**[Assignment 12](https://mymasonportal.gmu.edu/webapps/assignment/uploadAssignment?content_id=_7816461_1&course_id=_331470_1&group_id=&mode=view)**

**Description:**

Python Data and Computations in the book, Computational Methods for Bioinformatics: Python 3.4, Third Edition by Jason Kinser.

Part 1

Worksheet 1, 2, and 3 exercises for learning module 31 DNA Alignment

Part 2

Problems 1 through 5 of Chapter 25 Sequence Alignment.

Part 3

In the Data section of the course website you will find a .zip file titled DNA\_Genome.zip with 15 genomes. Choose 1 of the genomes and use the dynamic programming procedure and code modules for sequence alignment that was presented in Chapter 25.6 to perform sequence alignment with 4 of the other sequences. Report the results of the sequence alignments you chose.

This is an individual assignment and should be your own work.

**Format:** Please submit the assignment in pdf or MSWord compatible format. Please show code used and output of the code in your submission.

**Point value:**

Part 1) 12 points (4 points per problem)

Part 2) 20 points (4 points per problem)

Part 3) 8 points  (2 points per alignment)

**Due Date:** April 30, 2018

## Part 1

See worksheets

## Part 2

Problems

1. Create a random sequence and copy it. In the copy remove a couple of letters at different places. Use NW to align these two sequences.

import dynprog as dyn

import blosum as b

seq1='ASDFASDFASDFASDF'

seq2='ASFASDFASDASDF'

subvals=dyn.FastSubValues(b.BLOSUM50,b.PBET,seq1,seq2)

scormat, arrow = dyn.FastNW(subvals,seq1,seq2)

print(scormat, arrow)

[[ 0 -8 -16 -24 -32 -40 -48 -56 -64 -72 -80 -88 -96 -104

-112]

[ -8 5 -3 -11 -19 -27 -35 -43 -51 -59 -67 -75 -83 -91

-99]

[ -16 -3 10 2 -6 -14 -22 -30 -38 -46 -54 -62 -70 -78

-86]

[ -24 -11 2 5 0 -6 -6 -14 -22 -30 -38 -46 -54 -62

-70]

[ -32 -19 -6 10 2 -3 -11 2 -6 -14 -22 -30 -38 -46

-54]

[ -40 -27 -14 2 15 7 -1 -6 7 -1 -9 -17 -25 -33

-41]

[ -48 -35 -22 -6 7 20 12 4 -1 12 4 -4 -12 -20

-28]

[ -56 -43 -30 -14 -1 12 28 20 12 4 20 12 4 -4

-12]

[ -64 -51 -38 -22 -9 4 20 36 28 20 12 17 9 1

4]

[ -72 -59 -46 -30 -17 -4 12 28 41 33 25 17 18 10

2]

[ -80 -67 -54 -38 -25 -12 4 20 33 46 38 30 22 18

10]

[ -88 -75 -62 -46 -33 -20 -4 12 25 38 54 46 38 30

22]

[ -96 -83 -70 -54 -41 -28 -12 4 17 30 46 51 43 35

38]

[-104 -91 -78 -62 -49 -36 -20 -4 9 22 38 51 52 44

36]

[-112 -99 -86 -70 -57 -44 -28 -12 1 14 30 43 56 52

44]

[-120 -107 -94 -78 -65 -52 -36 -20 -7 6 22 35 48 64

56]

[-128 -115 -102 -86 -73 -60 -44 -28 -15 -2 14 27 40 56

72]] [[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

[0 2 1 1 1 1 1 1 1 1 1 1 1 1 1]

[0 0 2 1 1 1 1 1 1 2 1 1 1 1 1]

[0 0 0 2 2 2 2 1 1 1 1 1 1 2 1]

[0 0 0 2 2 2 2 2 1 1 1 1 1 1 1]

[0 0 0 0 2 1 1 0 2 1 1 2 1 1 1]

[0 0 0 0 0 2 1 1 0 2 1 1 2 1 1]

[0 0 0 0 0 0 2 1 1 1 2 1 1 2 1]

[0 0 0 2 0 0 0 2 1 1 1 2 2 1 2]

[0 0 0 0 2 0 0 0 2 1 1 1 2 1 1]

[0 0 0 0 0 0 0 0 0 2 1 1 2 2 1]

[0 0 0 0 0 0 2 0 0 0 2 1 1 1 1]

[0 0 0 2 0 0 0 0 0 0 0 2 1 1 2]

[0 2 0 0 0 0 0 0 2 0 0 2 2 1 1]

[0 0 0 0 0 0 0 0 0 2 0 0 2 2 1]

[0 0 0 0 0 0 0 0 0 0 0 0 0 2 1]

[0 0 0 2 0 0 0 0 0 0 0 0 0 0 2]]

2. Repeat the above problem but change the gap penalty. Does the alignment change if the gap penalty is -16? Does it change if it is -2?

import dynprog as dyn

import blosum as b

seq1='ASDFASDFASDFASDF'

seq2='ASFASDFASDASDF'

subvals=dyn.FastSubValues(b.BLOSUM50,b.PBET,seq1,seq2)

scormat, arrow = dyn.FastNW(subvals,seq1,seq2)

print(scormat, arrow)

scormat, arrow = dyn.FastNW(subvals,seq1,seq2, -16)

print(scormat, arrow)

scormat, arrow = dyn.FastNW(subvals,seq1,seq2, -2)

print(scormat, arrow)

*[[ 0 -8 -16 -24 -32 -40 -48 -56 -64 -72 -80 -88 -96 -104*

*-112]*

*[ -8 5 -3 -11 -19 -27 -35 -43 -51 -59 -67 -75 -83 -91*

*-99]*

*[ -16 -3 10 2 -6 -14 -22 -30 -38 -46 -54 -62 -70 -78*

*-86]*

*[ -24 -11 2 5 0 -6 -6 -14 -22 -30 -38 -46 -54 -62*

*-70]*

*[ -32 -19 -6 10 2 -3 -11 2 -6 -14 -22 -30 -38 -46*

*-54]*

*[ -40 -27 -14 2 15 7 -1 -6 7 -1 -9 -17 -25 -33*

*-41]*

*[ -48 -35 -22 -6 7 20 12 4 -1 12 4 -4 -12 -20*

*-28]*

*[ -56 -43 -30 -14 -1 12 28 20 12 4 20 12 4 -4*

*-12]*

*[ -64 -51 -38 -22 -9 4 20 36 28 20 12 17 9 1*

*4]*

*[ -72 -59 -46 -30 -17 -4 12 28 41 33 25 17 18 10*

*2]*

*[ -80 -67 -54 -38 -25 -12 4 20 33 46 38 30 22 18*

*10]*

*[ -88 -75 -62 -46 -33 -20 -4 12 25 38 54 46 38 30*

*22]*

*[ -96 -83 -70 -54 -41 -28 -12 4 17 30 46 51 43 35*

*38]*

*[-104 -91 -78 -62 -49 -36 -20 -4 9 22 38 51 52 44*

*36]*

*[-112 -99 -86 -70 -57 -44 -28 -12 1 14 30 43 56 52*

*44]*

*[-120 -107 -94 -78 -65 -52 -36 -20 -7 6 22 35 48 64*

*56]*

*[-128 -115 -102 -86 -73 -60 -44 -28 -15 -2 14 27 40 56*

*72]] [[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]*

*[0 2 1 1 2 1 1 1 1 1 1 2 1 1 1]*

*[0 0 2 1 1 2 1 1 1 1 1 1 1 1 1]*

*[0 0 0 2 2 2 2 1 1 1 2 1 1 1 1]*

*[0 0 0 2 1 2 2 2 1 1 1 1 1 1 2]*

*[0 2 0 0 2 1 1 0 2 1 1 1 1 1 1]*

*[0 0 0 0 0 2 1 1 0 2 1 1 2 1 1]*

*[0 0 0 0 0 0 2 1 1 0 2 1 1 2 1]*

*[0 0 0 2 0 0 0 2 1 1 1 2 1 1 2]*

*[0 0 0 0 0 0 0 0 2 1 1 1 2 1 1]*

*[0 0 2 0 0 0 0 0 0 2 1 1 2 2 1]*

*[0 0 0 0 0 0 0 0 0 0 2 1 1 2 1]*

*[0 0 0 0 0 0 0 0 0 0 0 2 1 1 2]*

*[0 2 0 0 2 0 0 0 2 0 0 2 2 1 1]*

*[0 0 0 0 0 2 0 0 0 0 0 0 2 2 1]*

*[0 0 0 0 0 0 2 0 0 0 0 0 0 2 1]*

*[0 0 0 0 0 0 0 2 0 0 0 0 0 0 2]]*

*[[ 0 -16 -32 -48 -64 -80 -96 -112 -128 -144 -160 -176 -192 -208*

*-224]*

*[ -16 5 -11 -27 -43 -59 -75 -91 -107 -123 -139 -155 -171 -187*

*-203]*

*[ -32 -11 10 -6 -22 -38 -54 -70 -86 -102 -118 -134 -150 -166*

*-182]*

*[ -48 -27 -6 5 -8 -22 -30 -46 -62 -78 -94 -110 -126 -142*

*-158]*

*[ -64 -43 -22 2 2 -11 -27 -22 -38 -54 -70 -86 -102 -118*

*-134]*

*[ -80 -59 -38 -14 7 3 -13 -29 -17 -33 -49 -65 -81 -97*

*-113]*

*[ -96 -75 -54 -30 -9 12 3 -13 -28 -12 -28 -44 -60 -76*

*-92]*

*[-112 -91 -70 -46 -25 -4 20 4 -12 -28 -4 -20 -36 -52*

*-68]*

*[-128 -107 -86 -62 -41 -20 4 28 12 -4 -20 -7 -23 -39*

*-44]*

*[-144 -123 -102 -78 -57 -36 -12 12 33 17 1 -15 -6 -22*

*-38]*

*[-160 -139 -118 -94 -73 -52 -28 -4 17 38 22 6 -10 -6*

*-22]*

*[-176 -155 -134 -110 -89 -68 -44 -20 1 22 46 30 14 -2*

*-11]*

*[-192 -171 -150 -126 -105 -84 -60 -36 -15 6 30 43 27 11*

*6]*

*[-208 -187 -166 -142 -121 -100 -76 -52 -31 -10 14 35 44 28*

*12]*

*[-224 -203 -182 -158 -137 -116 -92 -68 -47 -26 -2 19 40 44*

*28]*

*[-240 -219 -198 -174 -153 -132 -108 -84 -63 -42 -18 3 24 48*

*39]*

*[-256 -235 -214 -190 -169 -148 -124 -100 -79 -58 -34 -13 8 32*

*56]] [[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]*

*[0 2 1 1 2 1 1 1 2 1 1 1 1 1 1]*

*[0 0 2 1 1 1 1 1 1 2 1 1 1 1 1]*

*[0 0 0 2 2 2 2 1 1 1 2 1 1 2 1]*

*[0 0 0 2 2 2 2 2 1 1 1 1 1 1 1]*

*[0 0 0 0 2 2 1 1 2 1 1 2 1 1 1]*

*[0 0 0 0 0 2 2 1 2 2 1 1 2 1 1]*

*[0 0 0 0 0 0 2 1 1 2 2 1 1 1 1]*

*[0 0 0 0 0 0 0 2 1 1 1 2 2 1 2]*

*[0 0 0 0 0 0 0 0 2 1 1 1 2 1 1]*

*[0 0 2 0 0 2 0 0 0 2 1 1 2 2 1]*

*[0 0 0 0 0 0 2 0 0 0 2 1 1 1 2]*

*[0 0 0 0 0 0 0 0 0 0 0 2 2 1 2]*

*[0 2 0 0 2 0 0 0 2 0 0 2 2 1 1]*

*[0 0 2 0 0 0 0 0 0 0 0 0 2 2 1]*

*[0 0 0 0 0 0 2 0 0 0 0 0 0 2 2]*

*[0 0 0 0 0 0 0 0 0 0 0 0 0 0 2]]*

*[[ 0 -2 -4 -6 -8 -10 -12 -14 -16 -18 -20 -22 -24 -26 -28]*

*[ -2 5 3 1 -1 -3 -5 -7 -9 -11 -13 -15 -17 -19 -21]*

*[ -4 3 10 8 6 4 2 0 -2 -4 -6 -8 -10 -12 -14]*

*[ -6 1 8 6 6 6 12 10 8 6 4 2 0 -2 -4]*

*[ -8 -1 6 16 14 12 10 20 18 16 14 12 10 8 6]*

*[-10 -3 4 14 21 19 17 18 25 23 21 19 17 15 13]*

*[-12 -5 2 12 19 26 24 22 23 30 28 26 24 22 20]*

*[-14 -7 0 10 17 24 34 32 30 28 38 36 34 32 30]*

*[-16 -9 -2 8 15 22 32 42 40 38 36 35 33 31 40]*

*[-18 -11 -4 6 13 20 30 40 47 45 43 41 39 37 38]*

*[-20 -13 -6 4 11 18 28 38 45 52 50 48 46 44 42]*

*[-22 -15 -8 2 9 16 26 36 43 50 60 58 56 54 52]*

*[-24 -17 -10 0 7 14 24 34 41 48 58 57 55 53 62]*

*[-26 -19 -12 -2 5 12 22 32 39 46 56 63 61 59 60]*

*[-28 -21 -14 -4 3 10 20 30 37 44 54 61 68 66 64]*

*[-30 -23 -16 -6 1 8 18 28 35 42 52 59 66 76 74]*

*[-32 -25 -18 -8 -1 6 16 26 33 40 50 57 64 74 84]] [[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]*

*[0 2 1 1 2 1 1 1 2 1 1 1 1 1 1]*

*[0 0 2 1 1 2 1 1 1 1 1 1 2 1 1]*

*[0 0 0 1 2 2 2 1 1 1 1 1 1 2 1]*

*[0 0 0 2 1 1 0 2 1 1 1 1 1 1 2]*

*[0 2 0 0 2 1 1 0 2 1 1 1 1 1 1]*

*[0 0 2 0 0 2 1 1 0 2 1 1 2 1 1]*

*[0 0 0 0 0 0 2 1 1 1 2 1 1 1 1]*

*[0 0 0 2 0 0 0 2 1 1 0 2 1 1 2]*

*[0 2 0 0 0 0 0 0 2 1 1 2 1 1 0]*

*[0 0 2 0 0 2 0 0 0 2 1 1 1 1 1]*

*[0 0 0 0 0 0 2 0 0 0 2 1 1 2 1]*

*[0 0 0 0 0 0 0 0 0 0 0 2 2 1 2]*

*[0 2 0 0 2 0 0 0 2 0 0 2 1 1 0]*

*[0 0 0 0 0 2 0 0 0 0 0 0 2 1 1]*

*[0 0 0 0 0 0 0 0 0 0 2 0 0 2 1]*

*[0 0 0 2 0 0 0 2 0 0 0 0 0 0 2]]*

3. Create a scoring matrix which is, M[i, j] = (5 i = j −1 i 6= j . Align two amino acid sequences (of at least 100 characters) using the BLOSUM50 matrix and the above M matrix. Are the alignments significantly different?

from blosum import BLOSUM50, PBET

import dynprog as dpg

import random as rd

N=101

rd.seed(23)

seq1 = ''.join(rd.choices(PBET, k=N))

seq2 = ''.join(rd.choices(PBET, k=N))

subvals = dpg.FastSubValues(BLOSUM50, PBET, seq1, seq2)

schmat, arrow = dpg.FastNW(subvals, seq1, seq2)

t1, t2 = dpg.SWBacktrace(schmat, arrow, seq1, seq2)

print(t1, t2)

*YY-WRKH HFTMCQH*

4. Modify the BlosumScore algorithm to align DNA strings such that the 3-rd element in each codon is weighted half as much as the other two.

def BlosumScore2( mat, abet, s1, s2, gap=-8 ):

sc = 0

n = min( [len(s1), len(s2)] )

for i in range( n ):

if s1[i] == '-' or s2[i] == '-' and s1[i] != s2[i]:

if i // 3 == 0:

sc += gap \* 0.5

sc += gap

elif s1[i] == '.' or s2[i] == '.':

pass

else:

n1 = abet.index( s1[i] )

n2 = abet.index( s2[i] )

if i // 3 == 0:

sc += mat[n1,n2] \* 0.5

sc += mat[n1,n2]

return sc

def prob4():

from blosum import BLOSUM50, PBET

from simplealign import BlosumScore

import dynprog as dpg

import random as rd

N=101

rd.seed(23)

seq1 = ''.join(rd.choices(PBET, k=N))

seq2 = ''.join(rd.choices(PBET, k=N))

score = BlosumScore(BLOSUM50, PBET, seq1, seq2)

score2 = BlosumScore2(BLOSUM50, PBET, seq1, seq2)

print(score, score2)

prob4()

-119 -*117*.5

5. Create a string with the form XAXBX where X is a set of random letters and A and B are specific strings designed by the user. Each X can be a different length. Create a second string with the form YAYBY where Y is a different set of random letters and each Y can have a different length. Align the sequences using Smith-Waterman. The scoring matrix will have two major maximum for the alignments of the A and B regions. Modify the program to extract both alignments.

def stringXAXBX(a, b):

import string

import random as rd

from blosum import PBET

rd.seed(123)

N = max(len(a), len(b))

Nx = rd.randint(1, N)

X = ''.join(rd.choices(PBET, k=Nx))

Ny = rd.randint(1, N)

Y = ''.join(rd.choices(PBET, k=Ny))

Nz = rd.randint(1, N)

Z = ''.join(rd.choices(PBET, k=Nz))

return '{}{}{}{}{}'.format(X,a,Y,b,Z)

import dynprog as dpg

from blosum import BLOSUM50, PBET

import random as rd

a = ''.join(rd.choices(PBET, k=12))

b = ''.join(rd.choices(PBET, k=12))

Xstr = stringXAXBX(a, b)

Ystr = stringXAXBX(a, b)

subvals = dpg.FastSubValues(BLOSUM50, PBET, Xstr, Ystr)

scmat, arrow = dpg.FastSW(subvals, Xstr, Ystr)

t1, t2, = dpg.SWBacktrace(scmat, arrow, Xstr, Ystr)

print(t1, t2)

*QWEAHGTSHYQKGQTWGKERPTNSEIVYGLLAEEC QWEAHGTSHYQKGQTWGKERPTNSEIVYGLLAEEC*

## Part 3

import dynprog as dpg

import blosum as b

# get 1 genome

gen1 = open('DNA\_Genome/960523 01.seq').read()

# with 4 of the other sequences.

gen2=open('DNA\_Genome/960523 02.seq').read()

gen3=open('DNA\_Genome/960523 03.seq').read()

gen4=open('DNA\_Genome/960523 04.seq').read()

gen5=open('DNA\_Genome/960523 05.seq').read()

other\_genes=[gen2,gen3,gen4,gen5]

# perform sequence alignment w/ FastNW

for gene in other\_genes:

subvals = dpg.FastSubValues(b.BLOSUM50, b.PBET, gen1, gene)

scormat, arrow = dpg.FastNW(subvals, gen1, gene)

t1, t2 = dpg.SWBacktrace(scormat, arrow, gen1, gene)

print(t1, t2)

*TNNGA-ATTGCATACGACTCACTATAGGGCGAATTCGAGCTCGGTACCCGGGGATCCTCTAGAGT--CG-ACCTGCAGGCATGCA-AG-CTTGAGTATTCTATAGTGTCACCTAAATAGCTTGGCGTAATCATG-GTCAT-AGCTGTTTCCTGTGTGAAATTGTTATCCGCTCACAATTCC-ACACAACA-TACGAGCCGGAAGCATAAAGTGTAAAGCCTGGGGTGCCTAATGAGTGAGCTAACTCACATTAATTGCGTTGCGCTCACTGC-CCGCTTTC-CAGTCGGGAAACCTGTC-GTGCCAGCTGCATTAATGAATCG-GCCAACGCGCGGGGAG-AGGCGGTTTGCGTATTGGGCG-C-TCTTCCG-CTTCCTCGCTCACTGACT-CG-CTGNGCTCGGTCGTTCGGCTGC-G--GCGAGCGGTATCAGCTCACTCAAAGG-C-GGG-T-AATACGGG--TT-ATCCACAGGA--ATCAG-GGGATAACGC-AG--GAAA-GACATGTGAGCAAA-AGGGCAGCAAAAGGG--CAGGAACCCTAAAAAGGCCGCGT-TGGTGGGNTTTTCCATAGGGTCCCCCC-CCTGANGAGATAAAAAANCGA-GGTCAC TNCGAGACNCCNTTCGGGTAGCT-CGCNNCGATTAC-ATCTCGGNG-ATGNATNACNGC-AGCGTNACNCTCC-GCCAG-AAGNANAGTCTAGNTACTTCTGGNGGANC-CC--ACTCCCATGGTGTGA-CGGGCGGTGTGTNCAANGNCC-G-GT-AACGNATNCACCGCN-ACA-TNCTGATNCGGGANTACTAG-CG-ATTCCGANTTCANGCAGTC-GAGNNGCAGACTGCG-ATCCGGACT-ACGATCGGTTTTCTGGGATTA--GCTCCAC-CTCGCGGNTTGGNAACCN-TCTGTNCCGAC--CATTGTNGCA-CGTGTGTA-GCCCAGGCCGTAAGGGNCATGATGACTTGACGTCATC-CCCACCTTCCTC-CGGTTTGTCACCGGCAGT-CTCCTTAGAGTGGCCACCATTACGTGCTG-GT-AACTAAGGGAAAGGTTTGGGCTCGTTACGGGATTTTAACCCCAACATCTTCAACGAAACGGAGCTTGANGAAAAGGCATGCAAGGAACCTGGTCTTCAAATTNTTCCCCGNAGGGGAACCAAATCTTTTCTNTTGGAAAGTTCAATGGGTTTTNAAGGCCTGGNAAGGTTCTTCCGGGTGGGTTTC*

*NNGAATTGCATACGACTCACTATAGGGCGAATTCGAGCTCGGTACCCGGGGATCCTCTAGAGTCGACC-T-GCAGGCATGCAAGCTTGAGTATTCTATAGTGTCACCTAAATAGCTTGGCGTAATCATGGTCATAGCTGTTTCCTGTGTGAAATTGTTA-TC-CGCTCACAATTCCACACAACATACGAGCCGGAAGCATAAAGTGTAAAGCCTGGGGTGC-CTAATGA-GTGAGC-TAACTC-ACAT-TAATTGCGTTGCGCTCACTGCCCGCTTTCCAGTCGGGAAAC-CTGTCGTGCCAGCT-GCATTAATGAATCGGCCA-AC-GCGCGGGGAGAGGCGGTTTGCGTATTGGGCGCTCTTC-CGCTTCCTCGCTCACTGACTCGCTGNGCTCGGTCGTTCGGCTGCGGCGAGCGGTATCAGCTCACTCAAAGGCGGGTAATACGG--GTTATCCACAGG-AATCAGGGGATAACG-CAGGAAAGACA-TGTGAGCAAAAGGGCAGCAAAAGG---GCAGGAACCC-TAAAAAGGCCGCGTTG-GTGGG-NT-TT-TCCATAGGGTCCC-C----C--CCCTGANGAGATAAAAAANCGAGGTCAC NNN--NTNCTTT--TCTCGGGATTA-GC---TCC-ACCTCGCGGCTTGGCAACCCTCT-GTACCGACCATTGTA-GCACGTGTG-TAGCCCAGGCCGTAAGGGC-CATG-ATGACTTGACGTCATCC---CCAC--CTTCCTCCGGTTTGTCACCGGCAGTCTC-CTTAGAGTGCC-CACCA-TTACGTGCTGGTAAC-TAA-G-G---A-CAAGGGTTGCGCTCGTTACGGGA-CTTAACCCAACATCTCACGAC-ACGAGCTGAC-GACAGCCATGCAG-CACCTGTCTCAAT-GTTCCCGAAGGCACCAATCTATC-TCTAGAAAGTTCATTGGATGTCAA-G-GCCTGGTAAG-GTTCTTCGCGTTGCTTCGAA-TTAAAC-CACA-TGCTCCACCGCTTGG-TGCGG-GCCCC-CGTCAATTCATTTGAGTTTTAACCTTGNGGNCGTACTCCCCAGGCGGTCAACTTAATGCGTTAGCTGCGCCACTAAGAGCTCAAGG-CTCCCAACGGNTAGTTGGAAATCGTTTACGGGGTGGATTACCAGGGTNTCTTATCCTGTTGGTCCCACGNTTCGACCCAG-NGTNA-GNTCAGNC-AGGTGGC*

*-TNNGA--ATTG-CATACGACTCACTATAGGGCGAA-TTCGAGCTCGGTACCCGGGGATCCTCTAG-A-G-TCGAC--CTG-C-AGGCATG-CAA-G--CTT-GAGT-A-TTC-TATAGTGTCAC---CTAAATAGCTTG-GCGTAATCATGGTCATAGCTGTTTCCTGTGTGAA-ATTGTTATC-CGCTCACAATTCCACACAACATACGAGCCGGAAGCATAAAGTGTAA-AG-CCTGGGGTGCCTAATGAGTGAGCTAACTCACATTAATTGCGTTGCGCTCACTGC-CCGCT-TTCCAGTCGGGAAACCTGTCGTGCCAGCTGCATTAATGAATCGGCCAACGCGCGGGGAGAGGCGGTTTGCGT-ATTGG-GCGCTCTTCCGCTTCCT-CGCTCACTGACTCGCTGNGCTC-GGTCGTTCGGCTGCG-GCGAGCGGTATC-AGCTCACTCAAAGGCGGGTAATACGGGTTATCCACAG-GAATCAGGGGATAACGCAGGAAAGACATGTGAGCAAAAGGGCAGCAAAAGG-GCAGGAAC-CCTAAAAAGGCCGCGTTG-GTGGGNTTTTCCATAGGGTCCCCCCCCTGANGAGATAAAAAANCGAGGTCAC ATCNGACGACAGNCATGCAGCAC-CTGTCT--CAATGTTCCCG-AAGGNACCAATCTATC-TCTAGAAAGTTCATTGGATGTCAAGGCCTGGTAAGGTTCTTCGCGTTGCTTCGAATTAAACCACATGCTCCACCGCTTGTGCGGGCCCCCG-NCAATTC-ATTT-GAGTTTTAACCTTGNGGNCGTACTCCCCAGGNGG-NCAACTTA-ATG-CGTTAGCTGCGCCACTAAGAGNTCAAGGNT-CCCAACG-GCTAGTTGACA-TCGTTTACGGNGTGG-ACTACCAGGGTATCTAATCCTGTTTGNTCCCCACGCTT-TC-GCA-CCTCAGTG--TCAG-TATCAGTCCAGGTG-GTCGNCTT-CGCCACTGGTGTTC-CTNCCTATATCTACGCATT-TCAC-CGCT-TACACAGGAAATTCCACCACCCTCTACCATACTCTAGCTCG-TCAGT-TTNGG-A-TGCAGGT--CCCAGGTTGANCCCNGGGATTTCANAT-CCA-ACTT-TAAN-GAAC---CACCTACGGGNGCTTTTANGCC-AGTAATTCCGATTAACGTTGG--A-ACC-TCTGTTTACCCGGNTGTTGGAAAGAGTTAGCCGNGTTTC*

*NNGAATTGCATACGACTCACTATAGGGCGAATTCGAGCTCGGTACCCGGGGATCCTCTAGAGTCG-AC-CTGCAGGCATGCAAGCTTGA-GTATTCTATAGTGTCACCTAAATAGCTTGGCGTAATCATGGTC-A-TAG-C-T-GTTTCCTGTGTGAAATT---GT-TATCCGCTCACAATT---CCACACAACATACGAGCCGGAAGCATAAAGTGTAAAGCCTGG-GGT-GCCTA-ATGAGTGAGCTAACTCACATTAA-TT-GCGTTGCGCTCACTGC-CCG-CT-TTCCAG-TCGGGAAACCTGTCGTGCCAGCT-G-CATTAATGAAT-CGG--CC-A--AC--GCGCGGGGAGAG-GCGGTTTGCGTATTGGGCGCTC-TTC-CGC-TTCCTCGCTCACTGACTCGCTGNGCTCGGTCGTT-CGGCTGCGGCGA-GCGGTATC-AGCTCACTCAAAGGCGG-G-TAATAC-GG-GTTATCCACAGGAATCAGG--GG-AT-AACGCA-GGAAAGACATGTGAGCA-AAAGGGC-AGCAAAAGGGCAGGAACCCTAAAAAGGCCGCGTTGGTGGGNTTTTCCATAGGGTCCCCCCCC NNN--TN-CGTCC-ACGCTTGTGCGGGCCCCGTCAA-TTCATTTGAGTTTTAACCTTGCG-GCCGTACTCCCCAGGCGGTCAA-CTTAATGCGTTAGCT-GCGCCACTAAGAGNTCAAGGC-TCC-CAACGGCTAGTTGACATCGTTTACGGCGTGGACTACCAGGGTATCTAATCCTGTTTGNTCCCCACG-CTTTCGCACC-TCAGTGTCA-GTATCAGTCCAGGTGGTCGCCTTCGCCACTGGTGTTCCTNCCTATATCTACGCATTTCAC-CGCTACACAGGAAATTCCACCACCCTCTACCATAC-T-CTAGCTCGTCAGTTTTGAATGCAGTTCCCAGGTTGAGCCCGGGGATTTCACATCCAACTTAACGAAC-CACCTACGCGCGCTTTACGCCCAGTAATTC-CGATTAACGNTTGNANCCTCTGTATTACCGCGGGTGCTGGCACAGAGTTAGNCGGTGCTTATTCTGTCGGTAACGTCA-AAA-CAGGAAAGTATTAATTTACTGCCCTTCCTCCCAACTTAAAGTGCTTACAATCCGA-AGACTCTTCAAAAAGG--G-GATGGT-GGNTNAGGTTTTCGG-CCATGGCC*