Tarea_2

December 13, 2022

1 Desarrollo Tarea 2 - Bioinformática

- Vicente Álvarez
- Iñaki Oyarzun M.

1.0.1 Repositorio

Se adjunta un repositorio que contiene los resultados y notebook para visualizar desde el navegador

Link

1.0.2 Preparación de librerias

Se procede con la instalacion de las librerias a utilizar como parte del notebook

[26]:
```
!pip install biopython
```

```
Requirement already satisfied: biopython in
c:\users\inaki\appdata\local\programs\python\python311\lib\site-packages (1.80)
Requirement already satisfied: numpy in
c:\users\inaki\appdata\local\programs\python\python311\lib\site-packages (from
biopython) (1.23.5)
```

[27]:
```
from Bio.Data.CodonTable import unambiguous_dna_by_id
from Bio import SeqIO
from random import choice
from Bio.Seq import Seq
from Bio.SeqRecord import SeqRecord
```

[28]:
```
def altcodons(codon, table):
    """List codons that code for the same aminonacid / are also stop.

    @param codon
    @table code table id
    @return list of codons

    """
    plus4 =␣
↪[["CGT","CGC","CGA","CGG","AGA","AGG"],["CTT","CTC","CTA","CTG","TTA","TTG"],["TCT","TCC","
```

```
    for l in plus4:
      if codon in l:
        return l

    tab = unambiguous_dna_by_id[table]

    if codon in tab.stop_codons:
        return tab.stop_codons

    try:
        aa = tab.forward_table[codon]
    except:
        return []

    return [
        k
        for (k, v) in tab.forward_table.items()
        if v == aa and k[0] == codon[0] and k[1] == codon[1]
    ]


def degeneration(codon, table):
    """Determine how many codons code for the same amino acid / are also stop

    @param codon the codon
    @param table code table id
    @param the number of codons also coding for the amino acid codon codes for

    """
    return len(altcodons(codon, table))


def is_x_degenerated(x, codon, table):
    """Determine if codon is x-fold degenerated.

    @param codon the codon
    @param table code table id
    @param true if x <= the degeneration of the codon

    """
    return x <= len(altcodons(codon, table))


def degenerated_subseq(seq, x, table):
    """Get a subsequence consisting of the x-fold degenerated codons only."""
    data = ""
    for i in range(0, len(seq), 3):
```

```
        codon = seq[i : i + 3].tostring()
        if is_x_degenerated(x, codon, table):
            data += codon
    return data
```

---

2 Pregunta 1

2.0.1 a)

Para el alineamiento de las secuencias, se vuelven a generar el blasting basado en los datos indicados en la tarea anterior, se indica en la siguiente foto los resultados junto a los rangos en los que se

TCP-1/cpn60 chaperonin family protein [Candidatus Woesearchaeota archaeon]

Sequence ID: MBS3131989.1 Length: 485 Number of Matches: 1

Range 1: 230 to 241 GenPept Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Identities	Positives	Gaps
33.7 bits(72)	9.1	11/12(92%)	11/12(91%)	0/12(0%)

```
Query  1    VINAKIALVARE  12
            V NAKIALVARE
Sbjct  230  VKNAKIALVARE  241
```

Download GenPept Graphics

FGGY-family carbohydrate kinase [Shinella sp. AETb1-6]

Sequence ID: WP_160871018.1 Length: 451 Number of Matches: 1

See 1 more title(s) See all Identical Proteins(IPG)

Range 1: 6 to 23 GenPept Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Identities	Positives	Gaps
33.7 bits(72)	9.1	12/18(67%)	13/18(72%)	5/18(27%)

```
Query  1   VI-----NAKIALVARES  13
           VI     NAK+ALVARES
Sbjct  6   VIDIGKTNAKVALVARES  23
```

Download GenPept Graphics

FGGY-family carbohydrate kinase [Shinella sumterensis]

Sequence ID: WP_134650126.1 Length: 451 Number of Matches: 1

See 2 more title(s) See all Identical Proteins(IPG)

Range 1: 6 to 23 GenPept Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Identities	Positives	Gaps
33.7 bits(72)	9.1	12/18(67%)	13/18(72%)	5/18(27%)

```
Query  1   VI-----NAKIALVARES  13
           VI     NAK+ALVARES
Sbjct  6   VIDIGKTNAKVALVARES  23
```

Download GenPept Graphics

T9SS-dependent M36 family metallopeptidase [Saprospiraceae bacterium]

Sequence ID: MBK9253950.1 Length: 941 Number of Matches: 1

Range 1: 499 to 512 GenPept Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Identities	Positives	Gaps
32.9 bits(70)	18	9/14(64%)	11/14(78%)	0/14(0%)

```
Query  1    VINAKIALVARESU  14
            VIN KIAL+ R S+
Sbjct  499  VINSKIALIDRGSC  512
```

Download GenPept Graphics

hypothetical protein [Xanthomonadales bacterium]

Sequence ID: MCB1584601.1 Length: 494 Number of Matches: 1

Range 1: 136 to 149 GenPept Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Identities	Positives	Gaps
32.9 bits(70)	18	9/14(64%)	11/14(78%)	0/14(0%)

```
Query  2    INAKIALVARESUN  15
            IN KIAL+ R S+N
Sbjct  136  INGKIALIERGSCN  149
```

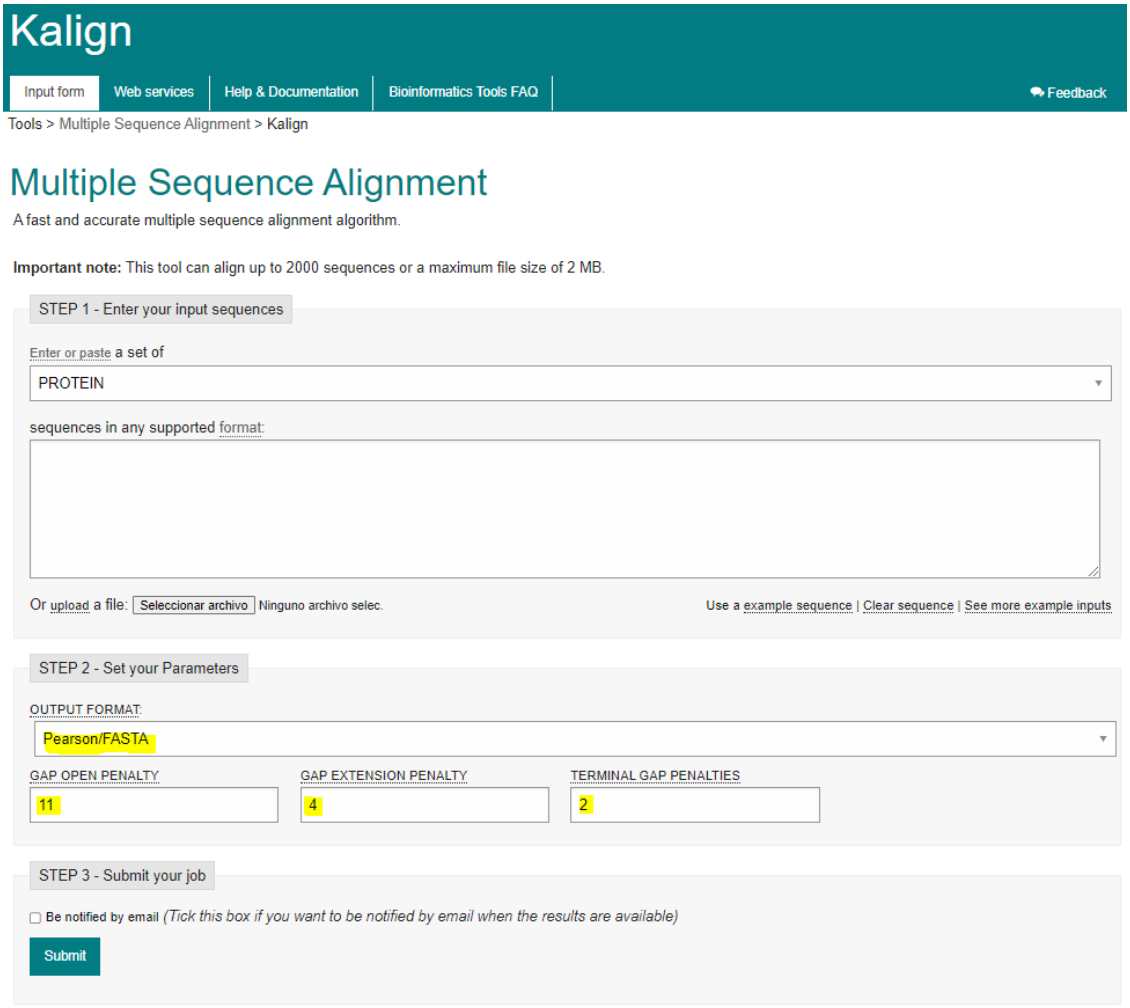

encontró.

Resultados Blasting

Luego se decide hacer uso de Kalign, con la siguiente configuración y archivo unificado de las primeras 5 secuencias que entrega BLAST:

```
>MBS3131989.1 TCP-1/cpn60 chaperonin family protein [Candidatus Woesearchaeota archaeon]
MGQQVQPIFILPEGSQRSIGRDAQRTNIMAARLVAETVRTTLGPKGMDKMIVDSLGDVTVTNDGVTILKEMNIEHPSAKM
IVEVAKTQEDEVGDGTTTAVVLAGELLKNAEDLLEQEVHPAVIARGYRLAETKAQQLLNEMAEKITPNDTEILKKIAVTA
MTGKGAEYSKEKLSSLAVEAVRSITDADGSIDKDNIKIEKRTGDSVENSELIKGIVIDKERLSSSMPRVVKNAKIALVAR
EIEIKKTEVDAKIEITNPDQLQAFLDQEEQMLKGMVDRIAATGANVMICQKGIDDLAIHFLAKAGIYAVRRASESDMKKI
ARATGGKIMTSIKEISKDDLGFAGAVEEKKVGDEEFTYIMECKNPKAVTIMVRGGTEHVTAEIERAVTDAVGDVAAAIKS
GKVVAGAGAPEIELAMGLRKFAESLSGREQLAVNAFADAMEVIPRTLVENAGLDPIDTLTDLKAAHAKKQKWAGIDVFSG
KVMDA
>WP_160871018.1 FGGY-family carbohydrate kinase [Shinella sp. AETb1-6]
MRHIAVIDIGKTNAKVALVARESLTEVAVMRRPNAVLKGPPYPHYDIEGLWDFILDALRRFQAQFGVDAISMTTHGASAV
LLDDEGKLAAPALDYEFDGPDGLAAEYDAARPPFAETGSPRLPLGLNLGAQLFWLFRTVPGLLERTRTILTYPQYWAFRL
SGVAVNEVTSLGCHTDLWDPEAGRYSTLVEREGWAVLMAPVVRASDRLGPVLPQIASATGLAPATPVICGIHDSNASLYA
HLVARDEPFAVVSTGTWVISMAMGGAKVTLDPARDTLVNVNAHGDAVPSARFMGGREFERLIGDARNAHAAADVDAVLAR
GVMLLPAVENQSGPFQGRKAEWTVDAATLAPAERFVVVSFYLALMTATGLEIVGAQGPVLVEGPFAQNAAYLDMLAASTG
RPVEAVAGTGTSVGAALLDADGRPPSTDRANPRPRHEAALRRYASAWQARL
>WP_134650126.1 FGGY-family carbohydrate kinase [Shinella sumterensis]
MRHIAVIDIGKTNAKVALVARESLTEVAVMRRPNAVLKGPPYPHYDIEGLWDFILDALRRFQAQFGVDAISMTTHGASAV
LLDDEGKLAAPALDYEFDGPDSLAAEYDAARPPFAETGSPRLPLGLNLGAQLFWLFKTVPGLLERTRTILTYPQYWAFRL
SGVAVNEVTSLGCHTDLWDPEAGRYSTLVEREGWAALMAPVVRASNRLGPVLPQIASATGLAPATPVICGIHDSNASLYA
HLVARDEPFAVVSTGTWVISMAMGGAKVTLDPARDTLVNVNAHGDAVPSARFMGGREFERLIGDARNAHAAADVDAVLAR
GVMLLPAVENQSGPFQERKAEWTVDAATLAPAERFVAVSFYLALMTATGLEIVGAQGPVLVEGPFAQNAAYLDMLAASTG
RPVEAVAGTGTSVGAALLDADGRPPATGRANPRPRHEAALRRYASAWQARL
>MBK9253950.1 T9SS-dependent M36 family metallopeptidase [Saprospiraceae bacterium]
MQVKGFYSNGKKYIIMLSVWNLNFRFLKFSCYLYILIFIQTLNLKAQDLNEIYLLHLSELKSVNNSDDFDDYLISNQHTS
KRSGLLHTYIQQRYQGITVRDGVLSIHQKSDGSLTQLNDQFIRNLSSKINSEVPLLSSLECLLKIANDFGYNYTLPIQTI
EKKETQDQQTVFAKSGISTEDIPARLIYFPINENQLKLSWEFFIHETDASNWWQIVIDAETGDILERRNLYLNCDFGHPD
YNNQICSHKALKHFRMPGIYDVTESESETISNAYRVYPLGVESPSHGGRVLVINPADSIASPYGWHDTNGVTGAEHTTTK
GNNVEAREDIDGNNNTLGAMAQGGPDLVFDFPIDFTQQPAVSQNAAITNLFYWNNVIHDVFYQYGFDEPAGNFQQNNYGN
GGLANDFVRADALDGESTNNANFATPPDGNNPRMQMFLWNPGSGITFTVNTPISVSGTYGASKAAFGSQTYSVSGDVVIA
NDGTGNPTLACNALTNGAVINSKIALIDRGSCEFGAKCLNAQNAGAIAVIVCNNVAGDPFSMGAGAVGNQVTIPCVMISQ
SNCNTLRLQIPGLTVSMTGTQSIQIDGDYDNCIITHEYGHGISIRLTGGAGNAGCLNNQEQMGEGWSDWFGLMLTMNESD
IGSTPRGIGNYVISQPSNGNGIRPYPYTTNMSINPHTYDAIKTASIPHGIGSVWCAMLWELTWSLIAEYGFDPDIYRGTG
GNNIALTLVTEALKLQPCSPGFVTGRDAILQADLALYGGIHQCMIWEAFAKRGLGFSAVQGSSNNRSDGTQAFNLPPSCC
KYVRNTNNSGDYSLRSAINCAVAGDTIRFSPLLIGQAISLTEGPIVLNKNLNFKTRNPDAIRILAPESSPAFHIEENIEI
FIEKLGLTGAQSGQIRSVINNGNLTLKDVIVRDKVLEDIGHIIKNEGQLTFEGSSAILKVN
>MCB1584601.1 hypothetical protein [Xanthomonadales bacterium]
LANTDNNDFIYVNRMARISGSNAFLAATQTGIFKTTDGGTSWNEVSHFDTNGRGFVDLKVDPTNPNHLLAYHFGDGNIEV
VNMHISTPANLTGNYPVIPAAFGPEIPDSGVSGEIILVADDTNPTDDGCETIQNDINGKIALIERGSCNFTVKVINAQNA
GAIAVAVFNNVADELFTMGGDDPDINIPAMMITKGLGDRIKASIAPVQVNIQIDEEVQLNNFVMRSTNGGDFWAILDNHG
LPDLDVERMEIAFGSDGKTYIAASTVPEEVDNQQTPGTLGLYRSTGAGNTQFEKTDSDTNFIERQGWYDLAIAVNPDDSN
HVIMGAVDQYATHDGGTTIDRKTWWFSRSGFLAQYIHADHHGYFFSPHSSDHIYTASDGGISKSEDGGESWFHRNNGLNI
SQSYGIAVSPDGQQVTSGTQDNGSQLFFGDQQAWLEWQGGDGGFSGWDQQQGQYVYGSYVEGQLYGSNNGGYSAQAMELP
DTEGARFIQPFVLD
```

Alineamiento



Configuración Kalign

De los desarrollos realizados, Kalign entrega los siguientes resultados:

```
>MBS3131989.1
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
--------------MGQQVQPIFILPEGSQRSIGRDAQRTNIMAARLVAETVRTTLGP--
--KGMDKMIVDSLGDVTVTNDGVTILKEMNIEHPSAKMIVEVAKTQEDEVGDGTTTAVVL
AGELLKNAEDLLEQEVHP-AVIARGYRLAETKAQQL---LNEMAEKITPNDT-EILKK--
IAVTAMTGKGAEYSKEKLSSLAVEAVRSI---TDA-DGSIDKDNIKIEKRTGDSVENSEL
IKGIVIDKE-----RLSSSMPRVVKNAKIALVAREI-EIKKTEVDAKIEIT--NPDQLQA
FLDQEEQMLKGMVDRIAATG--ANVMICQKGIDDLA---IHFLAKAGIYAVRRASESD--
MKK-IARATGGKIMTSIKEISKDDLGFA---GAVEEKKVGDEEFTYIME-----CKNPKA
VTIMVRGGTEH-VTAEIERAVTDAVGDVAAAIKSGKVVAGAGAPEIELAMGLRKFAESLS
GREQLAVNAFADAMEVIPRTLVENAGLDPIDTLTDLKAA----H-AKKQKWAGIDVFSGK
VMDA---------
```

Alineamiento proteína 1

```
>WP_160871018.1
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
-------------M-RHIAVIDIGKTNAKVALVARESLTEVAVMRRPNAVLKGPPYPHY
DIEGLWDFILDALRRFQA-QFGVDAI-SMTTHGASAVLL-------DDE---GKLAAPAL
DYEF--DGPDGLAAEYDA-AR----PPFAETGSPRLPLGLNLGAQLFWLFRTVPGLLE--
RTRTILTYP--QYWAFRLSGVAVNEVTSLGCHTDLWDPEAGRYSTLVE-REGWAV----L
MAPVVRASD-----RLGPVLPQIASATGLAPATPVICGIHDSNASLYAHLV--ARDEPFA
VVSTGTWVIS-----MAMGG--AKVTL-DPARDTLV---NVNAHGDAVPSARFMGGRE--
FERLIGDARNAHAAADVDAVLARGVMLL---PAVENQS-G--PFQGRKA-----EWTVDA
ATLAP--AERF-VVVSFYLALMTATGLEIVGAQGPVLVEG---PFAQNAAYLDMLAAS-T
GRP---VEAVAGTGTSVGAALLDADGRPPSTDRANPRPR----HEAALRRYA--SAWQAR
L------------
```

Alineamiento proteína 2

```
>WP_134650126.1
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
------------------------------------------------------------
-------------M-RHIAVIDIGKTNAKVALVARESLTEVAVMRRPNAVLKGPPYPHY
DIEGLWDFILDALRRFQA-QFGVDAI-SMTTHGASAVLL-------DDE---GKLAAPAL
DYEF--DGPDSLAAEYDA-AR----PPFAETGSPRLPLGLNLGAQLFWLFKTVPGLLE--
RTRTILTYP--QYWAFRLSGVAVNEVTSLGCHTDLWDPEAGRYSTLVE-REGWAA----L
MAPVVRASN-----RLGPVLPQIASATGLAPATPVICGIHDSNASLYAHLV--ARDEPFA
VVSTGTWVIS-----MAMGG--AKVTL-DPARDTLV---NVNAHGDAVPSARFMGGRE--
FERLIGDARNAHAAADVDAVLARGVMLL---PAVENQS-G--PFQERKA-----EWTVDA
ATLAP--AERF-VAVSFYLALMTATGLEIVGAQGPVLVEG---PFAQNAAYLDMLAAS-T
GRP---VEAVAGTGTSVGAALLDADGRPPATGRANPRPR----HEAALRRYA--SAWQAR
L------------
```

Alineamiento proteína 3

Alineamiento proteína 4

Alineamiento proteína 5

Notar que de las 5 proteinas alineadas, solo la proteina 1, 4 y 5 contienen los segmentos con el nombre ingresado (subrayado). En cambio las otras no se encontraban como parte del alineamiento.

2.0.2 b)

Para este caso, se hace uso de los siguientes segmentos (Obtenidos de la imagen anterior de BLAST):

- Proteina 1: VKNAKIALVARE
- Proteina 2: VIDIGKTNAKVALVARES
- Proteina 3: VIDIGKTNAKVALVARES
- Proteina 4: VINSKIALIDRGSC
- Proteina 5: INGKIALIERGSCN

Al alinearlos nuevamente utilizando Kalign, se obtiene lo siguiente:

```
>MBS3131989.1
-----VKNAKIALVARE---
>WP_160871018.1
VIDIGKTNAKVALVARES--
>WP_134650126.1
VIDIGKTNAKVALVARES--
>MBK9253950.1
------INSKIALIDRGSC-
>MCB1584601.1
------INGKIALIERGSCN
```

Alineamiento de las 5 proteínas

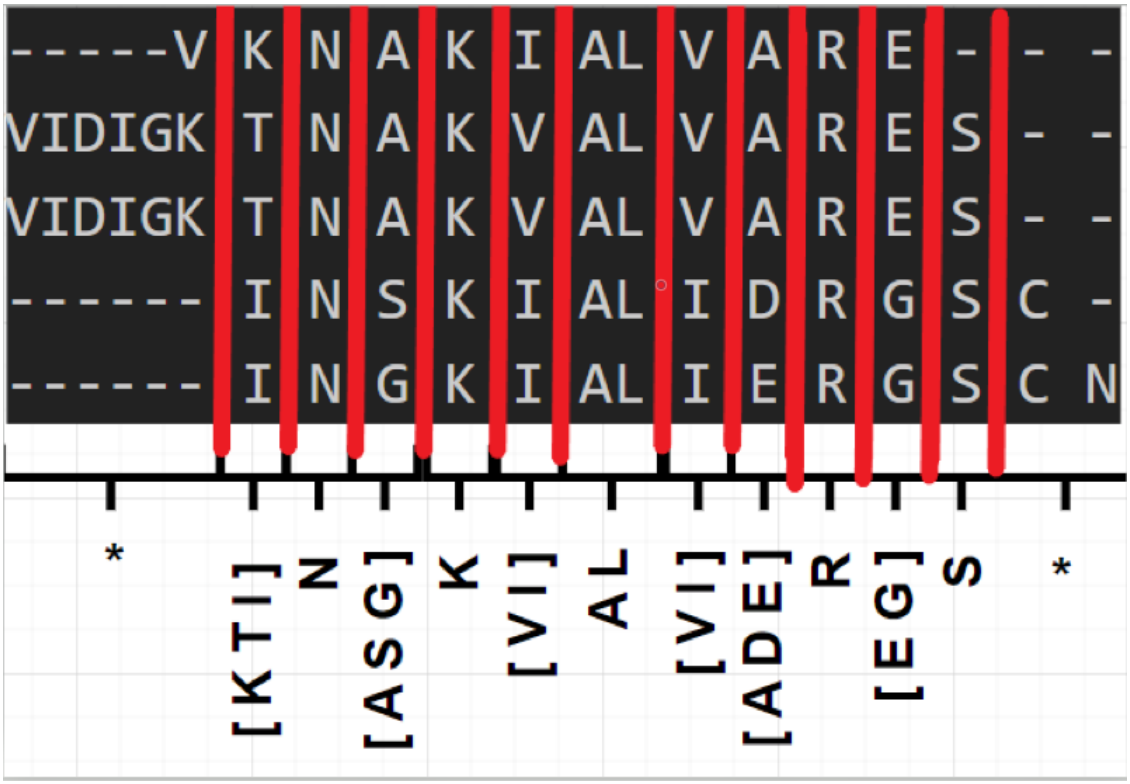
2.0.3 c)

Usando los alineamientos se construye y dibuja el HMM

Alineamiento

```
-----VKNAKIALVARE---
VIDIGKTNAKVALVARES--
VIDIGKTNAKVALVARES--
------INSKIALIDRGSC-
------INGKIALIERGSCN
```

Se plantea una expresión regular para estudiar los patrones de la siguiente manera:

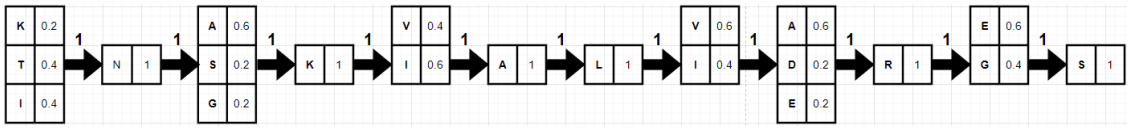


Expresion regular para estudio de patrones

Siendo de la siguiente forma:

* [KTI] N [ASG] K [VI] AL [VI] [ADE] R [EG] S *

Con ello se puede obtener y dibujar HMM, obteniendo lo siguiente:



HMM Desarrollado

2.0.4 d)

Para determinar la secuenca de estados internos mas probable se realiza para la emision del nombre basta con observar del modelo HMM que justamente aquellos estados con más probabilidad son los que benefician la creacion del nombre, por lo que, si se sigue la secuencia a partir de la seleccion del estado con mayor valor se obtiene que la secuencia mas probable para generar el nombre es:

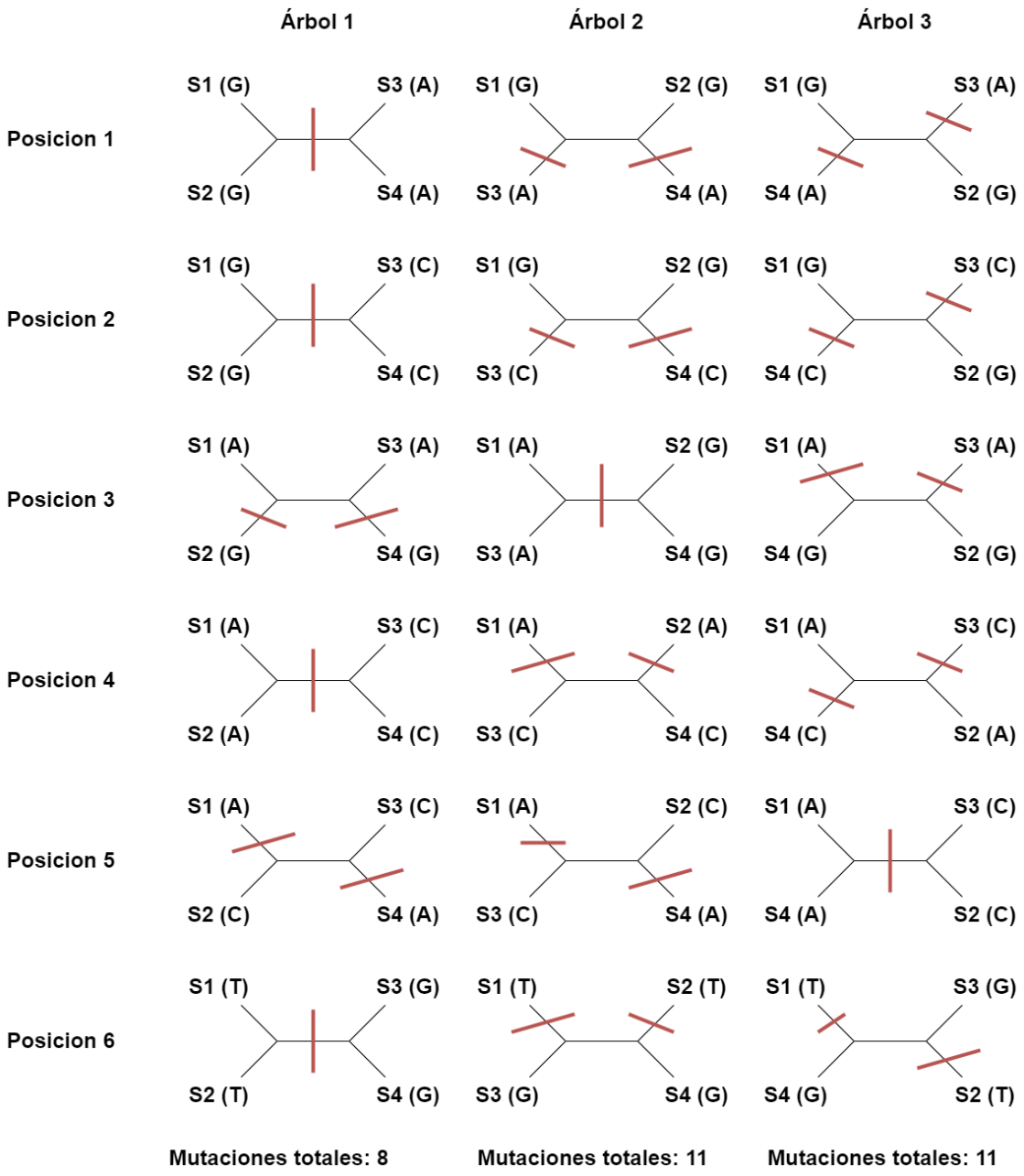
P(I N A K I A L V A R E S) $= 0.031e-2$

---

3 Pregunta 2

3.0.1 a)

Para este desarrollo, tal como se menciona el enunciado, se prodcede a realizar el método de máxima parsimonia de manera manual, habiendo identificado las posiciones 6 (1 en el dibujo), 7 (2 en el dibujo), 9 (3 en el dibujo), 10 (4 en el dibujo), 13 (5 en el dibujo) y 14 (6 en el dibujo). Generando con ello la siguiente serie de arboles sin enraizar según cada posición y cantidad de mutaciones:

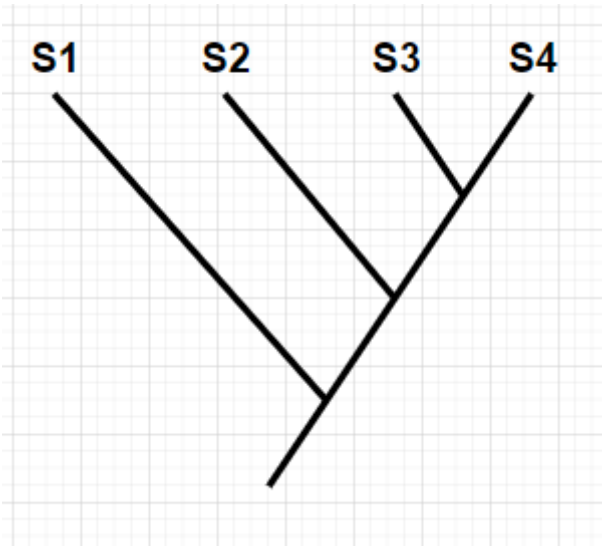


Desarrollo a mano de máxima parsimonia

Con esto se logra determinar con máxima parsimonia que el mejor árbol filogenético será el primero.

3.0.2 b)

Realizando un enraizado a partir de que se sabe el outgrup (S1), se llega a obtener el siguiente arbol con los parentezcos.



Desarrollo para enraizar sabiendo como outgrup S1

De lo que se observa que, en este caso S3 y S4 comparten un parentezco en mayor medida en comparación a S1 y S2, que quedan mas separados.

3.0.3 c)

Se puede inferir que la secuencia ancestral al menos tuvo 3 mutaciones las cuales hacen clave la diferenciación o creación de las secuencias 1, 2, 3 y 4.

---

4 Pregunta 3

4.0.1 a)

Para el desarrollo de esta pregunta, se toma por lo entendido en clase que los codones tanto de inicio y fin no son tomados en cuenta.

[43]:
```
def split_codon(seq):
  split_seq = [seq[i:i+3] for i in range(0, len(seq), 3)]
  return split_seq

def seq_rand(seq):
  """Randomiza los codónes de una secuencia codificadora por sus homologos

  @param seq la secuencia codificadora a randomizar (string)
  @param la secuencia seq randomizada
  """
  codones = split_codon(seq)
  for i in range(1, len(codones)):
    options = altcodons(codones[i], 1)
    codones[i] = choice(options)
```

```
  res = "".join(codones)
  return res
```

4.0.2 b)

[41]:
```
def nucleo_score(codons):
  best_codon = ""
  max_score = 0
  for c in codons:
    s = c.count('C')+c.count('G')
    if s >= max_score:
      best_codon = c
      max_score = s
  return best_codon

def seq_maxGC(seq):
  """Aumenta el porcentaje de bases G y C

  @param seq la secuencia codificadora a alterar (string)
  @param la secuencia seq maximizada para G y C
  """
  codones = split_codon(seq)
  for i in range(1, len(codones)):
    options = altcodons(codones[i], 1)
    b = nucleo_score(options)
    codones[i] = b
  res = "".join(codones)
  return res
```

4.0.3 c)

[46]:
```
unparsed = SeqIO.read('content/cds.fna','fasta')
sequence = str(unparsed.seq)

randomized_sequence = seq_rand(sequence)
maximized_sequence = seq_maxGC(sequence)

recR = SeqRecord(
    Seq(
        randomized_sequence,
    ),
)

recGC = SeqRecord(
    Seq(
        maximized_sequence,
```

```
    ),
)

print("Secuencia randomizada:\n",maximized_sequence)
print("\nSecuencia maximizada:\n",maximized_sequence)

SeqIO.write(recR, "content/cds_random.fna", "fasta")
SeqIO.write(recGC, "content/cds_maxGC.fna", "fasta")
```

```
Secuencia randomizada:
 ATGGACCTGAGCGCGCTGCGGGTGGAGGAGGTGCAGAACGTGATCAACGCGATGCAGAAGATCCTGGAGTGCCCGATCT
GCCTGGAGCTGATCAAGGAGCCGGTGAGCACGAAGTGCGACCACATCTTCTGCAAGTTCTGCATGCTGAAGCTGCTGAAC
CAGAAGAAGGGGCCGAGCCAGTGCCCGCTGTGCAAGAACGACATCACGAAGCGGAGCCTGCAGGAGAGCACGCGGTTCAG
CCAGCTGGTGGAGGAGCTGCTGAAGATCATCTGCGCGTTCCAGCTGGACACGGGGCTGGAGTACGCGAACAGCTACAACT
TCGCGAAGAAGGAGAACAACAGCCCCGGAGCACCTGAAGGACGAGGTGAGCATCATCCAGAGCATGGGGTACCGGAACCGG
GCGAAGCGGCTGCTGCAGAGCGAGCCGGAGAACCCGAGCCTGCAGGAGACGAGCCTGAGCGTGCAGCTGAGCAACCTGGG
GACGGTGCGGACGCTGCGGACGAAGCAGCGGATCCAGCCGCAGAAGACGAGCGTGTACATCGAGCTGGGGAGCGACAGCA
GCGAGGACACGGTGAACAAGGCGACGTACTGCAGCGTGGGGGACCAGGAGCTGCTGCAGATCACGCCGCAGGGGACGCGG
GACGAGATCAGCCTGGACAGCGCGAAGAAGGCGGCGTGCGAGTTCAGCGAGACGGACGTGACGAACACGGAGCACCACCA
GCCGAGCAACAACGACCTGAACACGACGGAGAAGCGGGCGGCGGAGCGGCACCCGGAGAAGTACCAGGGGAGCAGCGTGA
GCAACCTGCACGTGGAGCCGTGCGGGACGAACACGCACGCGAGCAGCCTGCAGCACGAGAACAGCAGCCTGCTGCTGACG
AAGGACCGGATGAACGTGGAGAAGGCGGAGTTCTGCAACAAGAGCAAGCAGCCGGGGCTGGCGCGGAGCCAGCACAACCG
GTGGGCGGGGAGCAAGGAGACGTGCAACGACCGGCGGACGCCGAGCACGGAGAAGAAGGTGGACCTGAACGCGGACCCGC
TGTGCGAGCGGAAGGAGTGGAACAAGCAGAAGCTGCCGTGCAGCGAGAACCCGCGGGACACGGAGGACGTGCCGTGGATC
ACGCTGAACAGCAGCATCCAGAAGGTGAACGAGTGGTTCAGCCGGAGCGACGAGCTGCTGGGGAGCGACGACAGCCACGA
CGGGGAGAGCGAGAGCAACGCGAAGGTGGCGGACGTGCTGGACGTGCTGAACGAGGTGGACGAGTACAGCGGGAGCAGCG
AGAAGATCGACCTGCTGGCGAGCGACCCGCACGAGGCGCTGATCTGCAAGAGCGAGCGGGTGCACAGCAAGAGCGTGGAG
AGCAACATCGAGGACAAGATCTTCGGGAAGACGTACCGGAAGAAGGCGAGCCTGCCGAACCTGAGCCACGTGACGGAGAA
CCTGATCATCGGGGCGTTCGTGACGGAGCCGCAGATCATCCAGGAGCGGCCGCTGACGAACAAGCTGAAGCGGAAGCGGC
GGCCGACGAGCGGGCTGCACCCGGAGGACTTCATCAAGAAGGCGGACCTGGCGGTGCAGAAGACGCCGGAGATGATCAAC
CAGGGGACGAACCAGACGGAGCAGAACGGGCAGGTGATGAACATCACGAACAGCGGGCACGAGAACAAGACGAAGGGGGA
CAGCATCCAGAACGAGAAGAACCCGAACCCGATCGAGAGCCTGGAGAAGGAGAGCGCGTTCAAGACGAAGGCGGAGCCGA
TCAGCAGCAGCATCAGCAACATGGAGCTGGAGCTGAACATCCACAACAGCAAGGCGCCGAAGAAGAACCGGCTGCGGCGG
AAGAGCAGCACGCGGCACATCCACGCGCTGGAGCTGGTGGTGAGCCGGAACCTGAGCCCGCCGAACTGCACGGAGCTGCA
GATCGACAGCTGCAGCAGCAGCGAGGAGATCAAGAAGAAGAAGTACAACCAGATGCCGGTGCGGCACAGCCGGAACCTGC
AGCTGATGGAGGGGAAGGAGCCGGCGACGGGGGCGAAGAAGAGCAACAAGCCGAACGAGCAGACGAGCAAGCGGCACGAC
AGCGACACGTTCCCGGAGCTGAAGCTGACGAACGCGCCGGGGAGCTTCACGAAGTGCAGCAACACGAGCGAGCTGAAGGA
GTTCGTGAACCCGAGCCTGCCGCGGGAGGAGAAGGAGGAGAAGCTGGAGACGGTGAAGGTGAGCAACAACGCGGAGGACC
CGAAGGACCTGATGCTGAGCGGGGAGCGGGTGCTGCAGACGGAGCGGAGCGTGGAGAGCAGCAGCATCAGCCTGGTGCCG
GGGACGGACTACGGGACGCAGGAGAGCATCAGCCTGCTGGAGGTGAGCACGCTGGGGAAGGCGAAGACGGAGCCGAACAA
GTGCGTGAGCCAGTGCGCGGCGTTCGAGAACCCGAAGGGGCTGATCCACGGGTGCAGCAAGGACAACCGGAACGACACGG
AGGGGTTCAAGTACCCGCTGGGGCACGAGGTGAACCACAGCCGGGAGACGAGCATCGAGATGGAGGAGAGCGAGCTGGAC
GCGCAGTACCTGCAGAACACGTTCAAGGTGAGCAAGCGGCAGAGCTTCGCGCCGTTCAGCAACCCGGGGAACGCGGAGGA
GGAGTGCGCGACGTTCAGCGCGCACAGCGGGAGCCTGAAGAAGCAGAGCCCGAAGGTGACGTTCGAGTGCGAGCAGAAGG
AGGAGAACCAGGGGAAGAACGAGAGCAACATCAAGCCGGTGCAGACGGTGAACATCACGGCGGGGTTCCCGGTGGTGGGG
CAGAAGGACAAGCCGGTGGACAACGCGAAGTGCAGCATCAAGGGGGGGAGCCGGTTCTGCCTGAGCAGCCAGTTCCGGGG
GAACGAGACGGGGCTGATCACGCCGAACAAGCACGGGCTGCTGCAGAACCCGTACCGGATCCCGCCGCTGTTCCCGATCA
```

AGAGCTTCGTGAAGACGAAGTGCAAGAAGAACCTGCTGGAGGAGAACTTCGAGGAGCACAGCATGAGCCCGGAGCGGGAG
ATGGGGAACGAGAACATCCCGAGCACGGTGAGCACGATCAGCCGGAACAACATCCGGGAGAACGTGTTCAAGGAGGCGAG
CAGCAGCAACATCAACGAGGTGGGGAGCAGCACGAACGAGGTGGGGAGCAGCATCAACGAGATCGGGAGCAGCGACGAGA
ACATCCAGGCGGAGCTGGGGCGGAACCGGGGGCCGAAGCTGAACGCGATGCTGCGGCTGGGGGTGCTGCAGCCGGAGGTG
TACAAGCAGAGCCTGCCGGGGAGCAACTGCAAGCACCCGGAGATCAAGAAGCAGGAGTACGAGGAGGTGGTGCAGACGGT
GAACACGGACTTCAGCCCGTACCTGATCAGCGACAACCTGGAGCAGCCGATGGGGAGCAGCCACGCGAGCCAGGTGTGCA
GCGAGACGCCGGACGACCTGCTGGACGACGGGGAGATCAAGGAGGACACGAGCTTCGCGGAGAACGACATCAAGGAGAGC
AGCGCGGTGTTCAGCAAGAGCGTGCAGAAGGGGGAGCTGAGCCGGAGCCCGAGCCCGTTCACGCACACGCACCTGGCGCA
GGGGTACCGGCGGGGGGCGAAGAAGCTGGAGAGCAGCGAGGAGAACCTGAGCAGCGAGGACGAGGAGCTGCCGTGCTTCC
AGCACCTGCTGTTCGGGAAGGTGAACAACATCCCGAGCCAGAGCACGCGGCACAGCACGGTGGCGACGGAGTGCCTGAGC
AAGAACACGGAGGAGAACCTGCTGAGCCTGAAGAACAGCCTGAACGACTGCAGCAACCAGGTGATCCTGGCGAAGGCGAG
CCAGGAGCACCACCTGAGCGAGGAGACGAAGTGCAGCGCGAGCCTGTTCAGCAGCCAGTGCAGCGAGCTGGAGGACCTGA
CGGCGAACACGAACACGCAGGACCCGTTCCTGATCGGGAGCAGCAAGCAGATGCGGCACCAGAGCGAGAGCCAGGGGGTG
GGGCTGAGCGACAAGGAGCTGGTGAGCGACGACGAGGAGCGGGGGACGGGGCTGGAGGAGAACAACCAGGAGGAGCAGAG
CATGGACAGCAACCTGGGGGAGGCGGCGAGCGGGTGCGAGAGCGAGACGAGCGTGAGCGAGGACTGCAGCGGGCTGAGCA
GCCAGAGCGACATCCTGACGACGCAGCAGCGGGACACGATGCAGCACAACCTGATCAAGCTGCAGCAGGAGATGGCGGAG
CTGGAGGCGGTGCTGGAGCAGCACGGGAGCCAGCCGAGCAACAGCTACCCGAGCATCATCAGCGACAGCAGCGCGCTGGA
GGACCTGCGGAACCCGGAGCAGAGCACGAGCGAGAAGGCGGTGCTGACGAGCCAGAAGAGCAGCGAGTACCCGATCAGCC
AGAACCCGGAGGGGCTGAGCGCGGACAAGTTCGAGGTGAGCGCGGACAGCAGCACGAGCAAGAACAAGGAGCCGGGGGTG
GAGCGGAGCAGCCCGAGCAAGTGCCCGAGCCTGGACGACCGGTGGTACATGCACAGCTGCAGCGGGAGCCTGCAGAACCG
GAACTACCCGAGCCAGGAGGAGCTGATCAAGGTGGTGGACGTGGAGGAGCAGCAGCTGGAGGAGAGCGGGCCGCACGACC
TGACGGAGACGAGCTACCTGCCGCGGCAGGACCTGGAGGGGACGCCGTACCTGGAGAGCGGGATCAGCCTGTTCAGCGAC
GACCCGGAGAGCGACCCGAGCGAGGACCGGGCGCCGGAGAGCGCGCGGGTGGGGAACATCCCGAGCAGCACGAGCGCGCT
GAAGGTGCCGCAGCTGAAGGTGGCGGAGAGCGCGCAGAGCCCGGCGGCGGCGCACACGACGGACACGGCGGGGTACAACG
CGATGGAGGAGAGCGTGAGCCGGGAGAAGCCGGAGCTGACGGCGAGCACGGAGCGGGTGAACAAGCGGATGAGCATGGTG
GTGAGCGGGCTGACGCCGGAGGAGTTCATGCTGGTGTACAAGTTCGCGCGGAAGCACCACATCACGCTGACGAACCTGAT
CACGGAGGAGACGACGCACGTGGTGATGAAGACGGACGCGGAGTTCGTGTGCGAGCGGACGCTGAAGTACTTCCTGGGGA
TCGCGGGGGGGAAGTGGGTGGTGAGCTACTTCTGGGTGACGCAGAGCATCAAGGAGCGGAAGATGCTGAACGAGCACGAC
TTCGAGGTGCGGGGGGACGTGGTGAACGGGCGGAACCACCAGGGGCCGAAGCGGGCGCGGGAGAGCCAGGACCGGAAGAT
CTTCCGGGGGCTGGAGATCTGCTGCTACGGGCCGTTCACGAACATGCCGACGGACCAGCTGGAGTGGATGGTGCAGCTGT
GCGGGGCGAGCGTGGTGAAGGAGCTGAGCAGCTTCACGCTGGGGACGGGGGTGCACCCGATCGTGGTGGTGCAGCCGGAC
GCGTGGACGGAGGACAACGGGTTCCACGCGATCGGGCAGATGTGCGAGGCGCCGGTGGTGACGCGGGAGTGGGTGCTGGA
CAGCGTGGCGCTGTACCAGTGCCAGGAGCTGGACACGTACCTGATCCCGCAGATCCCGCACAGCCACTACTGA

Secuencia maximizada:

ATGGACCTGAGCGCGCTGCGGGTGGAGGAGGTGCAGAACGTGATCAACGCGATGCAGAAGATCCTGGAGTGCCCGATCT
GCCTGGAGCTGATCAAGGAGCCGGTGAGCACGAAGTGCGACCACATCTTCTGCAAGTTCTGCATGCTGAAGCTGCTGAAC
CAGAAGAAGGGGCCGAGCCAGTGCCCGCTGTGCAAGAACGACATCACGAAGCGGAGCCTGCAGGAGAGCACGCGGTTCAG
CCAGCTGGTGGAGGAGCTGCTGAAGATCATCTGCGCGTTCCAGCTGGACACGGGGCTGGAGTACGCGAACAGCTACAACT
TCGCGAAGAAGGAGAACAACAGCCCGGAGCACCTGAAGGACGAGGTGAGCATCATCCAGAGCATGGGGTACCGGAACCGG
GCGAAGCGGCTGCTGCAGAGCGAGCCGGAGAACCCGAGCCTGCAGGAGACGAGCCTGAGCGTGCAGCTGAGCAACCTGGG
GACGGTGCGGACGCTGCGGACGAAGCAGCGGATCCAGCCGCAGAAGACGAGCGTGTACATCGAGCTGGGGAGCGACAGCA
GCGAGGACACGGTGAACAAGGCGACGTACTGCAGCGTGGGGGACCAGGAGCTGCTGCAGATCACGCCGCAGGGGACGCGG
GACGAGATCAGCCTGGACAGCGCGAAGAAGGCGGCGTGCGAGTTCAGCGAGACGGACGTGACGAACACGGAGCACCACCA
GCCGAGCAACAACGACCTGAACACGACGGAGAAGCGGGCGGCGGAGCGGCACCCGGAGAAGTACCAGGGGAGCAGCGTGA
GCAACCTGCACGTGGAGCCGTGCGGGACGAACACGCACGCGAGCAGCCTGCAGCACGAGAACAGCAGCCTGCTGCTGACG
AAGGACCGGATGAACGTGGAGAAGGCGGAGTTCTGCAACAAGAGCAAGCAGCCGGGGCTGGCGCGGAGCCAGCACAACCG
GTGGGCGGGGAGCAAGGAGACGTGCAACGACCGGCGGACGCCGAGCACGGAGAAGAAGGTGGACCTGAACGCGGACCCGC

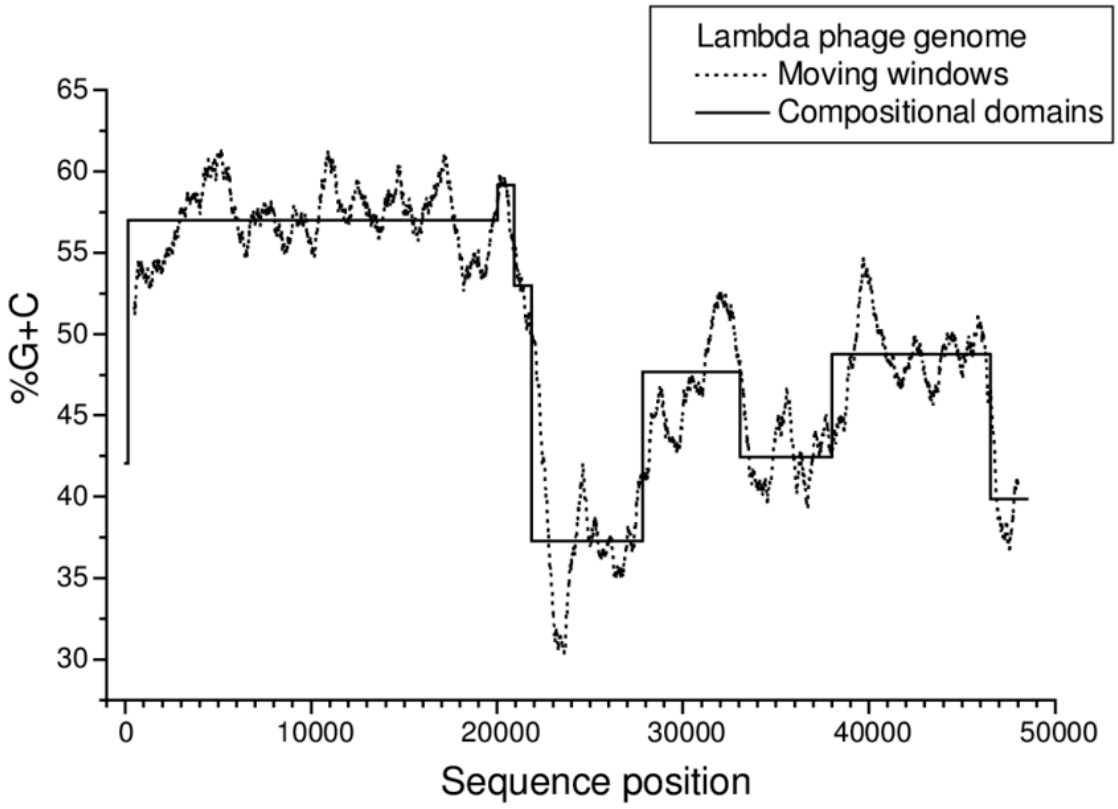
TGTGCGAGCGGAAGGAGTGGAACAAGCAGAAGCTGCCGTGCAGCGAGAACCCGCGGGACACGGAGGACGTGCCGTGGATC
ACGCTGAACAGCAGCATCCAGAAGGTGAACGAGTGGTTCAGCCGGAGCGACGAGCTGCTGGGGAGCGACGACAGCCACGA
CGGGGAGAGCGAGAGCAACGCGAAGGTGGCGGACGTGCTGGACGTGCTGAACGAGGTGGACGAGTACAGCGGGAGCAGCG
AGAAGATCGACCTGCTGGCGAGCGACCCGCACGAGGCGCTGATCTGCAAGAGCGAGCGGGTGCACAGCAAGAGCGTGGAG
AGCAACATCGAGGACAAGATCTTCGGGAAGACGTACCGGAAGAAGGCGAGCCTGCCGAACCTGAGCCACGTGACGGAGAA
CCTGATCATCGGGGCGTTCGTGACGGAGCCGCAGATCATCCAGGAGCGGCCGCTGACGAACAAGCTGAAGCGGAAGCGGC
GGCCGACGAGCGGGCTGCACCCGGAGGACTTCATCAAGAAGGCGGACCTGGCGGTGCAGAAGACGCCGGAGATGATCAAC
CAGGGGACGAACCAGACGGAGCAGAACGGGCAGGTGATGAACATCACGAACAGCGGGCACGAGAACAAGACGAAGGGGGA
CAGCATCCAGAACGAGAAGAACCCGAACCCGATCGAGAGCCTGGAGAAGGAGAGCGCGTTCAAGACGAAGGCGGAGCCGA
TCAGCAGCAGCATCAGCAACATGGAGCTGGAGCTGAACATCCACAACAGCAAGGCGCCGAAGAAGAACCGGCTGCGGCGG
AAGAGCAGCACGCGGCACATCCACGCGCTGGAGCTGGTGGTGAGCCGGAACCTGAGCCCGCCGAACTGCACGGAGCTGCA
GATCGACAGCTGCAGCAGCAGCGAGGAGATCAAGAAGAAGAAGTACAACCAGATGCCGGTGCGGCACAGCCGGAACCTGC
AGCTGATGGAGGGGAAGGAGCCGGCGACGGGGGCGAAGAAGAGCAACAAGCCGAACGAGCAGACGAGCAAGCGGCACGAC
AGCGACACGTTCCCGGAGCTGAAGCTGACGAACGCGCCGGGGAGCTTCACGAAGTGCAGCAACACGAGCGAGCTGAAGGA
GTTCGTGAACCCGAGCCTGCCGCGGGAGGAGAAGGAGGAGAAGCTGGAGACGGTGAAGGTGAGCAACAACGCGGAGGACC
CGAAGGACCTGATGCTGAGCGGGGAGCGGGTGCTGCAGACGGAGCGGAGCGTGGAGAGCAGCAGCATCAGCCTGGTGCCG
GGGACGGACTACGGGACGCAGGAGAGCATCAGCCTGCTGGAGGTGAGCACGCTGGGGAAGGCGAAGACGGAGCCGAACAA
GTGCGTGAGCCAGTGCGCGGCGTTCGAGAACCCGAAGGGGCTGATCCACGGGTGCAGCAAGGACAACCGGAACGACACGG
AGGGGTTCAAGTACCCGCTGGGGCACGAGGTGAACCACAGCCGGGAGACGAGCATCGAGATGGAGGAGAGCGAGCTGGAC
GCGCAGTACCTGCAGAACACGTTCAAGGTGAGCAAGCGGCAGAGCTTCGCGCCGTTCAGCAACCCGGGGAACGCGGAGGA
GGAGTGCGCGACGTTCAGCGCGCACAGCGGGAGCCTGAAGAAGCAGAGCCCGAAGGTGACGTTCGAGTGCGAGCAGAAGG
AGGAGAACCAGGGGAAGAACGAGAGCAACATCAAGCCGGTGCAGACGGTGAACATCACGGCGGGGTTCCCGGTGGTGGGG
CAGAAGGACAAGCCGGTGGACAACGCGAAGTGCAGCATCAAGGGGGGGAGCCGGTTCTGCCTGAGCAGCCAGTTCCGGGG
GAACGAGACGGGGCTGATCACGCCGAACAAGCACGGGCTGCTGCAGAACCCGTACCGGATCCCGCCGCTGTTCCCGATCA
AGAGCTTCGTGAAGACGAAGTGCAAGAAGAACCTGCTGGAGGAGAACTTCGAGGAGCACAGCATGAGCCCGGAGCGGGAG
ATGGGGAACGAGAACATCCCGAGCACGGTGAGCACGATCAGCCGGAACAACATCCGGGAGAACGTGTTCAAGGAGGCGAG
CAGCAGCAACATCAACGAGGTGGGGAGCAGCACGAACGAGGTGGGGAGCAGCATCAACGAGATCGGGAGCAGCGACGAGA
ACATCCAGGCGGAGCTGGGGCGGAACCGGGGGCCGAAGCTGAACGCGATGCTGCGGCTGGGGGTGCTGCAGCCGGAGGTG
TACAAGCAGAGCCTGCCGGGGAGCAACTGCAAGCACCCGGAGATCAAGAAGCAGGAGTACGAGGAGGTGGTGCAGACGGT
GAACACGGACTTCAGCCCGTACCTGATCAGCGACAACCTGGAGCAGCCGATGGGGAGCAGCCACGCGAGCCAGGTGTGCA
GCGAGACGCCGGACGACCTGCTGGACGACGGGGAGATCAAGGAGGACACGAGCTTCGCGGAGAACGACATCAAGGAGAGC
AGCGCGGTGTTCAGCAAGAGCGTGCAGAAGGGGGAGCTGAGCCGGAGCCCGAGCCCGTTCACGCACACGCACCTGGCGCA
GGGGTACCGGCGGGGGGCGAAGAAGCTGGAGAGCAGCGAGGAGAACCTGAGCAGCGAGGACGAGGAGCTGCCGTGCTTCC
AGCACCTGCTGTTCGGGAAGGTGAACAACATCCCGAGCCAGAGCACGCGGCACAGCACGGTGGCGACGGAGTGCCTGAGC
AAGAACACGGAGGAGAACCTGCTGAGCCTGAAGAACAGCCTGAACGACTGCAGCAACCAGGTGATCCTGGCGAAGGCGAG
CCAGGAGCACCACCTGAGCGAGGAGACGAAGTGCAGCGCGAGCCTGTTCAGCAGCCAGTGCAGCGAGCTGGAGGACCTGA
CGGCGAACACGAACACGCAGGACCCGTTCCTGATCGGGAGCAGCAAGCAGATGCGGCACCAGAGCGAGAGCCAGGGGGTG
GGGCTGAGCGACAAGGAGCTGGTGAGCGACGACGAGGAGCGGGGGACGGGGCTGGAGGAGAACAACCAGGAGGAGCAGAG
CATGGACAGCAACCTGGGGGAGGCGGCGAGCGGGTGCGAGAGCGAGACGAGCGTGAGCGAGGACTGCAGCGGGCTGAGCA
GCCAGAGCGACATCCTGACGACGCAGCAGCGGGACACGATGCAGCACAACCTGATCAAGCTGCAGCAGGAGATGGCGGAG
CTGGAGGCGGTGCTGGAGCAGCACGGGAGCCAGCCGAGCAACAGCTACCCGAGCATCATCAGCGACAGCAGCGCGCTGGA
GGACCTGCGGAACCCGGAGCAGAGCACGAGCGAGAAGGCGGTGCTGACGAGCCAGAAGAGCAGCGAGTACCCGATCAGCC
AGAACCCGGAGGGGCTGAGCGCGGACAAGTTCGAGGTGAGCGCGGACAGCAGCACGAGCAAGAACAAGGAGCCGGGGGTG
GAGCGGAGCAGCCCGAGCAAGTGCCCGAGCCTGGACGACCGGTGGTACATGCACAGCTGCAGCGGGAGCCTGCAGAACCG
GAACTACCCGAGCCAGGAGGAGCTGATCAAGGTGGTGGACGTGGAGGAGCAGCAGCTGGAGGAGAGCGGGCCGCACGACC
TGACGGAGACGAGCTACCTGCCGCGGCAGGACCTGGAGGGGACGCCGTACCTGGAGAGCGGGATCAGCCTGTTCAGCGAC
GACCCGGAGAGCGACCCGAGCGAGGACCGGGCGCCGGAGAGCGCGCGGGTGGGGAACATCCCGAGCAGCACGAGCGCGCT
GAAGGTGCCGCAGCTGAAGGTGGCGGAGAGCGCGCAGAGCCCGGCGGCGGCGCACACGACGGACACGGCGGGGTACAACG

```
CGATGGAGGAGAGCGTGAGCCGGGAGAAGCCGGAGCTGACGGCGAGCACGGAGCGGGTGAACAAGCGGATGAGCATGGTG
GTGAGCGGGCTGACGCCGGAGGAGTTCATGCTGGTGTACAAGTTCGCGCGGAAGCACCACATCACGCTGACGAACCTGAT
CACGGAGGAGACGACGCACGTGGTGATGAAGACGGACGCGGAGTTCGTGTGCGAGCGGACGCTGAAGTACTTCCTGGGGA
TCGCGGGGGGGAAGTGGGTGGTGAGCTACTTCTGGGTGACGCAGAGCATCAAGGAGCGGAAGATGCTGAACGAGCACGAC
TTCGAGGTGCGGGGGGACGTGGTGAACGGGCGGAACCACCAGGGGCCGAAGCGGGCGCGGGAGAGCCAGGACCGGAAGAT
CTTCCGGGGGCTGGAGATCTGCTGCTACGGGCCGTTCACGAACATGCCGACGGACCAGCTGGAGTGGATGGTGCAGCTGT
GCGGGGCGAGCGTGGTGAAGGAGCTGAGCAGCTTCACGCTGGGGACGGGGGTGCACCCGATCGTGGTGGTGCAGCCGGAC
GCGTGGACGGAGGACAACGGGTTCCACGCGATCGGGCAGATGTGCGAGGCGCCGGTGGTGACGCGGGAGTGGGTGCTGGA
CAGCGTGGCGCTGTACCAGTGCCAGGAGCTGGACACGTACCTGATCCCGCAGATCCCGCACAGCCACTACTGA
```

[46]: 1

4.0.4 d)

En base a una investigación realizada para comprender la distribución que se da dentro de una secuencia para los porcentajes de GC solo se pudo hallar una que menciona una distribución clusterizada a lo largo de la secuencia, en este caso a partir de la Siguiente publicación Realizada por Oliver et. al. (2000). El cual plantea una distribución en un estilo de clusters los cuales con largas partes que pueden tener CG y otras secciones que no. Se adjunta la imagen del gráfico presentado para ello:



Con esto, se nos ocurre la siguiente metodología para trabajar: - 1. Calcular el tamaño de la sequencia codificadora, excluyendo el codón de inicio y stop. - 2. Obtener el total de nuecleótidos 'G' y 'C' y checkear el porcentaje de la sequencia al que corresponden. - 3. Tomar al azar codones

de la lista, teniendo una mayor probabilidad de seleccionar aquellos que tengan 'G' y 'C'. - 4. Una vez elegido el codón. Maximizarlo, si queda igual (en caso de ser 'GC'), aplicar este ítem de nuevo para alguno de los codones vecinos (si se repite, continuar al mismo sentido con el resto de vecinos que se decidió para evitar bucles). - 5. Repetir 3 y 4 de manera iterativa, y por cada iteración verificar el porcentaje hasta llegar al deseado para terminar el proceso.

De esta manera se busca crear una distribución semejante a la investigada. De forma que se construyan clústers como el del gráfico, el cual beneficie la creación de GC en un sector que ya los tenga.

---

5 Pregunta 4

5.0.1 a)

Se implementa a partir de la función original, un despiece para crear dos funciones aparte que serán llamadas para hacer lo solicitado, siendo en este caso top3 y last3 para entregar los 3 ORFs con porcentaje mas alto y los 3 con porcentaje más bajo.

[32]:
```
from math import inf

# Porcentaje más alto
def top3(proteins, seq):
  first = ()
  second = ()
  third = ()
  f_score = 0
  s_score = 0
  t_score =0

  for start, end in proteins:
    c = seq[start:end]
    s = c.count('C')+c.count('G')
    if s > f_score:
      first = (start, end)
      f_score = s
    elif s > s_score:
      second = (start, end)
      s_score = s
    elif s > t_score:
      third = (start, end)
      t_score = s

  return first, second, third

# Porcentaje mas bajo
def last3(proteins, seq):
  last3 = ()
```

```
  last2 = ()
  last = ()
  score_3 = inf
  score_2 = inf
  score_1 = inf

  for start, end in proteins:
    c = seq[start:end]
    s = c.count('C')+c.count('G')
    if s < score_1:
      last = (start, end)
      score_1 = s
    elif s < score_2:
      last2 = (start, end)
      score_2 = s
    elif s < score_3:
      last3 = (start, end)
      score_3 = s

  return last3, last2, last
```

[33]:
```
record = SeqIO.read("content/NC_005816.fna", "fasta")
table = 11
min_pro_len = 100


def find_orfs_with_trans(seq, trans_table, min_protein_length):
    answer = []
    seq_len = len(seq)
    for strand, nuc in [(+1, seq), (-1, seq.reverse_complement())]:
        for frame in range(3):
            trans = nuc[frame:].translate(trans_table)
            trans_len = len(trans)
            aa_start = 0
            aa_end = 0
            while aa_start < trans_len:
                aa_end = trans.find("*", aa_start)
                if aa_end == -1:
                    aa_end = trans_len
                if aa_end - aa_start >= min_protein_length:
                    if strand == 1:
                        start = frame + aa_start * 3
                        end = min(seq_len, frame + aa_end * 3 + 3)
                    else:
                        start = seq_len - frame - aa_end * 3 - 3
                        end = seq_len - frame - aa_start * 3
                    answer.append((start, end, strand, trans[aa_start:aa_end]))
```

```
                aa_start = aa_end + 1
    answer.sort()
    return answer


orf_list = find_orfs_with_trans(record.seq, table, min_pro_len)
proteins = []
for start, end, strand, pro in orf_list:
  proteins.append((start,end))

  print(
      "%s...%s - length %i, strand %i, %i:%i"
      % (pro[:30], pro[-3:], len(pro), strand, start, end)
  )
```

```
NQIQGVICSPDSGEFMVTFETVMEIKILHK…GVA - length 355, strand 1, 41:1109
WDVKTVTGVLHHPFHLTFSLCPEGATQSGR…VKR - length 111, strand -1, 491:827
KSGELRQTPPASSTLHLRLILQRSGVMMEL…NPE - length 285, strand 1, 1030:1888
RALTGLSAPGIRSQTSCDRLRELRYVPVSL…PLQ - length 119, strand -1, 2830:3190
RRKEHVSKKRRPQKRPRRRRFFHRLRPPDE…PTR - length 128, strand 1, 3470:3857
GLNCSFFSICNWKFIDYINRLFQIIYLCKN…YYH - length 176, strand 1, 4249:4780
RGIFMSDTMVVNGSGGVPAFLFSGSTLSSY…LLK - length 361, strand -1, 4814:5900
VKKILYIKALFLCTVIKLRRFIFSVNNMKF…DLP - length 165, strand 1, 5923:6421
LSHTVTDFTDQMAQVGLCQCVNVFLDEVTG…KAA - length 107, strand -1, 5974:6298
GCLMKKSSIVATIITILSGSANAASSQLIP…YRF - length 315, strand 1, 6654:7602
IYSTSEHTGEQVMRTLDEVIASRSPESQTR…FHV - length 111, strand -1, 7788:8124
WGKLQVIGLSMWMVLFSQRFDDWLNEQEDA…ESK - length 125, strand -1, 8087:8465
TGKQNSCQMSAIWQLRQNTATKTRQNRARI…AIK - length 100, strand 1, 8741:9044
QGSGYAFPHASILSGIAMSHFYFLVLHAVK…CSD - length 114, strand -1, 9264:9609
```

[34]:
```
t3 = top3(proteins, record.seq)
l3 = last3(proteins, record.seq)

print("Top 3 Más alto:\n",t3)
print("...")
print("Top 3 más bajo:\n",l3)
```

```
Top 3 Más alto:
 ((41, 1109), (1030, 1888), (4814, 5900))
…
Top 3 más bajo:
 ((8087, 8465), (9264, 9609), (8741, 9044))
```

---

6 Pregunta 5

6.0.1 a) y b)

Se crea el programa a partir de la librería random de python para llevar a cabo la solicitud, utilizando Biopython para extraer las secuencias. Almacenando el resultado en un archivo fasta.

[48]:
```
from random import shuffle

record = SeqIO.read("content/plasmido.fna", "fasta")

recordL = list(record.seq)
shuffle(recordL)
rand_seq = ''.join(recordL)

rRec = SeqRecord(
    Seq(
        rand_seq,
    ),
)
SeqIO.write(rRec, "content/rand_plasmido.fna", "fasta")
```

[48]: 1

6.0.2 c)

Genoma orignial:

```
GeneMark.hmm PROKARYOTIC (Version 3.26)
Date: Thu Dec  8 19:45:57 2022
Sequence file name: seq.fna
Model file name: /home/genemark/bin/gmsuite/heu_11.mod
RBS: false

Model information: Heuristic_model_for_genetic_code_11_and_GC_54

FASTA definition line: gi|10955253|ref|NC_002119.1| Escherichia coli plasmid CloDF13, complete sequence
Predicted genes
   Gene    Strand    LeftEnd    RightEnd      Gene     Class
    #                                       Length
    1        -        1167        1424         258        1
    2        -        1434        2831        1398        1
    3        +        3446        3691         246        1
    4        +        3813        4379         567        1
    5        -        6054        6452         399        1
    6        -        6469        6747         279        1
    7        -        6754        6963         210        1
    8        -        7061        7510         450        1
    9        -        7649        8806        1158        1
   10        -        8886        9131         246        1
```

Genoma randomizado:

```
GeneMark.hmm PROKARYOTIC (Version 3.26)
Date: Thu Dec  8 19:46:33 2022
Sequence file name: seq.fna
Model file name: /home/genemark/bin/gmsuite/heu_11.mod
RBS: false

Model information: Heuristic_model_for_genetic_code_11_and_GC_54

FASTA definition line: <unknown id> <unknown description>
Predicted genes
   Gene    Strand    LeftEnd    RightEnd       Gene     Class
    #                                        Length
    1        +          <1         141          141        1
    2        -        4313        4600          288        1
```

Se evidencia inmediatamente que hay muchos más candidatos en el gen original del plásmido.

Una de las razones que podrian influir en esto es el hecho de que es necesario un codón de inicio (AUG), el cual tiene una probabilidad de 1/64 de ocurrir, y un codón de terminación, el cual tiene una probabilidad de 3/64 de ocurrir.