“AÑO DEL BICENTENARIO DEL PERÚ: 200 AÑOS DE INDEPENDENCIA”.

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

CIENCIA DE LA COMPUTACIÓN

BIOINFORMÁTICA

Práctica 06

Estudiantes:
Miguel Alexander, Herrera Cooper
Milagros Celia, Cruz Mamani
Yara Jeanette, Quispe Quispe
Ingrid Sally, Espinel Quispe

Docente:
Mg. Vicente Enrique, Machaca Arceda

May 27, 2021

Contents

1 Actividad 1

Implemente el algoritmo de alineamiento de secuencias utilizando programación dinámica (Needleman–Wunsch). Evalue sus resultados con las secuencias:

- S_1 : AAG
- S_2 : AGC

Utilice gapOpen = gapEXT EN D = -5 y la siguiente matriz de sustitución:

	A	**C**	**G**	**T**
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Table 1: Matriz de Similitud

La salida debe incluir el score matrix y todos los alineamientos posibles.

Resolución

1.1 Código

A continuación, mostramos el algoritmo implementado en Python.

```
import sys
import numpy as np

def Matriz_Sustitucion(file):
  next(file)
  r_c = {}
  s = []
  i = 0
  for line in file:
    if len(line.strip())!=0:
      line=line.rstrip('\n')

      r_c[line.split('\t')[0:1][0]] = i
      i = i+1
      s.append(list(map(int,line.split('\t')[1:])))
  file.close()
  return r_c, s

def Obtener_Secuencias(F, i, j, secuencia_alineada_1 = "",
    secuencia_alineada_2 = ""):
  if F[i][j][1]==0:
    print (secuencia_alineada_1)
    print (secuencia_alineada_2)
    Obtener_Puntaje(secuencia_alineada_1, secuencia_alineada_2)
    print ()
```

```
25      return
26
27    if len(F[i][j][1])>1:
28      direcciones = F[i][j][1]
29      for n in range(len(direcciones)):
30        F[i][j][1] = direcciones[n]
31        Obtener_Secuencias(F, i, j, secuencia_alineada_1,
      secuencia_alineada_2)
32
33    else:
34      if F[i][j][1] == 'D':
35        secuencia_alineada_1 = secuencia_1[j-1] + secuencia_alineada_1
36        secuencia_alineada_2 = secuencia_2[i-1] + secuencia_alineada_2
37        i = i-1
38        j = j-1
39
40      elif F[i][j][1]=='U':
41        secuencia_alineada_1 = "-" + secuencia_alineada_2
42        secuencia_alineada_2 = secuencia_2[i-1] + secuencia_alineada_2
43        i = i-1
44      else:
45        secuencia_alineada_2 = "-" + secuencia_alineada_2
46        secuencia_alineada_1 = secuencia_1[j-1] + secuencia_alineada_1
47        j = j-1
48
49      Obtener_Secuencias(F, i, j, secuencia_alineada_1, secuencia_alineada_2)


50
51
52  def Obtener_Puntaje(secuencia_1, secuencia_2):
53    puntaje = 0
54    for i in range(len(secuencia_1)):
55      if secuencia_1[i] == secuencia_2[i]:
56        print (s[r_c[secuencia_2[i]]][r_c[secuencia_1[i]]], end='', sep='')
57        puntaje = puntaje + s[r_c[secuencia_2[i]]][r_c[secuencia_1[i]]]
58      else:
59        if secuencia_1[i]=='-' or secuencia_2[i]=='-':
60          print ('(',d,')', end='', sep='')
61          puntaje = puntaje + d
62        else:
63          print ('(',s[r_c[secuencia_2[i]]][r_c[secuencia_1[i]]] ,')', end=''
      , sep='')
64          puntaje = puntaje + s[r_c[secuencia_2[i]]][r_c[secuencia_1[i]]]
65
66      print ('+', end='', sep='')
67
68    print("\b",end="")
69    print ('=', puntaje)
70
71
72  def Alineamiento_Global(F, i, j):
73    direcciones = ''
74
75    diag = F[i-1][j-1][0] + s[r_c[secuencia_2[i-1]]][r_c[secuencia_1[j-1]]]
76    up = F[i-1][j][0] + d
77    left = F[i][j-1][0] + d
78
```

```
  F[i][j][0]  = max(diag, up, left)

  if F[i][j][0]==left:
    direcciones = direcciones + 'L'

  if F[i][j][0]==diag:
    direcciones = direcciones + 'D'

  if F[i][j][0]==up:
    direcciones = direcciones + 'U'

  F[i][j][1] = direcciones

  if i==fila-1 and j==columna-1:
    Guardar_Resultado(F)
    Obtener_Secuencias(F, i, j)
    return

  elif j<columna-1:
    Alineamiento_Global(F, i ,j+1)
  else:
    Alineamiento_Global(F, i+1 ,1)


def Guardar_Resultado(F):
  f = open(archivo_final, "w")

  f.write('              ')
  for i in range(columna-1):
    f.write(secuencia_1[i]+'        ')

  f.write('\n')

  secuencia_2_t = ' '+ secuencia_2

  for r in range(fila):
    f.write(secuencia_2_t[r])
    for c in range(columna):
      f.write(' ( '+ '{}'.format(F[r][c][0])+' '+'{}'.format(F[r][c][1])+')
    ')

    f.write('\n')
  f.close()

  f = open(archivo_final, "r")
  print(f.read())


if __name__ == "__main__":
  filename = sys.argv[1]
  d = int(sys.argv[2])
  archivo_final = "resultado.txt"

  file = open(filename, "r")

  r_c, s = Matriz_Sustitucion(file)

```

```
secuencia_1 = "AAG"
secuencia_2 = "AGC"

# columnaa y fila  de 0's al inicio de la matriz
columna = len(secuencia_1)+1
fila = len(secuencia_2)+1

F = np.zeros([fila, columna], dtype='i,O')

# Agregando 0's
for i in range(1,columna):
  F[0][i][0] = i*d
  F[0][i][1] = 'L'

for i in range(1,fila):
  F[i][0][0] = i*d
  F[i][0][1] = 'U'

Alineamiento_Global(F, 1, 1)
```

1.2 Pruebas

Ahora veremos la prueba realizada según lo solicitado en el ejercicio

1.2.1 Input

Evaluamos los resultados con las secuencias

```
secuencia_1 = "AAG"
secuencia_2 = "AGC"
```

1.2.2 Output

Seguidamente pasamos a ver el resultado generado :

```
cooper@cooper-Legion-Y545:~/Escritorio/5to/9no_SEMESTRE/BIOINFORMATICS/Semana_07
$ python3 ejercicio1.py substitution_matrix.txt -5
              A         A         G
  ( 0 0)  ( -5 L)  ( -10 L)  ( -15 L)
A ( -5 U)  ( 2 D)  ( -3 LD)  ( -8 L)
G ( -10 U)  ( -3 U)  ( -3 D)  ( -1 D)
C ( -15 U)  ( -8 U)  ( -8 U)  ( -6 U)

AAG-
A-GC
2+(-5)+2+(-5)= -6

AAG-
-AGC
(-5)+2+2+(-5)= -6
```

Figure 1: Matriz Final - Alineaciones - Puntajes

2 Actividad 2

Utilice el algoritmo implentado anteriormente, pero esta vez con secuencies reales. Se recomienda utilizar las secuencias de proteinas de la Práctica 3.

Utilice *gapOpen* = *gapEXTEND* = -5, *identicalMatch* = 2 y *mismatch* = -2.

Resolución

2.1 Código

A continuación, mostramos el algoritmo implementado en Python.

```
import sys
import numpy as np

def Alineamiento_Global(secuencia_1, secuencia_2,penalidad):
  columna = len(secuencia_1)+1
  fila = len(secuencia_2)+1
  F = np.zeros([fila, columna], dtype=int)
  # Agregamos 0's
  for i in range(1,columna):
    F[0][i] = i*penalidad

  for i in range(1,fila):
    F[i][0] = i*penalidad

  for i in range(1,fila):
    for j in range(1, columna):
      valor = identicalMatch
      if secuencia_2[i-1] != secuencia_1[j-1]:
        valor = mismatch
      F[i][j] = max(F[i-1][j-1] + valor, F[i-1][j] + penalidad, F[i][j-1] +
    penalidad)
  print("Matriz:\n")
  print (F)

  i = fila-1
  j = columna-1

  secuencia_alineada_1 = ""
  secuencia_alineada_2 = ""

  while (i > 0 or j > 0): # alineamento
    valor = identicalMatch
    if secuencia_2[i-1] != secuencia_1[j-1]:
      valor = mismatch
    if (i>0 and j>0 and F[i][j] == F[i-1][j-1] + valor):
      secuencia_alineada_1 = secuencia_1[j-1] + secuencia_alineada_1
      secuencia_alineada_2 = secuencia_2[i-1] + secuencia_alineada_2
      i = i-1
      j = j-1
    elif (i>0 and F[i][j]==F[i-1][j]+penalidad):
      secuencia_alineada_1 = "-" + secuencia_alineada_2
      secuencia_alineada_2 = secuencia_2[i-1] + secuencia_alineada_2
```

```
      i = i-1
    else:
      secuencia_alineada_2 = "-" + secuencia_alineada_2
      secuencia_alineada_1 = secuencia_1[j-1] + secuencia_alineada_1
      j = j-1
  print ()
  print("Secuencia Alineada 1 :\n")
  print (secuencia_alineada_1)
  print ()
  print("Secuencia Alineada 2:\n")
  print (secuencia_alineada_2)
  print("\n")
  print()

def Obtener_Secuencia(file):
  next(file)
  secuencia = ""
  for linea in file:
    if len(linea.strip())!=0:
      linea=linea.rstrip('\n')
      secuencia = secuencia + linea
  file.close()
  return secuencia

if __name__ == "__main__":
  file1 = sys.argv[1]
  file2 = sys.argv[2]
  penalidad = int(sys.argv[3])
  identicalMatch = 2
  mismatch = -2
  f1 = open(file1, "r")
  f2 = open(file2, "r")
  secuencia_1 = Obtener_Secuencia(f1)
  secuencia_2 = Obtener_Secuencia(f2)
  Alineamiento_Global(secuencia_1, secuencia_2,penalidad)
```

2.2 Pruebas

2.2.1 Input

De acuerdo a lo solicitado, realizaremos las pruebas con los archivos .fasta, trabajados en la Práctica 03.

1. P21333.fasta - Filamina-A (Humano)
2. Q8BTM8.fasta - Filamina-A (Ratón)

2.2.2 Output

Seguidamente pasamos a ver el resultado generado :

```
cooper@cooper-Legion-Y545:~/Escritorio/5to/9no_SEMESTRE/BIOINFORMATICS/Semana_07
$ python3 ejercicio2.py P21333.fasta.txt Q8BTM8.fasta.txt -5
Matriz:

[[     0     -5    -10 ... -13225 -13230 -13235]
 [    -5      2     -3 ... -13218 -13223 -13228]
 [   -10     -3      4 ... -13211 -13216 -13221]
 ...
 [-13225 -13218 -13211 ...   5006   5001   4996]
 [-13230 -13223 -13216 ...   5005   5008   5003]
 [-13235 -13228 -13221 ...   5000   5003   5010]]
```

Figure 2: Matriz Final - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

Alineamineto para P21333.fasta - Filamina-A (Humano)

```
Secuencia Alineada 1 :

MSSSHSRAGQSAAGAAPGGGVDTRDAEMPATEKDLAEDAPWKKIQQNTFTRWCNEHLKCVSKRIANLQTDLSDGLRLIALLEVLSQKKMHRKHNQRPTFRQMQ
LENVSVALEFLDRESIKLVSIDSKAIVDGNLKLILGLIWTLILHYSISMPMWDEEEDEEAKKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCA
PGLCPDWDSWDASKPVTNAREAMQQADDWLGIPQVITPEEIVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETR
SAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYVPEVTGTHKVTVLFAGQHIAKSPFEVYVDKSQGDASKVTAQGPGLEPSGNIANKTTYFEIFTAGAG
TGEVEVVIQDPMGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPSACRAVGRGLQPKGVRVKETADFKVYTKGAGSGEL
KVTVKGPKGEERVKQKDLGDGVYGFEYYPMVPGTYIVTITWGGQNIGRSPFEVKVGTECGNQKVRAWGPGLEGGVVGKSADFVVEAIGDDVGTLGFSVEGPSQ
AKIECDDKGDGSCDVRYWPQEAGEYAVHVLCNSEDIRLSPFMADIRDAPQDFHPDRVKARGPGLEKTGVAVNKPAEFTVDAKHGGKAPLRVQVQDNEGCPVEA
LVKDNGNGTYSCSYVPRKPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKVYGPGVAKTGLKAHEPTYFTVDCAEAGQGDVSIGIKCAPGVVGPAEADIDF
DIIRNDNDTFTVKYTPRGAGSYTIMVLFADQATPTSPIRVKVEPSHDASKVKAEGPGLSRTGVELGKPTHFTVNAKAAGKGKLDVQFSGLTKGDAVRDVDIID
HHDNTYTVKYTPVQQGPVGVNVTYGGDPIPKSPFSVAVSPSLDLSKIKVSGLGEKVDVGKDQEFTVKSKGAGGQGKVASKIVGPSGAAVPCKVEPGLGADNSV
VRFLPREEGPYEVEVTYDGVPVPGSPFPLEAVAPTKPSKVKAFGPGLQGGSAGSPARFTIDTKGAGTGGLGLTVEGPCEAQLECLDNGDGTCSVSYVPTEPGD
YNINILFADTHIPGSPFKAHVVPCFDASKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSEAGLPAEVYIQDHGDGTHTITYIPLCPGAYTVTIKYG
GQPVPNFPSKLQVEPAVDTSGVQCYGPGIEGQGVFREATTEFSVDARALTQTGGPHVKARVANPSGNLTETYVQDRGDGMYKVEYTPYEEGLHSVDVTYDGSP
VPSSPFQVPVTEGCDPSRVRVHGPGIQSGTTNKPNKFTVETRGAGTGGLGLAVEGPSEAKMSCMDNKDGSCSVEYIPYEAGTYSLNVTYGGHQVPGSPFKVPV
HDVTDASKVKCSGPGLSPGMVRANLPQSFQVDTSKAGVAPLQVKVQGPKGLVEPVDVVDNADGTQTVNYVPSREGPYSISVLYGDEEVPRSPFKVKVLPTHDA
SKVKASGPGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGKPKKTHIQDNHDGTYTVAYVPDVTGRYTILIKYGGDEIPFSPYRVRAVPTGDASKCTVT
VSIGGHGLGAGIGPTIQIGEETVITVDTKAAGKGKVTCTVCTPDGSEVDVDVVENEDGTFDIFYTAPQPGKYVICVRFGGEHVPNSPFQVTALAGDQPSVQPP
LRSQQLAPQYTYAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIPFTIKKGEITGEVRMPSGKVAQPTITDNKDGTVTVRYAPSEAGLHEMDIRYDNMHIPGS
PLQFYVDYVNCGHVTAYGPGLTHGVVNKPATFTVNTKDAGEGGLSLAIEGPSKAEISCTDNQDGTCSVSYLPVLPGDYSILVKYNEQHVPGSPFTARVTGDDS
MRMSHLKVGSAADIPINISETDLSLLTATVVPPSGREEPCLLKRLRNGHVGISFVPKETGEHLVHVKKNGQHVASSPIPVVISQSEIGDASRVRVSGQGLHEG
HTFEPAEFIIDTRDAGYGGLSLSIEGPSKVDINTEDLEDGTCRVTYCPTEPGNYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANVGSHCDLS
LKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYCIRFVPAEMGTHTVSVKYKGQHVPGSPFQFTVGPLGEGGAHKVRAGGPGLERAEAGVPAEFSIWTREA
GAGGLAIAVEGPSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNEEHIPDSPFVVPVASPSGDARRLTVSSLQESGLKVNQPASFAVSLNGAKGAIDAKVH
SPSGALEECYVTEIDQDKYAVRFIPRENGVYLIDVKFNGTHIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGNPAEFVVNTSNAGAGALSVTIDGPSKV
KMDCQECPEGYRVTYTPMAPGSYLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKATCAPQHGAPGPGPADASKVVAKGLGLSKAYVGQK
SSFTVDCSKAGNNMLLVGVHGPRTPCEEILVKHVGSRLYSVSYLLKDKGEYTLVVKWGDEHIPGSPYRVVVP
```

Figure 3: Alineamineto P21333 - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

Alineamineto para Q8BTM8.fasta - Filamina-A (Ratón)

```
Secuencia Alineada 2:

MSSSHSRCGQSAAVASPGGSIDSRDAEMPATEKDLAEDAPWKKIQQNTFTRWCNEHLKCVSKRIANLQTDLSDGLRLIALLEVLSQKKMHRKHNQRPTFRQMQ
LENVSVALEFLDRESIKLVSIDSKAIVDGNLKLILGLIWTLILHYSISMPMWDEEEDEEAKKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCA
PGLCPDWDSWDASKPVNNAREAMQQADDWLGIPQVITPEEIVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETR
SAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYVPEVTGTHKVTVLFAGQHIAKSPFEVYVDKSQGDASKVTAQGPGLEPSGNIANKTTYFEIFTAGAG
MGEVEVVIQDPTGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPAACRAIGRGLQPKGVRVKETADFKVYTKGAGSGEL
KVTVKGPKGEERVKQKDLGDGVYGFEYYPTIPGTYTVTITWGGQNIGRSPFEVKVGTECGNQKVRAWGPGLEGGIVGKSADFVVEAIGDDVGTLGFSVEGPSQ
AKIECDDKGDGSCDVRYWPQEAGEYAVHVLCNSEDIRLSPFMADIREAPQDFHPDRVKARGPGLEKTGVAVNKPAEFTVDAKHAGKAPLRVQVQDNEGCSVEA
TVKDNGNGTYSCSYVPRKPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKVYGPGVAKTGLKAHEPTYFTVDCTEAGQGDVSIGIKCAPGVVGPTEADIDF
DIIRNDNDTFTVKYTPCGAGSYTIMVLFADQATPTSPIRVKVEPSHDASKVKAEGPGLNRTGVELGKPTHFTVNAKTAGKGKLDVQFSGLAKGDAVRDVDIID
HHDNTYTVKYIPVQQGPVGVNVTYGGDHIPKSPFSVGVSPSLDLSKIKVSGLGDKVDVGKDQEFTVKSKGAGGQGKVASKIVSPSGAAVPCKVEPGLGADNSV
VRFVPREEGPYEVEVTYDGVPVPGSPFPLEAVAPTKPSKVKAFGPGLQGGNAGSPARFTIDTKGAGTGGLGLTVEGPCEAQLECLDNGDGTCSVSYVPTEPGD
YNINILFADTHIPGSPFKAHVAPCFDASKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSEAGLPAEVYIQDHGDGTHTITYIPLCPGAYTVTIKYG
GQPVPNFPSKLQVEPAVDTSGVQCYGPGIEGQGVFREATTEFSVDARALTQTGGPHVKARVANPSGNLTDTYVQDCGDGTYKVEYTPYEEGVHSVDVTYDGSP
VPSSPFQVPVTEGCDPSRVRVHGPGIQSGTTNKPNKFTVETRGAGTGGLGLAVEGPSEAKMSCMDNKDGSCSVEYIPYEAGTYSLNVTYGGHQVPGSPFKVPV
HDVTDASKVKCSGPGLSPGMVRANLPQSFQVDTSKAGVAPLQVKVQGPKGLVEPVDVVDNADGTQTVNYVPSREGSYSISVLYGEEEVPRSPFKVKVLPTHDA
SKVKASGPGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGKPKKTHIQDNHDGTYTVAYVPDVPGRYTILIKYGGDEIPFSPYRVRAVPTGDASKCTVT
VSIGGHGLGAGIGPTIQIGEETVITVDTKAAGKGKVTCTVCTPDGSEVDVDVVENEDGTFDIFYTAPQPGKYVICVRFGGEHVPNSPFQVTALAGDQPTVQTP
LRSQQLAPQYNYPQGSQQTWIPERPMVGVNGLDVTSLRPFDLVIPFTIKKGEITGEVRMPSGKVAQPSITDNKDGTVTVRYSPSEAGLHEMDIRYDNMHIPGS
PLQFYVDYVNCGHITAYGPGLTHGVVNKPATFTVNTKDAGEGGLSLAIEGPSKAEISCTDNQDGTCSVSYLPVLPGDYSILVKYNDQHIPGSPFTARVTGDDS
MRMSHLKVGSAADIPINISETDLSLLTATVVPPSGREEPCLLKRLRNGHVGISFVPKETGEHLVHVKKNGQHVASSPIPVVISQSEIGDASRVRVSGQGLHEG
HTFEPAEFIIDTRDAGYGGLSLSIEGPSKVDINTEDLEDGTCRVTYCPTEPGNYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANIGSHCDLS
LKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYCIRFVPAEMGMHTVSVKYKGQHVPGSPFQFTVGPLGEGGAHKVRAGGPGLERAEVGVPAEFGIWTREA
GAGGLAIAVEGPSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNEEHIPDSPFVVPVASPSGDARRLTVSSLQESGLKVNQPASFAVSLNGAKGAIDAKVH
SPSGALEECYVTEIDQDKYAVRFIPRENGIYLIDVKFNGTHIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGSPAEFIVNTSNAGAGALSVTIDGPSKV
KMDCQECPEGYRVTYTPMAPGSYLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKVATVPQHATSGPGPADVSKVVAKGLGLSKAYVGQK
SNFTVDCSKAGNNMLLVGVHGPRTPCEEILVKHMGSRLYSVSYLLKDKGEYTLVVKWGDEHIPGSPYRIMVP
```

Figure 4: Alineamineto Q8BTM8 - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

- Se debe tomar en cuenta que la matriz mostrada en la Figura2 está comprimida debido a la volumen de datos en los archivos fasta.
- En consecuencia podemos visulizar el resultado del alineamineto de ambas secuencias de Filamina A, tanto para el Humano como para el ratón en la Figura3 y la Figura4 respectivamente.

3 Actividad 3

Evalue la pregunta anterior con otros valores de *gapOpen*, *apEXTEND*, *identicalMatch* y *mismatch*. Verifique si obtiene el mismo alineamiento y comente sus resultados.

Resolución

3.1 Codigo

En el codigo cambiaremos los valores gapOPEN, gapEXTEND, identicalMatch, mismatch.

3.2 Resutados

- Primero visualizaremos la matriz generada

```
C:\Users\51931\Downloads>python ejercicio2.py P21333.fasta Q8BTM8.fasta -3
Matriz:

[[    0    -3    -6 ... -7935 -7938 -7941]
 [   -3     4     1 ... -7928 -7931 -7934]
 [   -6     1     8 ... -7921 -7924 -7927]
 ...
 [-7935 -7928 -7921 ... 10098 10095 10092]
 [-7938 -7931 -7924 ... 10102 10102 10099]
 [-7941 -7934 -7927 ... 10099 10099 10106]]
```

Figure 5: Matriz con gapOpen = gapEXTEND = -3

- Ahora las dos secuencias alineadas

```
Secuencia Alineada 1 :

MSSSHSRAGQSAAGAAPGGGVDTRDAEMPATEKDLAEDAPWKKIQQNTFTRWCNEHLKCVSKRIANLQTDLSDGLRLIALLEVLSQKKMHRKHNQRPTFRQMQLENVSVALEFLDRESIKLVS
IDSKAIVDGNLKLILGLIWTLILHYSISMPMWDEEEDEEAKKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCAPGLCPDWDSWDASKPVTNAREAMQQADDWLGIPQVITPEE
IVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRSAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYVPEVTGTHKVTVLFAGQHIAKSP
FEVYVDKSQGDASKVTAQGPGLEPSGNIANKTTYFEIFTAGAGTGEVEVVIQDPMGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPSACRAVGRGLQP
KGVRVKETADFKVYTKGAGSGELKVTVKGPKGEERVKQKDLGDGVYGFEYYPMVPGTYIVTITWGGQNIGRSPFEVKVGTECGNQKVRAWGPGLEGGVVGKSADFVVEAIGDDVGTLGFSVEG
PSQAKIECDDKGDGSCDVRYWPQEAGEYAVHVLCNSEDIRLSPFMADIRDAPQDFHPDRVKARGPGLEKTGVAVNKPAEFTVDAKHGGKAPLRVQVQDNEGCPVEALVKDNGNGTYSCSYVPR
KPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKVYGPGVAKTGLKAHEPTYFTVDCAEAGQGDVSIGIKCAPGVVGPAEADIDFDIIRNDNDTFTVKYTPRGAGSYTIMVLFADQATPTSP
IRVKVEPSHDASKVKAEGPGLSRTGVELGKPTHFTVNAKAAGKGKLDVQFSGLTKGDAVRDVDIIDHHDNTYTVKYTPVQQGPVGVNVTYGGDPIPKSPFSVAVSPSLDLSKIKVSGLGEKVD
VGKDQEFTVKSKGAGGQGKVASKIVGPSGAAVPCKVEPGLGADNSVVRFLPREEGPYEVEVTYDGVPVPGSPFPLEAVAPTKPSKVKAFGPGLQGGSAGSPARFTIDTKGAGTGGLGLTVEGP
CEAQLECLDNGDGTCSVSYVPTEPGDYNINILFADTHIPGSPFKAHVVPCFDASKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSEAGLPAEVYIQDHGDGTHTITYIPLCPGAYT
VTIKYGGQPVPNFPSKLQVEPAVDTSGVQCYGPGIEGQGVFREATTEFSVDARALTQTGGPHVKARVANPSGNLTETYVQDRGDGMYKVEYTPYEEGLHSVDVTYDGSPVPSSPFQVPVTEGC
DPSRVRVHGPGIQSGTTNKPNKFTVETRGAGTGGLGLAVEGPSEAKMSCMDNKDGSCSVEYIPYEAGTYSLNVTYGGHQVPGSPFKVPVHDVTDASKVKCSGPGLSPGMVRANLPQSFQVDTS
KAGVAPLQVKVQGPKGLVEPVDVVDNADGTQTVNYVPSREGPYSISVLYGDEEVPRSPFKVKVLPTHDASKVKASGPGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGKPKKTHIQDN
HDGTYTVAYVPDVTGRYTILIKYGGDEIPFSPYRVRAVPTGDASKCTVTVSIGGHGLGAGIGPTIQIGEETVITVDTKAAGKGKVTCTVCTPDGSEVDVDVVENEDGTFDIFYTAPQPGKYVI
CVRFGGEHVPNSPFQVTALAGDQPSVQPPLRSQQLAPQYTYAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIPFTIKKGEITGEVRMPSGKVAQPTITDNKDGTVTVRYAPSEAGLHEMDIR
YDNMHIPGSPLQFYVDYVNCGHVTAYGPGLTHGVVNKPATFTVNTKDAGEGGLSLAIEGPSKAEISCTDNQDGTCSVSYLPVLPGDYSILVKYNEQHVPGSPFTARVTGDDSMRMSHLKVGSA
ADIPINISETDLSLLTATVVPPSGREEPCLLKRLRNGHVGISFVPKETGEHLVHVKKNGQHVASSPIPVVISQSEIGDASRVRVSGQGLHEGHTFEPAEFIIDTRDAGYGGLSLSIEGPSKVD
INTEDLEDGTCRVTYCPTEPGNYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANVGSHCDLSLKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYCIRFVPAEMGTHTVSV
KYKGQHVPGSPFQFTVGPLGEGGAHKVRAGGPGLERAEAGVPAEFSIWTREAGAGGLAIAVEGPSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNEEHIPDSPFVVPVASPSGDARRLTV
SSLQESGLKVNQPASFAVSLNGAKGAIDAKVHSPSGALEECYVTEIDQDKYAVRFIPRENGVYLIDVKFNGTHIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGNPAEFVVNTSNAGAG
ALSVTIDGPSKVKMDCQECPEGYRVTYTPMAPGSYLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTK-AT-VPQH-ATSGPGPADVSKVVAKGLGLSKAYVGQKSNFTVD
CSKAGNNMLLVGVHGPRTPCEEILVKHMGSRLYSVSYLLKDKGEYTLVVKWGDEHIPGSPYRIMVP
```

Figure 6: Alineamiento P21333 - gapOpen = gapEXTEND = -3, identicalMatch = 4 y mismatch = -3

```
Secuencia Alineada 2:

MSSSHSRCGQSAAVASPGGSIDSRDAEMPATEKDLAEDAPWKKIQQNTFTRWCNEHLKCVSKRIANLQTDLSDGLRLIALLEVLSQKKMHRKHNQRPTFRQMQLENVSVALEFLDRESIKLVS
IDSKAIVDGNLKLILGLIWTLILHYSISMPMWDEEEDEEAKKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCAPGLCPDWDSWDASKPVNNAREAMQQADDWLGIPQVITPEE
IVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRSAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYVPEVTGTHKVTVLFAGQHIAKSP
FEVYVDKSQGDASKVTAQGPGLEPSGNIANKTTYFEIFTAGAGMGEVEVVIQDPTGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPAACRAIGRGLQP
KGVRVKETADFKVYTKGAGSGELKVTVKGPKGEERVKQKDLGDGVYGFEYYPTIPGTYTVTITWGGQNIGRSPFEVKVGTECGNQKVRAWGPGLEGGIVGKSADFVVEAIGDDVGTLGFSVEG
PSQAKIECDDKGDGSCDVRYWPQEAGEYAVHVLCNSEDIRLSPFMADIREAPQDFHPDRVKARGPGLEKTGVAVNKPAEFTVDAKHAGKAPLRVQVQDNEGCSVEATVKDNGNGTYSCSYVPR
KPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKVYGPGVAKTGLKAHEPTYFTVDCTEAGQGDVSIGIKCAPGVVGPTEADIDFDIIRNDNDTFTVKYTPCGAGSYTIMVLFADQATPTSP
IRVKVEPSHDASKVKAEGPGLNRTGVELGKPTHFTVNAKTAGKGKLDVQFSGLAKGDAVRDVDIIDHHDNTYTVKYIPVQQGPVGVNVTYGGDHIPKSPFSVGVSPSLDLSKIKVSGLGDKVD
VGKDQEFTVKSKGAGGQGKVASKIVSPSGAAVPCKVEPGLGADNSVVRFVPREEGPYEVEVTYDGVPVPGSPFPLEAVAPTKPSKVKAFGPGLQGGNAGSPARFTIDTKGAGTGGLGLTVEGP
CEAQLECLDNGDGTCSVSYVPTEPGDYNINILFADTHIPGSPFKAHVAPCFDASKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSEAGLPAEVYIQDHGDGTHTITYIPLCPGAYT
VTIKYGGQPVPNFPSKLQVEPAVDTSGVQCYGPGIEGQGVFREATTEFSVDARALTQTGGPHVKARVANPSGNLTDTYVQDCGDGTYKVEYTPYEEGVHSVDVTYDGSPVPSSPFQVPVTEGC
DPSRVRVHGPGIQSGTTNKPNKFTVETRGAGTGGLGLAVEGPSEAKMSCMDNKDGSCSVEYIPYEAGTYSLNVTYGGHQVPGSPFKVPVHDVTDASKVKCSGPGLSPGMVRANLPQSFQVDTS
KAGVAPLQVKVQGPKGLVEPVDVVDNADGTQTVNYVPSREGSYSISVLYGEEEVPRSPFKVKVLPTHDASKVKASGPGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGKPKKTHIQDN
HDGTYTVAYVPDVPGRYTILIKYGGDEIPFSPYRVRAVPTGDASKCTVTVSIGGHGLGAGIGPTIQIGEETVITVDTKAAGKGKVTCTVCTPDGSEVDVDVVENEDGTFDIFYTAPQPGKYVI
CVRFGGEHVPNSPFQVTALAGDQPTVQTPLRSQQLAPQYNYPQGSQQTWIPERPMVGVNGLDVTSLRPFDLVIPFTIKKGEITGEVRMPSGKVAQPSITDNKDGTVTVRYSPSEAGLHEMDIR
YDNMHIPGSPLQFYVDYVNCGHITAYGPGLTHGVVNKPATFTVNTKDAGEGGLSLAIEGPSKAEISCTDNQDGTCSVSYLPVLPGDYSILVKYNDQHIPGSPFTARVTGDDSMRMSHLKVGSA
ADIPINISETDLSLLTATVVPPSGREEPCLLKRLRNGHVGISFVPKETGEHLVHVKKNGQHVASSPIPVVISQSEIGDASRVRVSGQGLHEGHTFEPAEFIIDTRDAGYGGLSLSIEGPSKVD
INTEDLEDGTCRVTYCPTEPGNYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANIGSHCDLSLKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYCIRFVPAEMGMHTVSV
KYKGQHVPGSPFQFTVGPLGEGGAHKVRAGGPGLERAEVGVPAEFGIWTREAGAGGLAIAVEGPSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNEEHIPDSPFVVPVASPSGDARRLTV
SSLQESGLKVNQPASFAVSLNGAKGAIDAKVHSPSGALEECYVTEIDQDKYAVRFIPRENGIYLIDVKFNGTHIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGSPAEFIVNTSNAGAG
ALSVTIDGPSKVKMDCQECPEGYRVTYTPMAPGSYLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKVAT-VPQH-ATSGPGPADVSKVVAKGLGLSKAYVGQKSNFTVD
CSKAGNNMLLVGVHGPRTPCEEILVKHMGSRLYSVSYLLKDKGEYTLVVKWGDEHIPGSPYRIMVP
```

Figure 7: Alineamiento Q8BTM8 - gapOpen = gapEXTEND = -3, identicalMatch = 4 y mismatch = -3

- Observamos el dotmacher de cada uno

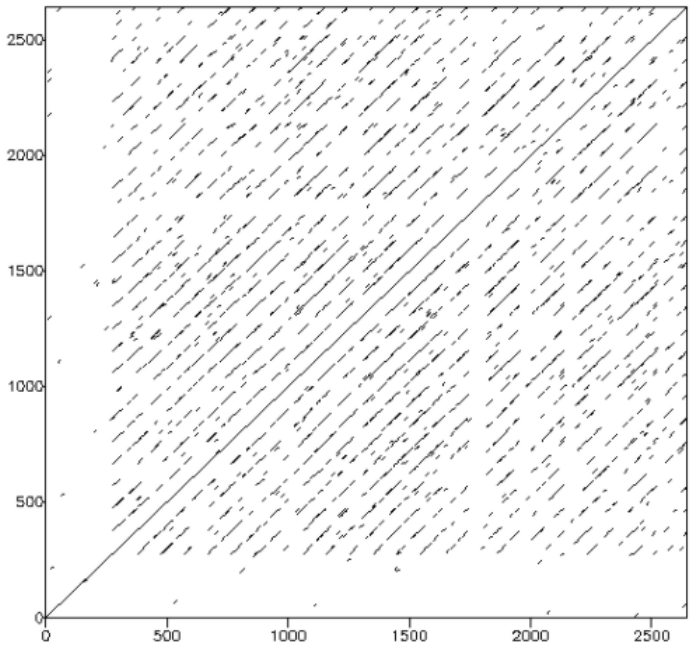

Figure 8: Alineamiento Global del Ejercicio 2

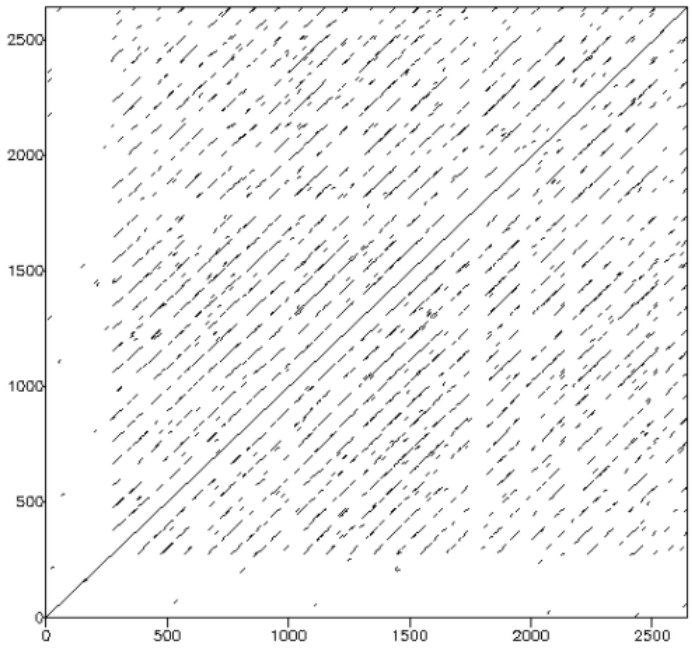

Figure 9: Alineamiento global del Ejercicio 3

- A simple vista parecen iguales, entonces los analizamos un poco más.

```
THIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGNPAEFVVNTSNAGAGALSVTIDGPSKVKMDCQECPEGYRVTYTPMAPGS
YLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKATCAPQHGAPGPGPADASKVVAKGLGLSKAYVGQKSSFTV
DCSKAGNNMLLVGVHGPRTPCEEILVKHVGSRLYSVSYLLKDKGEYTLVVKWGDEHIPGSPYRVVVP
```

Figure 10: Secuencia 1 de Alineamiento del Ejercicio 2

```
YLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTK-AT-VPQH-
ATSGPGPADVSKVVAKGLGLSKAYVGQKSNFTVDCSKAGNNMLLVGVHGPRTPCEEILVKHMGSRLYSVSYLLKDK-
GEYTLVVKWGDEHIPGSPYRIMVP
```

Figure 11: Secuencia 1 de alineamiento del Ejercicio 3

- Como se muestra en el imagen hay diferencias pocas pero las hay, lo mismo ocurre con la Secuenciación 2 de cada ejercicio

4 Conclusiones

1. Programando el Algoritmo Needleman-Wunsch se obtuvo una manera más rápida y practica de encontrar un alineamiento global , de esta manera se logra automatizar dicho proceso para **n**-casos de secuencias.

2. En términos de Programación Dinámica se usa un enfoque de abajo hacia arriba siendo de ayuda para los algoritmos de alineamiento de cadenas. ComputacionalmenteLa la complejidad del algoritmo(Needleman–Wunsch) para dos secuencias de longitud n y m es **O(mn)**.

3. Se probó el algoritmo con secuencias reales de proteínas y nos dió como resultado nuestra matriz y las dos secuencias de alineamiento global, lo cual serviria para futuros trabajos de procesamiento genómico, teniendo un correcto alineamiento a escala se puede usar la data para predicciones con LSTM.

4. Nos dimos cuenta que aunque podamos cambiar los valores (gapOPEN, gapEXTEND, identicalMatch, mismatch) las secuencias resultantes cambian, pero su cambio es mínimo(menos del 1% de la secuencia total), lo que nos da a decir que al alineamiento global se conserva en su mayoria.

5. Los valores *gapOPEN*, *gapEXTEND*, *identicalMatch*, *misMatch*, cada vez que sean variados, lo primero que cambiará será nuestra matriz y las secuencias resultantes seguirán sufriendo adulteraciones. El mínimo dependerá de los valores antes mencionados, y el cambio puede ser imperciptible.

6. Una desventaja que se puede tener con este algoritmo es que puede ocasionar muchos problemas si se busca es un alineamiento óptimo u constante.

5 Repositorio

La resolución de la práctica la encuentra en el siguiente enlace Practica_06