Algorithms for Bioinformatics

2018/2019

Multiple Sequence Alignment

Pedro G. Ferreira
[d**CC**] @ Faculty of Sciences University of Porto

Outline

- Generalization of the alignment problem to multiple sequences
- Progressive Alignments
- Implementation of simple version of MSA
- Exercises

Multiple Sequence Alignment

- In bioinformatics frequently we need to align multiple sequences, e.g.
  - identify recurrent motifs in a family of protein sequences (domains).
  - study homology relations between an evolving gene (Phylogenetic analysis).
  - assess conservation on secondary and tertiary protein structures.
  - epidemiological studies to understand the mutation rate of a gives species strain.

- The Multiple Sequence Alignment (MSA) generalizes the pairwise alignment problem for $N > 2$ sequences.

- The goal, as before, is to perform edit operations to align the characters in the different sequences to maximize the number of similar characters per columns.

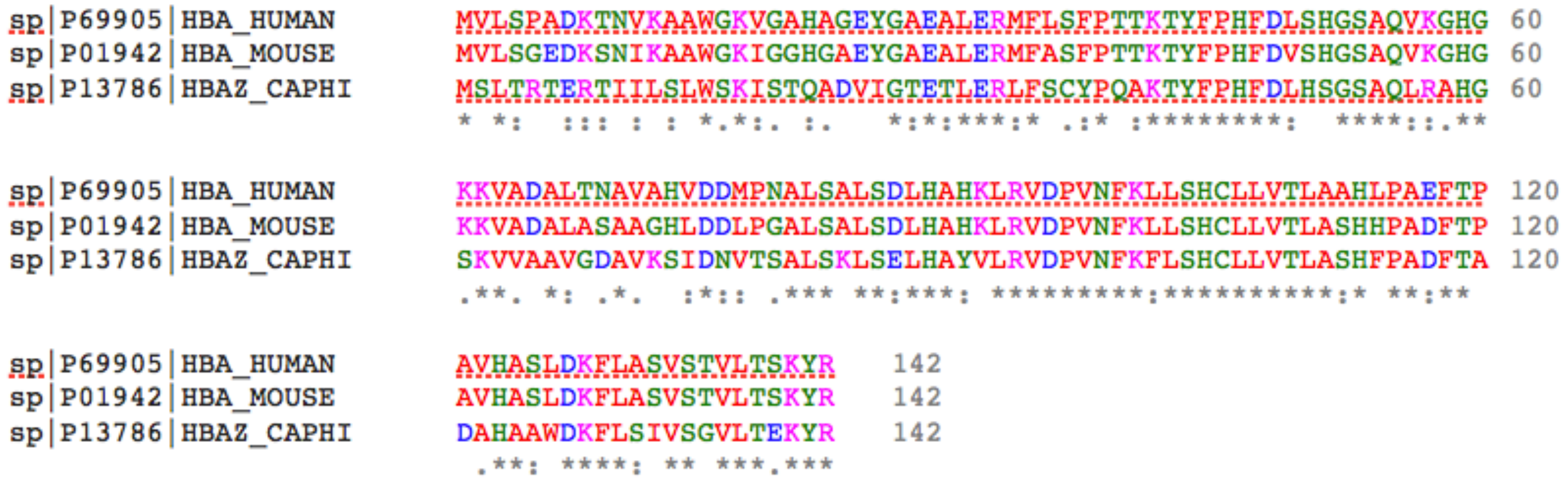
- MSAs can be either local or global.

Multiple Sequence Alignment

- Clustal Omega:

https://www.ebi.ac.uk/Tools/msa/clustalo/

Clustal Omega



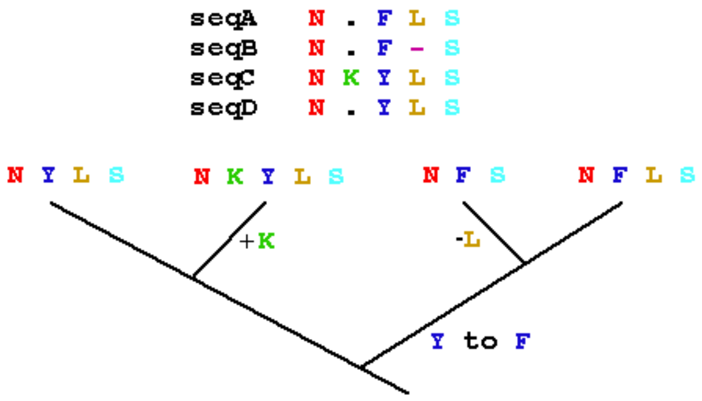
```
sp|P69905|HBA_HUMAN      MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHG 60
sp|P01942|HBA_MOUSE      MVLSGEDKSNIKAAWGKIGGHGAEYGAEALERMFASFPTTKTYFPHFDVSHGSAQVKGHG 60
sp|P13786|HBAZ_CAPHI     MSLTRTERTIILSLWSKISTQADVIGTETLERLFSCYPQAKTYFPHFDLHSGSAQLRAHG 60
                         * *:  ::: : : *.*:. :.   *:*:***:* .:* :********:  ****::.**

sp|P69905|HBA_HUMAN      KKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTP 120
sp|P01942|HBA_MOUSE      KKVADALASAAGHLDDLPGALSALSDLHAHKLRVDPVNFKLLSHCLLVTLASHHPADFTP 120
sp|P13786|HBAZ_CAPHI     SKVVAAVGDAVKSIDNVTSALSKLSELHAYVLRVDPVNFKFLSHCLLVTLASHFPADFTA 120
                         .**. *: .*.  :*:: .*** **:***: *********:**********:* **:**

sp|P69905|HBA_HUMAN      AVHASLDKFLASVSTVLTSKYR   142
sp|P01942|HBA_MOUSE      AVHASLDKFLASVSTVLTSKYR   142
sp|P13786|HBAZ_CAPHI     DAHAAWDKFLSIVSGVLTEKYR   142
                          .**: ****: ** ***.***
```

- File formats for MSA:

https://www.ebi.ac.uk/seqdb/confluence/display/JDSATMultiple+Sequence+Alignment+Tool+Input+Examples

Multiple Sequence Alignment

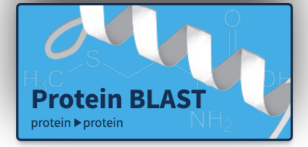
- MSA can be used to start phylogenetic analysis. The phylogenetic tree describes the distance between the sequences under analysis. From the alignment, each column shows if there are conservation of the residues (amino-acids), mutations or divergence from the common ancestor.



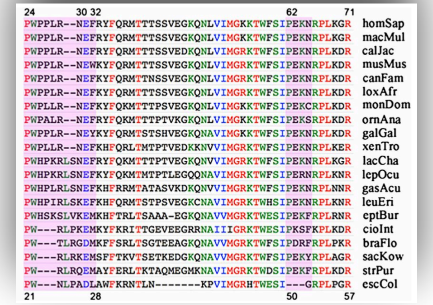
- Credit to Enrique Blanco; Taken from http://genome.crg.es/courses/msa/

PWPPLRSTVVNEFWGH …

Protein of unknown function

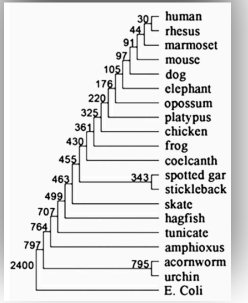


BLAST



```
24          30 32                          62       71
PWPPLR--NEFRYFQRMTTTSSVEGKQNLVIMGKKTWFSIPEKNRPLKGR  homSap
PWPPLR--NEFRYFQRMTTTSSVEGKQNLVIMGKKTWFSIPEKNRPLKGR  macMul
PWPPLR--NEFRYFQRMTTTSSVEDKQNLVIMGRKTWFSIPEKNRPLKDR  calJac
PWPPLR--NEWKYFQRMTTTSSVEGKQNLVIMGRKTWFSIPEKNRPLKDR  musMus
PWPPLR--NEFKYFQRMTTNSSVEGKQNLVIMGRKTWFSIPEKNRPLKDR  canFam
PWPPLR--NEFKYFQRMTTTSSVEGKQNLVIMGRKTWFSIPEKNRPLKDR  loxAfr
PWPLLR--NEFKYFQKMTTTPSVEGKQNLVIMGKKTWFSIPEKCRPLKDR  monDom
PWPALR--NEFRYFQKMTTTPTVKGKQNLVIMGKKTWFSIPEKSRPLKDR  ornAna
PWPPLR--NEYKYFQRMTSTSHVEGKQNAVIMGKKTWFSIPEKNRPLKDR  galGal
PWPLLR--NEFKHFQRLTMTPTVEDKKNVVIMGRKTWFSIPEKNRPLKER  xenTro
PWHPKRLSNEFRYFQKMTTTPTVEGKQNVVIMGRKTWFSIPEKNRPLKGR  lacCha
PWHPKRLSNEFKYFQKMTMTPTLEGQQNAVIMGRKTWFSIPERNRPLKNR  lepOcu
PWHPLRLSNEFKHFRRMTATASVKDKQNVVIMGRKTWFSIPEKNRPLNNR  gasAcu
PWHPIRLSKEFKHFQRMTSTPSVEGKCNAVIMGRKTWHSIPEKNRPLKNR  leuEri
PWHSKSLVKEMKHFTRLTSAAA-EGKQNAVVMGRKTWESIPEKFRPLRNR  eptBur
PW---RLPKEMKYFKRITTGEVEEGRRNAIIIGRKTWESIPKSFKPLKDR  cioInt
PW---TLRGDMKFFSRLTSGTEEAGKQNAVVMGRKTWFSIPDRFRPLPKR  braFlo
PW---RLRKEMSFFTKVTSETKEDGKQNAVVMGRKTWFSIPEKYRPLAGR  sacKow
PW---RLRQEMAYFERLTKTAQMEGMKNAVIMGRKTWDSIPEKFRPLKDR  strPur
PW---NLPADLAWFKRNTLN-------KPVIMGRHTWESI---GRPLPGR  escCol
21        28                            50      57
```

Multiple Sequence Alignment



Phylogenetic tree

- Functional significance of evolving protein sequence in dihydrofolate reductase from bacteria to human
- Liu et al PNAS June 18, 2013 110 (25) 10159-10164;

Multiple Sequence Alignment

- MSA can be represented as: Consensus sequence; Regular Expressions; Suffix trees; Blocks (ungapped regions); Position-specific scoring matrices; Profile Hidden Markov models

a

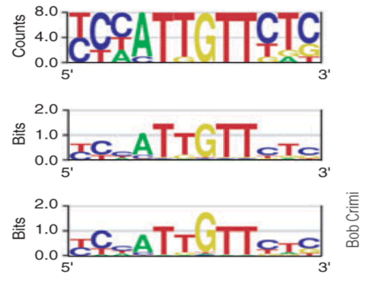
HEM13	CCCATTGTTCTC
HEM13	TTTCTGGTTCTC
HEM13	TCAATTGTTTAG
ANB1	CTCATTGTTGTC
ANB1	TCCATTGTTCTC
ANB1	CCTATTGTTCTC
ANB1	TCCATTGTTCGT
ROX1	CCAATTGTTTTG

b YCHATTGTTCTC

c

A	002700000010
C	464100000505
G	000001800112
T	422087088261

d **e** **f**



(a) Eight known genomic binding sites in three S. cerevisiae genes.

- (b) Degenerate consensus sequence.
- (c, d) Frequencies of nucleotides at each position.
- (e) Sequence logo showing the frequencies scaled relative to the information content (measure of conservation) at each position.
- (f) Energy normalized logo using relative entropy to adjust for low GC content in S. cerevisiae.

- "What are DNA sequence motifs?", Patrik D'haeseleer
- Nature Biotechnology 24, 423 - 425 (2006)

Multiple Sequence Alignment

- The complexity of MSA problem increases as the number of sequences to align increase. When N is large there are no very efficient algorithms to solve this problem.
- The MSA objective function will be similar to that of PSA. We will have a score with additive nature. The score of a column will depend on the number of introduced gaps and the scores from the substitution matrix.
- We now have multiple symbols per column. We need a way to compare between. The *sum of pairs* (SP) method considers all possible pairs of symbols within the column and sums them. Eventually, it counts the number of gaps only once.

```
-RADNS
ARCD-A
AR-D-A
```

```
Score = (sm(A;A)+g) + (3 x sm(R;R)) + (sm(A;C)+g) + (3 x sm(D;D)) + (2 x g) + (2 x sm(S; A)) + (sm(A;A))
```

Note that each gap is only considered once per column.

Multiple Sequence Alignment

- DP can be adapted for MSA. But DP has quadratic complexity and the complexity for MSA is exponential in the order of N. For a small number of sequences we can use this approach.

- Need more efficient approaches for larger values of N. Heuristic approaches provide approximate solutions when the classic approach does not work. It provides a trade-off between speed and optimality.

- Heuristic approaches:

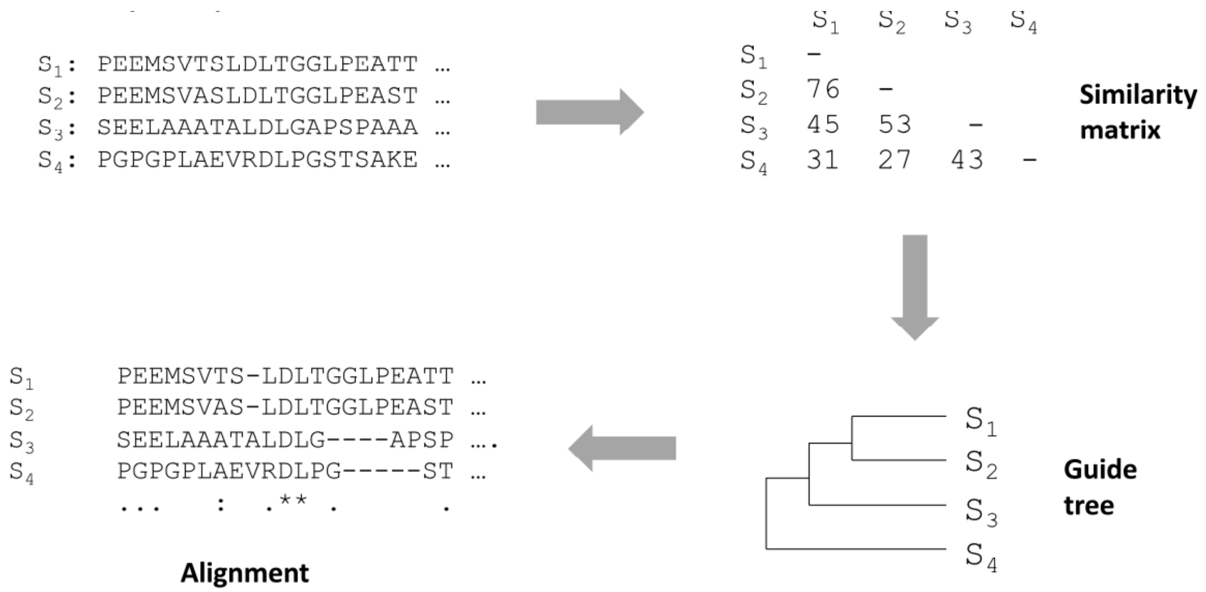
    - **Progressive**: start by aligning two sequences and iteratively add the other sequences to the alignment.
    - **Iterative**: consider an initial alignment and then improve it by adding, removing or moving gaps.
    - **Hybrid**: combine both strategies and use complementary information (e.g. protein structural information, other good alignments).

Multiple Sequence Alignment

- Progressive alignment idea: i) create an initial alignment with the two most similar/related sequences; ii) add in subsequent iterations, increasingly more distant sequences until all sequences are added to the alignment.

- The CLUSTAL algorithm is a classic MSA method that is at the basis of many MSA algorithms. Current version of the algorithm is Clustal Omega.

CLUSTAL OVERVIEW:

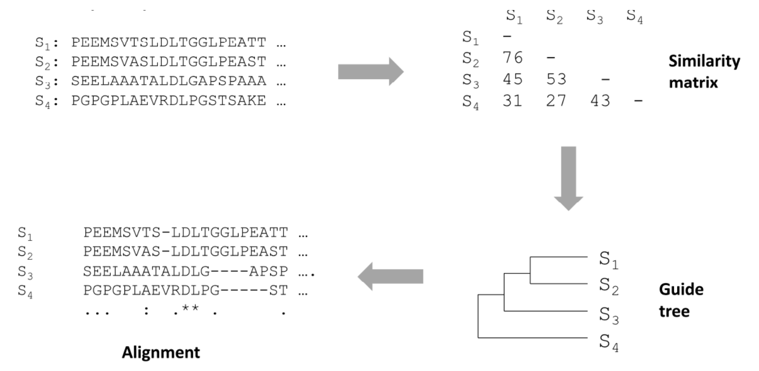

S1: PEEMSVTSLDLTGGLPEATT …
S2: PEEMSVASLDLTGGLPEAST …
S3: SEELAAATALDLGAPSPAAA …
S4: PGPGPLAEVRDLPGSTSAKE …
S1 S2 S3 S4
S1 -
S2 76 -
S3 45 53 -
S4 31 27 43 -
Similarity matrix
Guide tree
S1 S2 S3 S4
S1 PEEMSVTS-LDLTGGLPEATT …
S2 PEEMSVAS-LDLTGGLPEAST …
S3 SEELAAATALDLG----APSP ….
S4 PGPGPLAEVRDLPG-----ST …
. . .   :   .** .   .
Alignment

- From Bioinformatics Algorithms, Rocha & Ferreira

Multiple Sequence Alignment

Clustal Approach:

1. Calculate pairwise alignment of all sequences and build a similarity matrix;
2. Select the most similar sequences to form the basis of the MSA; the order of the sequences follows that of the guide tree;
3. To add more than two sequences we need to create, from the existing alignment, a summary of the content on each column.
   - ***Consensus representation***: each column is represented by the most common character;
   - ***Frequency representation***: each column is represented by the frequency of the characters, also called profile.
4. At each iteration align the new sequence with the profile of the current MSA.

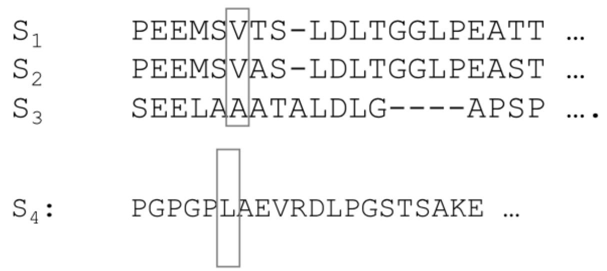

S1: PEEMSVTSLDLTGGLPEATT …
S2: PEEMSVASLDLTGGLPEAST …
S3: SEELAAATALDLGAPSPAAA …
S4: PGPGPLAEVRDLPGSTSAKE …
S1 S2 S3 S4
S1 -
S2 76 -
S3 45 53 -
S4 31 27 43 -
Similarity matrix
S1
S2
S3
S4
Guide tree
S1 PEEMSVTS-LDLTGGLPEATT …
S2 PEEMSVAS-LDLTGGLPEAST …
S3 SEELAAATALDLG----APSP ….
S4 PGPGPLAEVRDLPG-----ST …
... : .** . .
Alignment

○ From Bioinformatics Algorithms, Rocha & Ferreira

Multiple Sequence Alignment

- To compute the alignment of the new sequence with the profile of the current MSA an algorithm similar to DP is applied. It implicates a new way to compute the score of a column.

- The column score is the weighted average of the scores of all possible pairs:



```
S1      PEEMSVTS-LDLTGGLPEATT ...
S2      PEEMSVAS-LDLTGGLPEAST ...
S3      SEELAAATALDLG----APSP ....

S4:     PGPGPLAEVRDLPGSTSAKE ...
```

Score =
(SM(V,L)* 2 + SM(A,L)) / 3

- From Bioinformatics Algorithms, Rocha & Ferreira

- Once a gap is added to the alignment it will prevail as gap throughout the next alignment steps.

- In some situations we may need to join to sub-alignments corresponding to two different sub-branches of the tree. This will require joining two profiles in a generalisation of the above process.

Multiple Sequence Alignment

- Progressive alignment algorithms may present some problems related to their heuristic greedy nature. If wrong decisions are made early in the alignment they will be propagated and not corrected afterwards.

- Worst results occur when sequences have low similarity.

- Iterative algorithms start by generating an alignment using one method (progressive alignment or any other), then make selected changes on the alignment and evaluate the impact of these changes on the objective function.

Multiple Sequence Alignment

- We will implement a simple version of a MSA using a progressive alignment strategy. It is be based on Object-Oriented programming approach using functions previously implemented. The following classes will be used and implemented:

  - ***MySeq***: implements functions over biological sequences.
  - ***SubstMatrix***: substitution matrix.
  - ***PairwiseAlignment***: functions for aligning two sequences.
  - ***MyAlign***: stores and represents alignments of two or more sequences.
  - **MultipleAlignment**: implements a simplified progressive MSA Algorithm.

MyAlign

- Defines alignments using two variables:
  - Alignment type (DNA, RNA , protein)
  - List of sequences included in the alignment (including “-“ to represent gaps).

- Methods:
  - Constructor: sets the list of sequences and their biotype.
  - __len__(self): number of columns in the alignment
  - __getitem__(self, ij): returns the symbol in sequence i and position j
  - __str__(self): prints the alignment
  - num_seqs(self): number of sequences in the alignment
  - column (self, index): returns the list of symbols in column index

```python
def column (self, index):
    ''' Returns list with symbols in column index '''
    res = []
    for k in range(len(self.listseqs)):
        res.append(self.listseqs[k][index])
    return res

```

MyAlign

- A consensus sequence is the sequence composed of the most frequent character in each column of the alignment **ignoring the gap**.

- Implement the function consensus. It should scan through all the columns of the alignment and count the number of occurrences of each symbol. Get the most frequent per column and concatenate to the consensus sequence. Hints:
  - Scan column by column; for each column count occurrences
  - Ignore gaps
  - Keep counts in hash table
  - Scan keys from hash table and check if values are greater than current maximum.

MyAlign

- Consensus function:

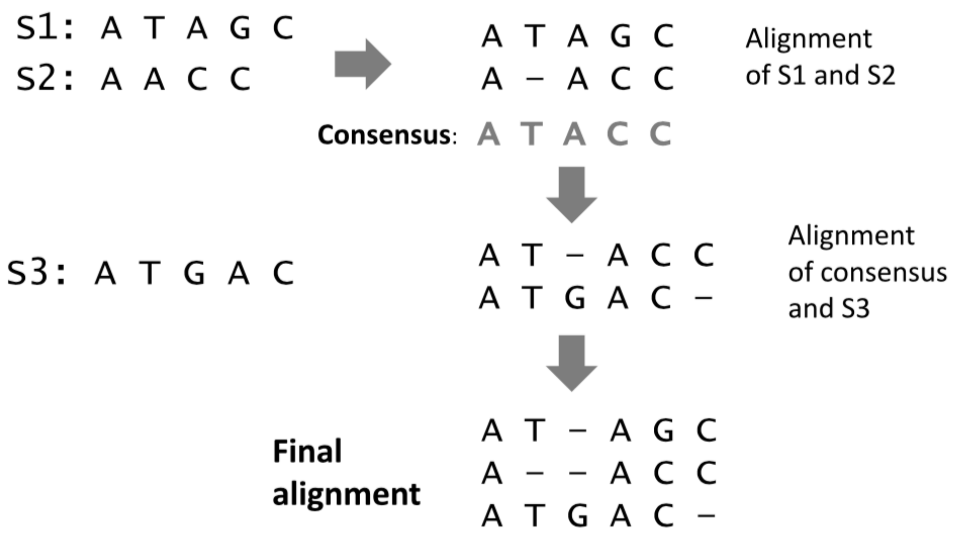
```python
def consensus (self):
    cons = ""
    for i in range(len(self)):
        cont = {}
        for k in range(len(self.listseqs)):
            c = self.listseqs[k][i]
            if c in cont:
                cont[c] = cont[c] + 1
            else:
                cont[c] = 1
        maximum = 0
        cmax = None
        for ke in cont.keys():
            if ke != "-" and cont[ke] > maximum:
                maximum = cont[ke]
                cmax = ke
        cons = cons + cmax
    return cons

```

MultipleAlign

- Implements the simplified progressive MSA. It receives as input:
  - Set of sequences in a given order.
  - Parameters of the objective function (substitution matrix and gap penalty).

  - Use the *Needleman-Wunsch* algorithm for pairwise alignment.
  - Built over the set of classes developed previously

Returns the best possible alignment of the sequences



Match = 1
Mismatch = -1
Gap penalty = -1

In case of ties in a column the consensus selects the first character in lexicographic order

- From Bioinformatics Algorithms, Rocha & Ferreira

MultipleAlign

- The function **add_seq_alignment** adds new sequences to existing alignments. It is called in each iteration of the main algorithm. It receives as input an existing alignment and a new sequence.

  - Align the consensus of previous alignment + new sequence = new alignment.

MultipleAlign

- The function **add_seq_alignment** adds new sequences to existing alignments. It is called in each iteration of the main algorithm:
  - Align the consensus of previous alignment + new sequence = new alignment.

```python
def add_seq_alignment (self, alignment, seq):
    res = []
    for i in range(len(alignment.listseqs)+1):
        res.append("")
    # create consensus from give alignments
    cons = MySeq(alignment.consensus(),alignment.al_type)
    self.alignpars.needleman_Wunsch(cons, seq)
    align2 = self.alignpars.recover_align()
    orig = 0
    for i in range(len(align2)):
        if align2[0,i]== '-':
            for k in range(len(alignment.listseqs)):
                res[k] += "-"
        else:
            for k in range(len(alignment.listseqs)):
                res[k] += alignment[k,orig]
            orig+=1
    res[len(alignment.listseqs)] = align2.listseqs[1]
    return MyAlign(res, alignment.al_type)

```

MultipleAlign

- The function **align_consensus** provides the general implementation of the MSA.
  - It should start by aligning the first two sequences; then recover the alignment.
  - Then, for the remaining sequences add the sequence to the alignment, i.e. to the resulting consensus sequence.
  - Return the resulting consensus sequence.

MultipleAlign

- The function **align_consensus** provides the general implementation of the MSA.
  - It should start by aligning the first two sequences; then recover the alignment.
  - Then, for the remaining sequences add the sequence to the alignment, i.e. to the resulting consensus sequence.
  - Return the resulting consensus sequence.

```python
def align_consensus(self):
    self.alignpars.needleman_Wunsch(self.seqs[0], self.seqs[1])
    res = self.alignpars.recover_align()

    for i in range(2, len(self.seqs)):
        res = self.add_seq_alignment(res, self.seqs[i])
    return res

```

Exercises

- Write a method ***ScoreColumn(self, charsCol)*** to add to the *MultipleAlign* class that allows to calculate the score of each column in the alignment (charsCol is a list of characters from the column of the alignment that can be retrieved using the method *column* from the class *MyAlign*). The score is calculated using the *Sum of Pairs (SP)* approach, i.e. the score will be the sum of the scores of each pair of characters in the alignment. If two gaps are found in each pair then the score will be zero.

- Using the previous methods, develop a new method called ***scoreSP(self, alignment)*** that returns the score of SP from a complete alignment.