SemiGlobalAlignment

Frank Li

April 3, 2020

1 Usage of the script

The code is designed for dealing with overlap alignment(aka semi-global alignment). For example, If we want to check if Sequence1(ATAGGTGATATA) will partially align with Sequence2(ATATACTGG), we just need to run following code in your terminal:

```
python3 SemiGlobalAlignment.py --q CTAGGTGATATA --s ATATACTGG
```

Listing 1: Unix Command

Where –q means your query sequence, –s means your subject sequence, Please make sure that the position of your query sequence and subject sequence are correct. This script can only handle the situation that suffix of your query sequence being aligned with the prefix of your subject sequence. If we are in the opposite situation, for example, S1(AAAAAATTTTT) and S2(GGGGGGGAAAAAA), you just need to simply put your S2 as query sequence and S1 as subject sequence.

Please install getopt package and numpy package.

You will get the following results as Figure 1:

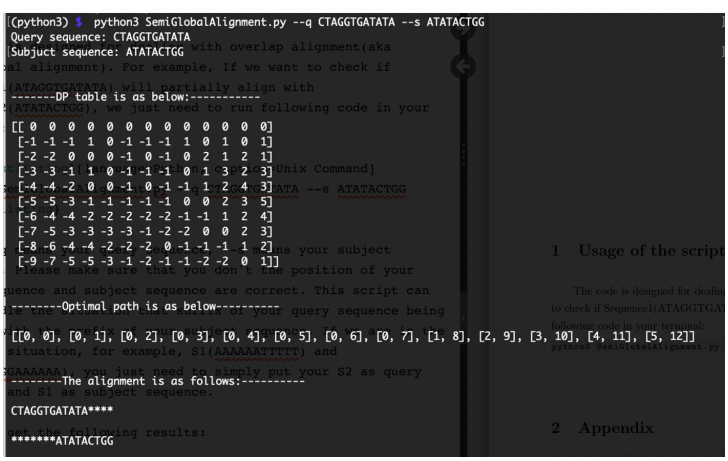

Figure 1: DP table, optimal path and final alignment

2 Appendix

```python
import os
import sys
import getopt
import numpy as np

def DP(query,subject,match = 1, mismatch = -1, indel = -1):
    n = len(subject) + 1
    m = len(query) + 1
    dp_table = np.zeros((n,m),dtype=np.int16) #int16 means each number at most occupy two bytes, ranging from -32768 ~ 32767
    #print(dp_table)
    # initilize
    dp_table[0,:] = 0
    for i in range(n):
        dp_table[i,0] = 0 + mismatch*i
    #print(dp_table)
    # forward calculation
    for i in range(1,n): # row
        for j in range(1,m): # column
            if query[j-1] == subject[i-1]:
                diag_move = dp_table[i-1,j-1] + match
            else:
                diag_move = dp_table[i-1,j-1] + mismatch
            hor_move = dp_table[i,j-1] + indel
            ver_move = dp_table[i-1,j] + indel
            dp_table[i,j] = max(diag_move,hor_move,ver_move)
    print('\n')
    print('-------DP table is as below:-----------\n')
    print(dp_table)
    # backward tracking
    row, column = 0,0
    trace = []
    max_last_column_index = np.argmax(dp_table[:,m-1])
    #print(max_last_column_index)
    row = max_last_column_index
    column = m-1
    trace.append([row,column])
    #print(row,column)
    current_entry = dp_table[row,column]
    #print(current_entry)
    while not (row == 0 and column ==0):
        if row > 0 and column > 0:
            if current_entry == dp_table[row-1,column-1] + 1 or current_entry == dp_table[row-1,column-1] - 1: # diag
                row,column = row-1,column-1
                trace.append([row,column])
                current_entry = dp_table[row,column]
            elif current_entry == dp_table[row-1,column] -1:  # vertical
                row,column = row-1, column
                trace.append([row,column])
                current_entry = dp_table[row,column]
            elif current_entry == dp_table[row,column-1] - 1: # horizontal
                row,column = row, column-1
```

```
                trace.append([row,column])
                current_entry = dp_table[row,column]
        elif row == 0 and column > 0: # must horizontal
            row,column = row, column-1
            trace.append([row,column])
            current_entry = dp_table[row,column]
        elif row > 0 and column == 0: # must vertical
            row,column = row-1, column
            trace.append([row,column])
            current_entry = dp_table[row,column]
    trace = trace[::-1]  # reverse the trace
    print('\n')
    print('--------Optimal path is as below----------\n')
    print(trace)
    # format the alignment
    for m in range(len(trace)):
        if not trace[m][0] == 0:
            break
    match_pos = m-1    # matching starts index 2 (third letter) as below
    leading_star_num = match_pos  # fill up 2 * at the beginning of subject sequence
    trailing_star_num = leading_star_num + len(subject) - len(query) # fill up 2 * at the end of query sequence
    print('\n')
    print('--------The alignment is as follows:----------\n')
    print('{0:*<{1}s}\n'.format(query,len(query)+trailing_star_num))
    print('{0:*>{1}s}\n'.format(subject,len(subject)+leading_star_num))

if __name__ == "__main__":
    # run as python3 SemiGlobalAlignment.py -q ATT -s TTT
    try:
        options, remainder = getopt.getopt(sys.argv[1:],'hq:s:',['help','q=','s=']) # getopt usage refer to pymotw.com/3/getopt/
    except getopt.GetoptError as err:
        print('ERROR:', err)
        usage()
        sys.exit(1)
    for opt, arg in options:
        if opt in ('--q','-q'):
            query = arg
            print('Query sequence:', arg)
        elif opt in ('--s','-s'):
            subject = arg
            print('Subjuct sequence:',arg)
        elif opt in ('--help','-h'):
            usage() # it doesn't work in getopt.getopt
            sys.exit()  # default is zero, means "successful termination", 2 means command line errors abnormal termination, 1 means other abnormal termination
    DP(query,subject)
```

Listing 2: Python Code