == nuclOne;
y = char(vecCol) == nuclTwo;

point = str2double(char(substitutionMatrix(x,y)));

end
```

p – rozmiar zmiennej 'substitutionMatrix'

Złożoność czasowa: O(p)

Złożoność przestrzenna: $O(p^2)$

findCooridnates.m

```
function [XCor,YCor] = findMaxCoordinates(scoredMatrix,maxValue)
%Finds coordinates of the max value and stores it in two separate matrices.

XCorLength = length(scoredMatrix(:,1)); %substitution +1
YCorLength = length(scoredMatrix(1,:)); %substitution +1

XCor = []; %substitution +1
YCor = []; %substitution +1

%find occurances of the max value
for m = 1:XCorLength %incrementation, checking condition, n*x; 2 + n*x
    for n = 1:YCorLength %incrementation, checking condition, m*x; 2 + m*x
        if scoredMatrix(m,n) == maxValue %checking condition +1
            XCor(end+1) = m; %substitution +1
            YCor(end+1) = n; %substitution +1
        end
    end
end

end
```

x – rozmiar zmiennej 'XCor'

y – rozmiar zmiennej ‘YCor’

Złożoność czasowa: O(xy)

Złożoność przestrzenna: O(xy)

smithWaterman.m

```
%
seq1 = strcat('-',seq1);
seq2 = strcat('-',seq2);

length1 = length(seq1)
length2 = length(seq2)

outputSeq = zeros(length1,length2);
indexMatrix = zeros(length1,length2);

substitutionMatrix = getScoringMatrix('subMatrix2txt');

for m = 2:length(seq1)
    for n = 2:length(seq2)
        if (seq1(m) == seq2(n))
            value = findMatch(substitutionMatrix,seq1(m),seq2(n)) + outputSeq(m-1,n-1); %zwraca punkt z txt; substitutio
        else %is mismatch
            value = findMatch(substitutionMatrix,seq1(m),seq2(n)) + outputSeq(m-1,n-1);
        end

        % Mój cały problem z tym programem wynikał z tych dwóch linijek.
        % Pomyliłem kolejność column z row i odkrycie tego zajęło mi dobre
        % kilka godzin :(
        value2 = outputSeq(m-1,n) + gap;
        value3 = outputSeq(m,n-1) + gap;

        [maxVal,index] = max([value value2 value3 0]);
        outputSeq(m,n) = maxVal;
        indexMatrix(m,n) = index;
    end
end
end
```

x – rozmiar ‘outputSeq’

y – rozmiar ‘indexMatrix’

Złożoność czasowa: O(mn) ?

Złożoność przestrzenna: O(xy)

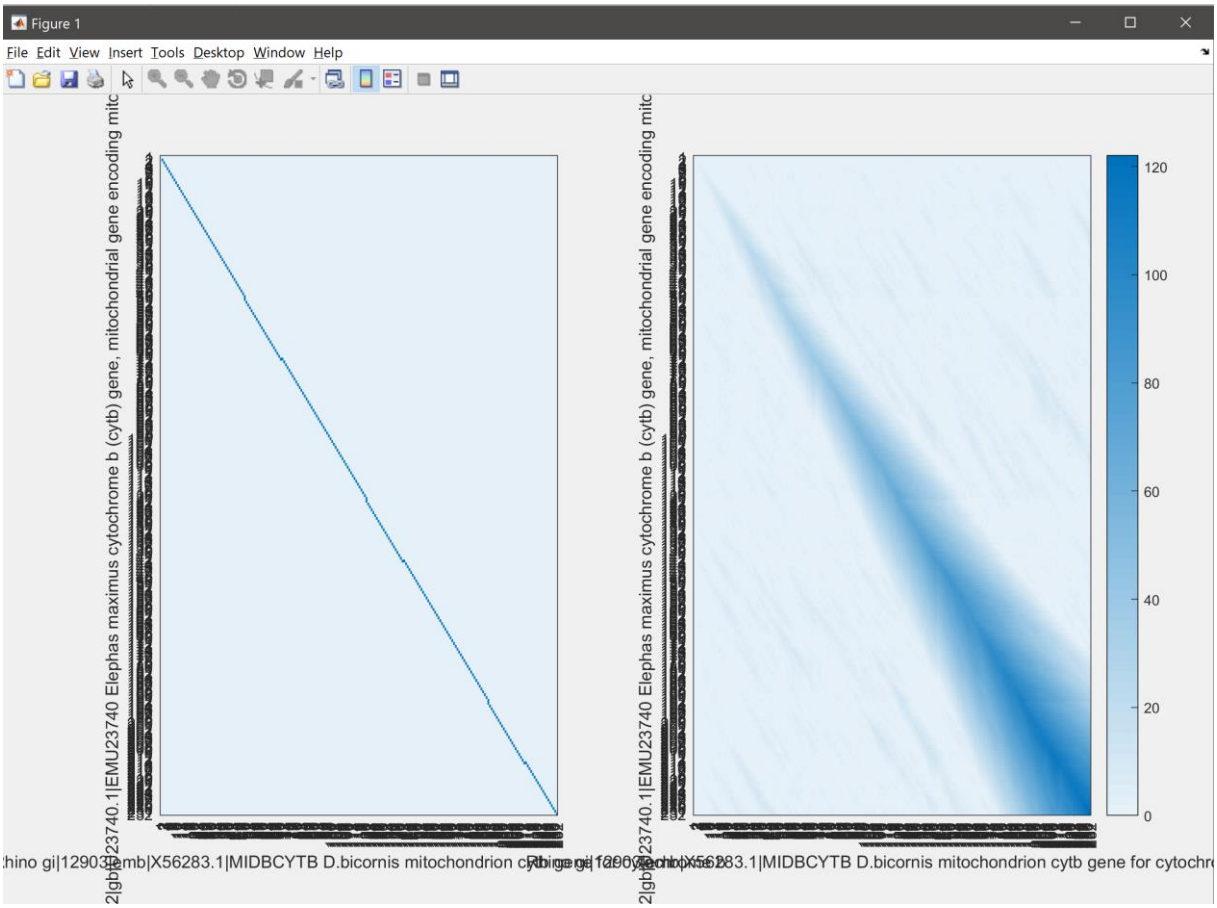
findPath.m

```
function [matrixPath,sequenceObject,matchCount,gapCount] = findPath(scoredMatrix,indexMatrix,XCor,YCor,seq1,seq2)
%FINDPATH Summary of this function goes here

matrixPath = zeros(length(scoredMatrix(:,1)),length(scoredMatrix(1,:)));
sequenceObject = struct;

for index = 1:length(XCor) %nie ma znaczenia czy X czy Y
    curRow = XCor(index);
    curCol = YCor(index);
    charArray1 = '';
    charArray2 = '';
    matchCount = 0;
    gapCount = 0;

    while scoredMatrix(curRow, curCol) ~= 0
        matrixPath(curRow, curCol) = 1;
        curIndex = indexMatrix(curRow, curCol);

        %curIndex - przejścia
        %if 1 - match/mismatch
        %if 2 - gap up
        %if 3 - gap left
        %if 4 - 0

        %scoreIndex - punktacja

        if(curIndex == 1)
            charArray1 = strcat(charArray1, seq1(curRow-1));
            charArray2 = strcat(charArray2, seq2(curCol-1));
            if(seq1(curRow-1) == seq2(curCol-1))
                matchCount = matchCount + 1;
            end
            curRow = curRow - 1;
            curCol = curCol - 1;

        elseif(curIndex == 2)
            charArray1 = strcat(charArray1, seq1(curRow-1));
            charArray2 = strcat(charArray2, '_');
            gapCount = gapCount + 1;
            %gap up
            curRow = curRow - 1;
        else
            charArray1 = strcat(charArray1, '_');
            charArray2 = strcat(charArray2, seq2(curCol-1));
            gapCount = gapCount + 1;
            %gap left
            curCol = curCol - 1;
        end
    end

    charArray1 = strcat(charArray1, seq1(curRow));
    charArray2 = strcat(charArray2, seq2(curCol));

    sequenceObject(index).seq1 = flip(charArray1(1:end-1));
    sequenceObject(index).seq2 = flip(charArray2(1:end-1));
    sequenceObject(index).matchCount = matchCount;
    sequenceObject(index).gapCount = gapCount;
end
end
```

Porównanie par sekwencji

Heatmap'a po stronie lewej ukazuje ścieżkę, a po prawej punktowanie.

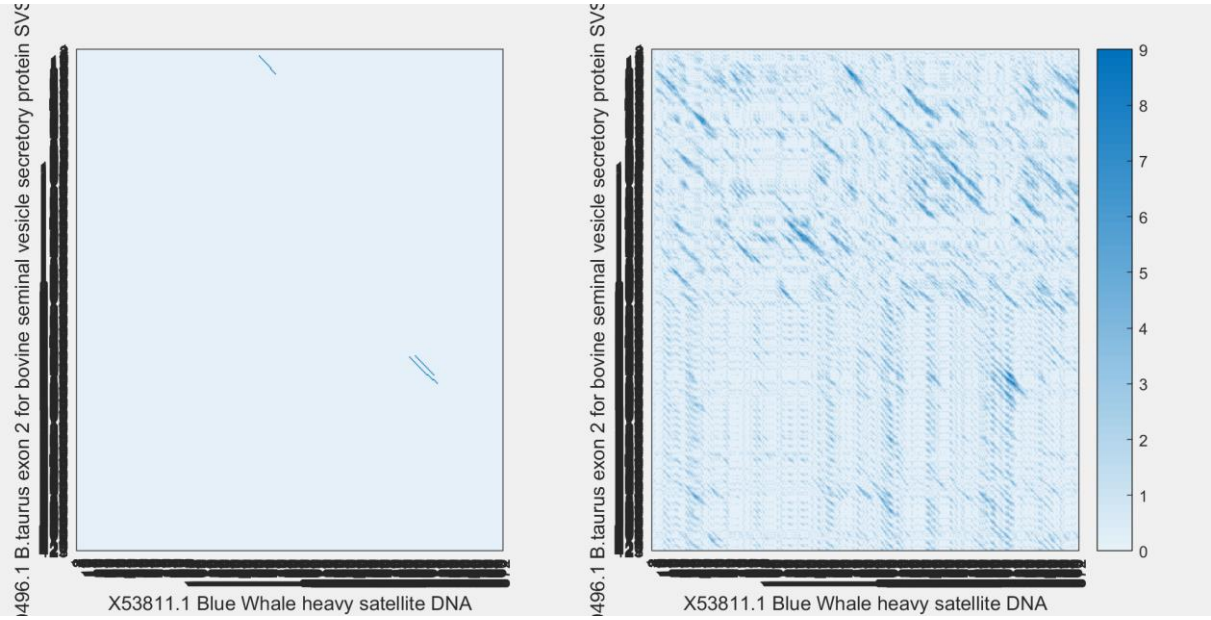
Powiązanych




```
Gap: -2
ID: Rhino gi|12903|emb|X56283.1|MIDBCYTB D.bicornis mitochondrion cytb gene for cytochrome b

GAAATTTTGGCTCTCTACTAGGAATCTGCCTAATCCTACAAATCCTAAC
CGGACTATTTCTTGCTATACATTATACACCAGACACAACAACTGCCTTCTCATCCGTTGCCCACATCTGT
CGAGAGGTAAACTACGGCTGAATTATCCGCTACCTACATGCAAACGGAGCATCCATATTTTTTATCTGCC
TATTCATCCACATAGGACGCGGCCTCTATTACGGATCCT
ID: AsiaticElephant gi|924712|gb|U23740.1|EMU23740 Elephas maximus cytochrome b (cytb) gene,

GAAATTTCGGCTCACTACTAGGAGCGTGCCTAATTACCCAAATCCTAACAGGATTATTCCTAGCCATACA
TTACACACCTGACACAATAACTGCATTTTCATCCATATCCCATATCTGCCGAGACGTCAACTACGGCTGA
ATTATTCGACAACTGCACTCAAACGGAGCATCTATCTTTTTCCTCTGCCTATACACACACATTGGACGAA
ACATCTACTATGGATCCT

***

GAAATTTTGGCTCTCTACTAGGAATCTGCCTAATCCTACAAATCCTAACCGGACTATTTCTTGCTATACA_TTATACACCAGACACAACAACT
GAAATTTCGGCTCACTACTAGGAGCGTGCCTAATTACCCAAATCCTAAC_AGGATTATTCCTAGCCATACATTACACACCTGACACAATAACT

Mode: similarity
Path length: 234
Match count: 181
Identity: 77.3504 %
Gap count: 6
Gap percentage: 2.5641 %
Score: 122
```

x

Niepowiązanych




```
Gap: -2
ID: X53811.1 Blue Whale heavy satellite DNA
TAGTTATTAAACCTATCCCACTCACTAGATACCCCTTAGCACATAAAGGAGTATTATTTGGGGGTCCAGCCATGGAGAAGAATTTAGACAC
ID: X60496.1 B.taurus exon 2 for bovine seminal vesicle secretory protein SVSP109
ATCCCCACTGGCTATCTATTTTACACACGGTGGTGTATATATGTCCACGCTACTCTCTCAATTTGTCCCACCAAACCCTGACCCTGCTTTG

***

TATTAAACCTATCCCA
TACTGAACCTA_CCCA

Mode: similarity
Path length: 16
Match count: 13
Identity: 81.25 %
Gap count: 1
Gap percentage: 6.25 %
Score: 9

***

GGGTACGGGTACGGGGA
GGGAACCGTGACGGGGA

Mode: similarity
Path length: 17
Match count: 13
```

```
Identity: 76.4706 %
Gap count: 0
Gap percentage: 0 %
Score: 9

***

GGTACGGGTACGGGGA_GGGG_TTC
GGCAGGGGAACCGTGACGGGGATTC

Mode: similarity
Path length: 25
Match count: 18
Identity: 72 %
Gap count: 2
Gap percentage: 8 %
Score: 9
```

Wnioski:

- Zauważamy, że sekwencje niepowiązane wykazują znacznie krótsze odcinki dopasowania, w porównaniu do sekwencji powiązanych, na podstawie powyższych przykładów – 25/234.
- Uzyskiwany przez sekwencje niepowiązane 'score' jest znacznie niższy od tych, które otrzymywane są przy porównywaniu sekwencji powiązanych, w przypadku powyższych przykładów – 9/122.
- Pomimo tego, że wartość 'identity' sekwencji niepowiązanych w niektórych przypadkach jest większa od sekwencji powiązanych, należy wziąć pod uwagę fakt, że ścieżki te są znacznie krótsze.
- Stosunek przerw do całej długości ścieżki jest mniejszy sekwencji powiązanych (na podstawie przykładu ~2%) niż sekwencji niepowiązanych (na podstawie przykładu - 8%)