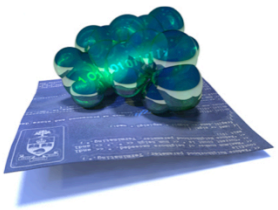
A
Bioinformatics
Course

# Optimal Sequence Alignment



Boris Steipe

*department of biochemistry – department of molecular genetics*
*University of Toronto*

proving homology

## How do we prove homology ?

If the alignment of two sequences scores so highly under a particular model of evolution from a common ancestor that a random chance similarity is sufficiently improbable, we may assume the sequences to be homologous.

## How do we measure compatibility with a model of evolution ?

Use a Scoring Matrix that quantifies relatedness *under a model of evolutionary relatedness*. Then score the correct alignment.

## What is the "correct" alignment ?

That is an alignment that pairs up those and only those residues that are descended by evolution from a common ancestor.

Motivating the need for optimal sequence alignments ...

### How do we generate the "correct" alignment ?

We can't. We can never guarantee that a particular alignment is correct! There is no possibility to *know* the ancestral sequence and the evolutionary sequence. Even the sequencing of ancient DNA does not guarantee we are looking at the actual progenitors.

### What can we do ?

**We can produce an optimal alignment. If the optimal alignment does not support homology, then the correct alignment will not support homology either.** But: we cannot guarantee that this is the correct alignment.

(In fact we can define scenarios in which it will not be, since a one-to-one relationship between residues may not be meaningful in distantly related sequences.)

inferring homology

In the absence of observation, the correct alignment remains unknown.
However: ...

If we produce the **best possible** alignment and we cannot infer homology from that, the "*correct*" alignment would not convince us either.

... and the *best possible alignment* **can** be constructed.

Note that this actually combines **two** objectives of optimal sequence alignments:
(i) use the score of the alignment o infer homology;
(ii) use the alignment itself to study contraints on structure and function.

to summarize the previous...

Sequence similarity can be measured as the sum of amino acid pairscores in an alignment.

Pairscores can be tabulated in a matrix.

The matrix defines what we mean by *similarity* when we apply it to amino acid pairs.

If the matrix represents our expectations about exchange likelihood in a model of evolution, the *similarity* measure it generates correlates with the likelihood that two sequences are homologous.

Once we can prove homology, we can infer shared properties between genes.

Once we can prove homology, we can infer shared properties between genes.

## How can we prove homology ?

If the alignment of two sequences is so indicative of a particular model of evolution that a random chance similarity is sufficiently improbable, we may assume homology of the sequences as the most plausible explanation.

## How can we measure compatibility with a particular model of evolution ?

Create the *correct alignment*. *Then* use a Scoring Matrix that quantifies similarity of all aligned pairs of amino acids *under a particular model of evolutionary relatedness*. Sum over all pair-scores.

## How can we measure compatibility with a particular model of evolution ?

Create the *correct alignment.* [...]

## What is the *correct* alignment ?

That is an alignment that pairs up those–and only those–residues that are the result of divergent evolution from a common ancestor. For this alignment the sum of pair-scores is a measure how compatible the similarities and differences between amino acids in the aligned pairs are with that model of how an ancestral sequence could have evolved into the present day sequences.

correct alignment

## How can we generate the *correct* alignment ?

We can't.

We can never guarantee that a particular alignment is correct! There is no possibility to *know* the ancestral sequence and the evolutionary trajectory. Even sequencing ancient DNA does not guarantee we are looking at the actual progenitors of observed present-day sequences.

## What can we do instead?

We can produce an *optimal alignment*. If the optimal alignment does not support homology, then the correct alignment will not support homology either. But: we cannot guarantee that this is the correct alignment.

(In fact we can define scenarios in which it can't be correct, since a one-to-one pair-relationship between residues may not be meaningful in distantly related sequences.)

correct alignment

In the absence of observation, the correct alignment remains unknown.
However: ...

If we produce the *best possible alignment* and we cannot infer homology from that, the *correct* alignment would not convince us either.

... and the *best possible alignment* can be constructed.

optimal alignment

## How can the best possible alignment be constructed ?

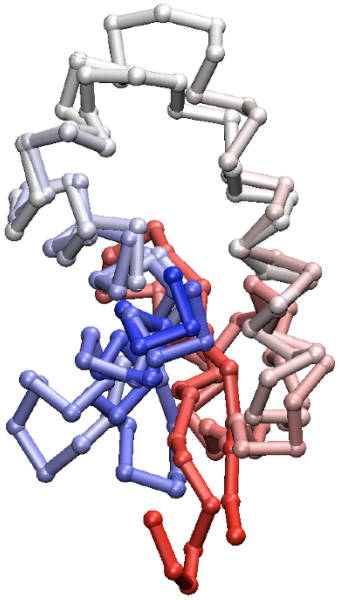
Can one generate all alignments, score them, and chose the best ?

Note that *best* in this context means: highest scoring.

... No. The existence of *indels* makes it intractable to consider all possible alignments.

indels

Related sequences often have different lengths. Ends can be lengthened and shortened, and internally, segments ranging from single residues to entire domains can have been inserted.

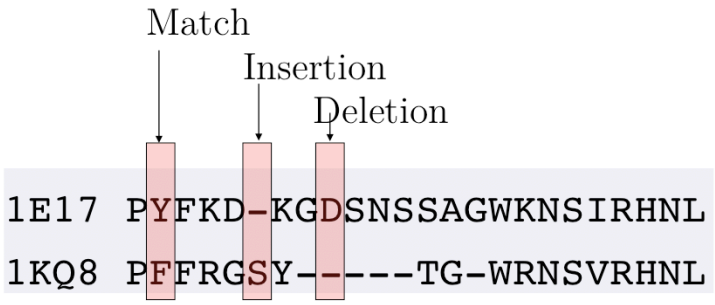


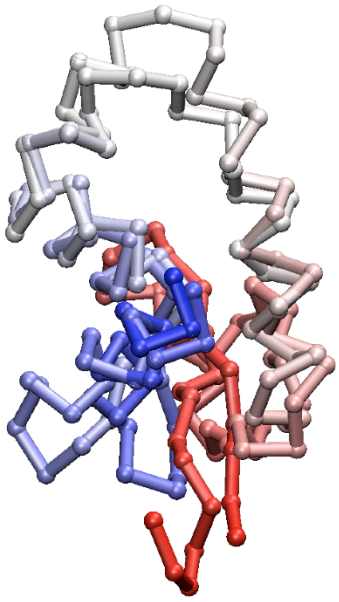
```
1E17 PYFKD-KGDSNSSAGWKNSIRHNL
1KQ8 PFFRGSY-----TG-WRNSVRHNL
```

In general, an insertion from the point of view of one sequence is the same as a deletion from the point of view of the other sequence, thus we often use the term "**indel**".

Note that the term insertion or deletion refers only to the sequences, not to the actual molecular event!

indels



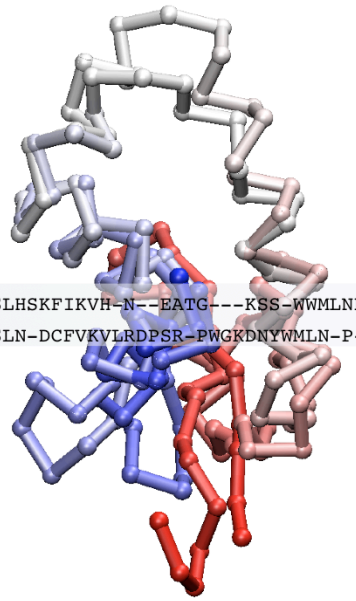
Since every position of the alignment can represent one of three states, the number of different alignments is on the order of $(3^{\text{length}})$–greater than the number of particles in the universe for the length of typical protein sequences.
This is an *intractable* problem.

Number of particles in the universe: on the order of $10^{81}$.

Alignments for two sequences of length 200: $\sim 3^{200} = 10^{95}$.

But: if we assume that the *global score is simply a sum of pair-scores*, we can devise an effective divide-and-conquer approach ...

1E17 RHNLSLHSKFIKVH-N--EATG---KSS-WWMLNPEGG
1KQ8 RHNLSLN-DCFVKVLRDPSR-PWGKDNYWMLN-P----



Note: a pair-score does not depend on the context of the aligned pair of amino acids, but only on the two amino acids themselves. Therefore it can be retrieved from a similarity matrix.

optimal alignment

**Premise:**

The total alignment score is the sum of all pair-scores for characters, minus a penalty for each indel.

**Pair scores depend only on the pair of characters under consideration.**

Since we determine a pair-score "locally" (without reference to its neighbourhood, or other context), simply by *looking it up in a scoring matrix*, we can subdivide the big problem of global alignment into many little problems that are easier to solve.

The premise of context independence makes finding an optimal alignment a solvable problem. It is can be shown that alignment problems that are not context-independent are NP hard, i.e. no algorithm exists that solves such a problem in a number of steps that is proportional to some polynomial of the alignment length. Rather, the number of steps in fully context-sensitive, gapped alignment must be proportional to some number to the power of the alignment length.

You can visualize this by considering that *context sensitive* really means: each local decision (whether to match two characters or insert an indel) is influenced by the state of all characters already in the alignment: all combinations of states are therefore distinct and must be considered separately. This is exactly the procedure which we have considered previously as the *brute-force* approacht o constructing alignments – and found to be intractable.

optimal alignment

The highest possible score of an alignment is the **(highest possible score of an alignment that is one residue shorter)**, extended in the best possible way by one residue ...

... the highest possible score of an alignment that is one residue shorter is the **(highest possible score of an alignment that is two residues shorter)**, extended in the best possible way by one residue ....

... the highest possible score of an alignment that is two residues shorter is the **(highest possible score of an alignment that is three residues shorter)**, extended in the best possible way by one residue ...

... the highest possible score of an alignment that is three residues shorter is the **(highest possible score of an alignment that is four residues shorter)**, extended in the best possible way by one residue ...

... the highest possible score of an alignment that is four residues shorter is the (highest possible score of an alignment that is five residues shorter), extended in the best possible way by one residue ...

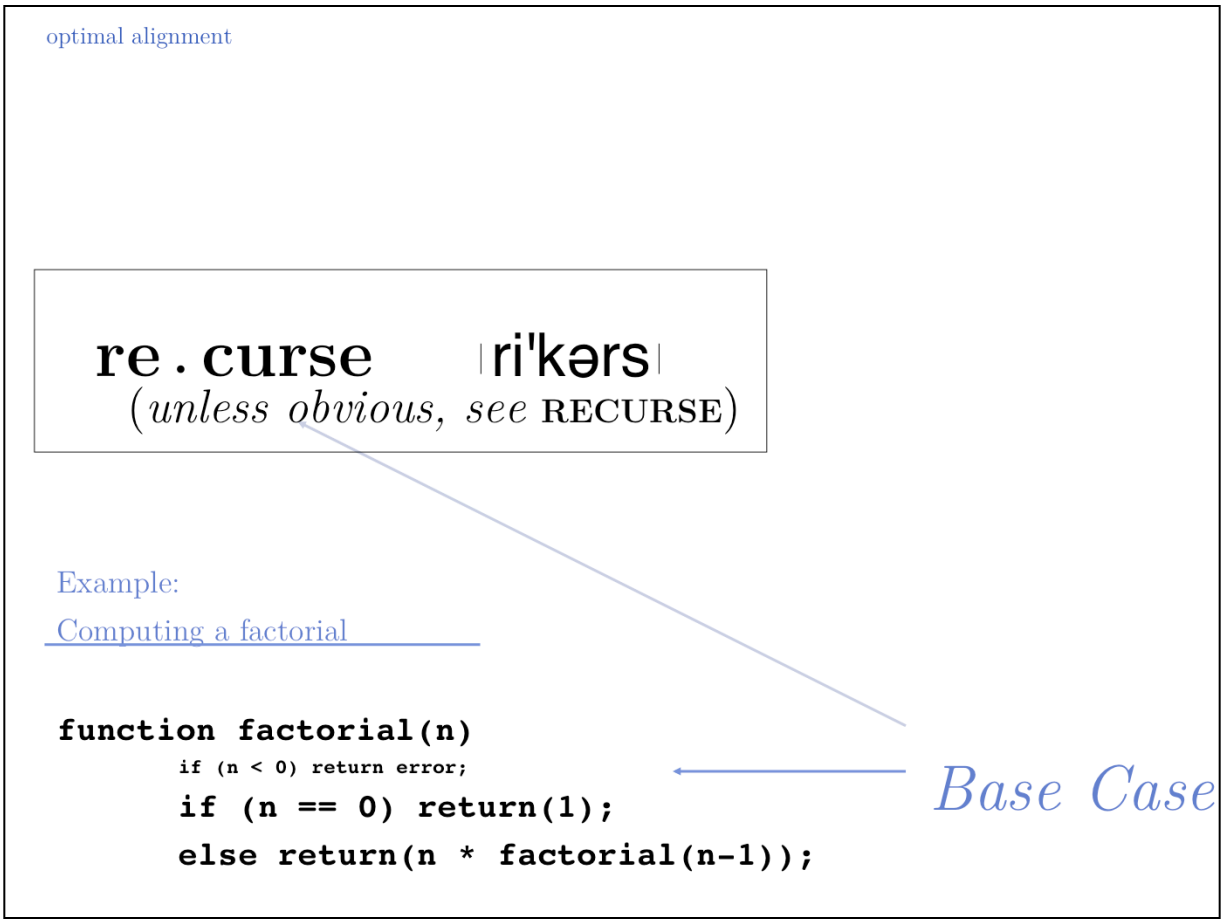
---

*... the highest possible score of an alignment that contains only a single pair of residues can be looked up in the scoring matrix.*

One can think of these types of algorithms as **recursive functions.**

**re . curse** |ri'kərs|
(*see* **RECURSE**)

This *ironic* (!) definition actually defines an infinite recursion - the rule is applied forever.




*Real* applications of recursive strategies or algorithms always require a so called **Base Case**: a situation where the recursion stops and a definite result is generated. More about this at Wikipedia: (http://en.wikipedia.org/wiki/Recursion_(computer_science)).

optimal alignment

## Recursive definition of alignment score in optimal alignment:

Global score for all positions up to and including $i, j$

Pairscore at position $i, j$

Best score prior to position $i, j$

$$S_{ij} = s_{ij} + \max \begin{cases} S_{i-1,\, j-1} & \text{or} \\ \max\limits_{2 \le x < i} S_{i-x,\, j-1} - w(x-1) & \text{or} \\ \max\limits_{2 \le y < j} S_{i-1,\, j-y} - w(y-1) & \end{cases}$$

Gap penalty function

With:

$i, j$ the $i^{\text{th}}$ resp $j^{\text{th}}$ position of the alignment

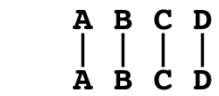
$x, y$ the length of an indel immediately preceding $i, j$

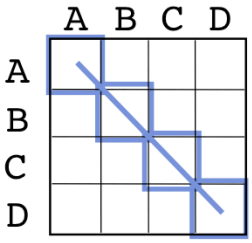
Optimal alignment, in the way we have defined the procedure a few slides ago, is simple to write as a recursion. However, implementing the approach as a recursion is very(!) inefficient since it requires looking up many values over and over again. For example if we are to calculate the score for i=9, j=10, we need to consider as one one of the possible extensions the cell i=8, j=9 and x=4 i.e. we need to calculate $s_{8\text{-}4,9\text{-}1}\text{-}w_{4\text{-}1} = s_{4,8}\text{-}w_3$. But this is the same value for s we previously had to calculate for the adjacent cell column: i=7, j=9, x=3: $s_{7\text{-}3,9\text{-}1}\text{-}w_{3\text{-}1} = s_{4,8}\text{-}w_2$, only with a different w. It is not the w-values that are costly to calculate however, but the s-values themselves, since we need to recurse all the way to the Base Case each time we want to calculate one. So while it is compact to write the alignment in the way given above, in practice we store each intermediate result that is going to be reused. This technique of storing useful intermediate results is called **Memoization** (not memo r ization) in computer science.

(cf. http://en.wikipedia.org/wiki/Memoization)

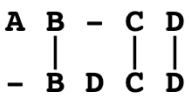
The actual algorithm therefore uses a compact and intuitive way to model the problem: store intermediate values in a matrix where rows and columns correspond to characters in the respective sequences. The highest score in the matrix is the optimal score and the cells that contribute to that score define the optimal alignment.

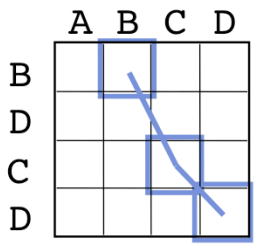
path matrix



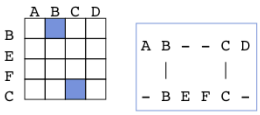




Any alignment can be represented as a path through a matrix that connects each intersection of row and column for two aligned characters.

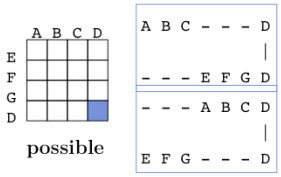


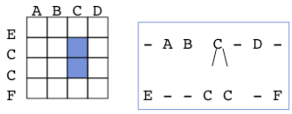

Stretches of *ungapped* aligned characters are *diagonally connected*. Indels **skip** over rows or columns. Paths that terminate away from the first or last cell represent end-gaps.



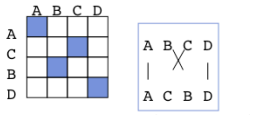

possible




possible




not allowed (duplication)




not allowed (inversion)

An alignment can be represented as a **path** through a **matrix** that has a row resp. column for every letter of the two sequences to be aligned. Any alignment can be represented as a path in such a matrix. Only a subset of arrangements correspond to legal paths that represent our normal definition of an alignment.

Note that – especially in genome/genome comparisons – duplications and inversions are common and specialized algorithms are available to perform such alignments (e.g. Shuffle-LAGAN (http://lagan.stanford.edu/ )).

algorithm

**Needleman & Wunsch (1970):**
**the optimal alignment is given by the path that leads to the highest possible sum of all the pair-scores it contains.**

First step: compile all pairwise scores into a matrix.

| | A | B | C | D |
|---|---|---|---|---|
| B | 0 | **1** | 0 | 0 |
| D | 0 | 0 | 0 | **1** |
| C | 0 | 0 | **1** | 0 |
| D | 0 | 0 | 0 | **1** |

(This example assumes: identity matrix - match=1, mismatch=0, no gap penalties)

The first step of the Needleman-Wunsch algorithm for global, optimal sequence alignment. This algorithmic strategy is frequently referred to as *Dynamic Programming*.

- http://en.wikipedia.org/wiki/Dynamic_programming
- http://en.wikipedia.org/wiki/Needleman-Wunsch_algorithm

algorithm

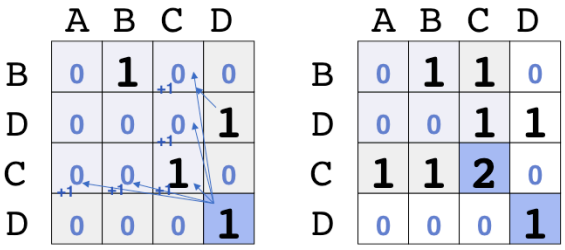
Second step: The highest score in the last column and row is the is highest pairscore we put there from the scoring matrix. This is the Base Case, if we think about the recursion, because there is no previous score we had to consider.

| | A | B | C | D |
|---|---|---|---|---|
| B | 0 | **1** | 0 | 0 |
| D | 0 | 0 | 0 | **1** |
| C | 0 | 0 | **1** | 0 |
| D | 0 | 0 | 0 | **1** |

(This example assumes: identity matrix - match=1, mismatch=0, no gap penalties)

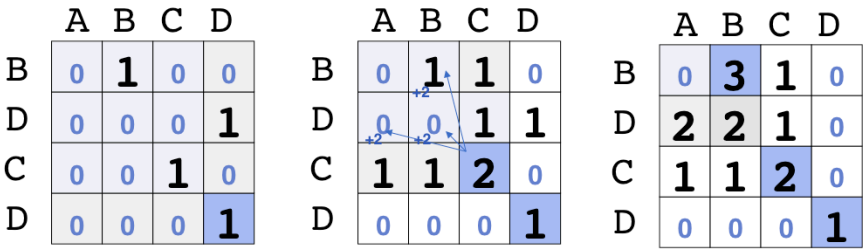
The next scores we need to calculate are the cells in the previous column or row...

Third step: Extend the path. Assign to each cell of the next column and row the highest value we can get by adding to its current value a value from a previous cell **that could be part of an alignment path**.




(This example assumes: identity matrix - match=1, mismatch=0, no gap penalties)

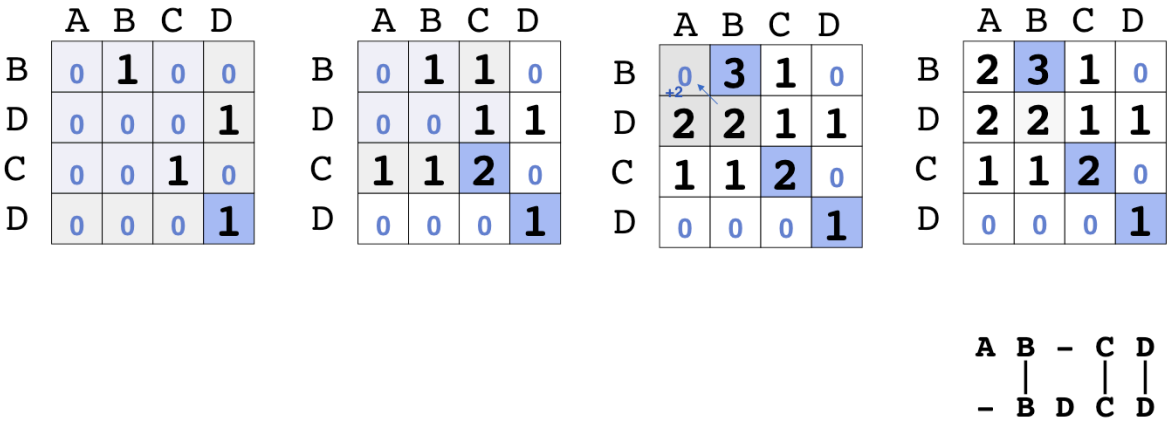
Repeat: (Assign to each cell of the next column and row the highest value we can get by adding to its current value a value from a previous cell **that could be part of an alignment path**.)




(This example assumes: identity matrix - match=1, mismatch=0, no gap penalties)

Final step: The **highest possible** score for the alignment path matrix is found after the matrix is filled.

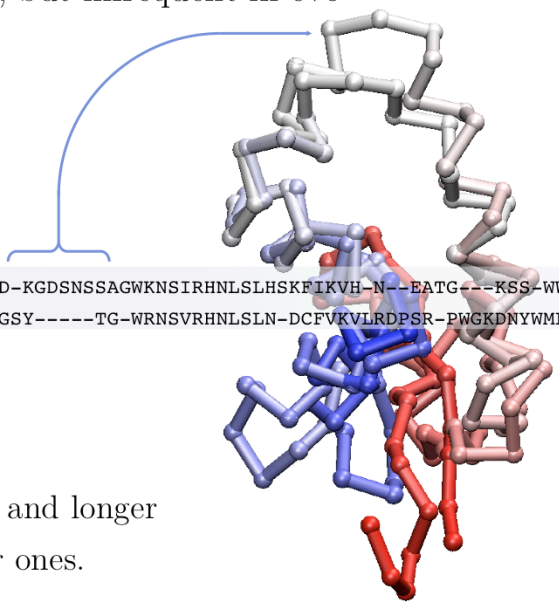
Once the highest possible score has been determined, we only need to find the **cells that have contributed to this score**. The optimal alignment is given by the path that contains these cells. The cells are simply retrieved by backtracking.




(This example assumes: identity matrix - match=1, mismatch=0, no gap penalties)

indels

In reality, related sequences *often* have different lengths. Ends can be lengthened and shortened, and segments ranging from single residues to entire domains can have been inserted or deleted. We need to take into account that indels are possible, but infrequent in evolution.



```
1E17 YAELISQAIESAPEK-RLTLAQIYEWMVRTVPYFKD-KGDSNSSAGWKNSIRHNLSLHSKFIKVH-N--EATG---KSS-WWMLNPEGG
1KQ8 YIALITMAIR-DSAGGRLTLAEINEYLMGKFPFFRGSY-----TG-WRNSVRHNLSLN-DCFVKVLRDPSR-PWGKDNYWMLN-P----
```

Empirically:
Insertion/deletion events are rare, and longer gaps are less frequent than shorter ones.

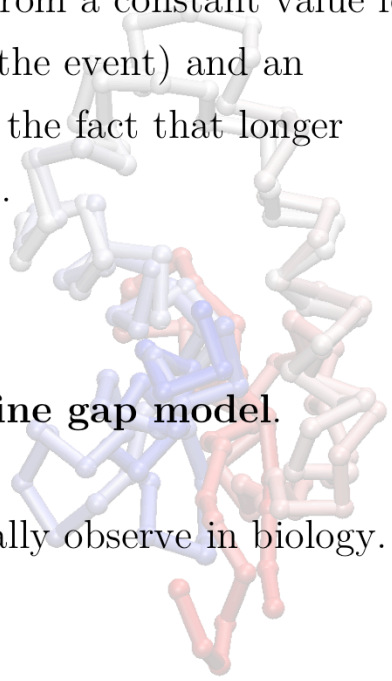
... unfortunately, we have no quantitative, mechanistic model for these events.

Commonly, a gap penalty is calculated from a constant value for opening the gap (to reflect the rarity of the event) and an increment for every extension (to reflect the fact that longer gaps are less frequent than shorter ones).

$$w(l) = a + bl$$

This type of gap penalty is called an **affine gap model**.

It does not reflect exactly what we actually observe in biology.



scoring indels

## Database analysis shows that gaps are log-distributed.

An attempt to model this situation has proposed a sum of exponentials ...

$$P(n) = \sum_i A_i e^{n\lambda_i}$$

... but other studies have **not** shown a clear advantage of logarithmic over affine gap penalties.

Qian & Goldstein (2001) *Proteins* B:102-104

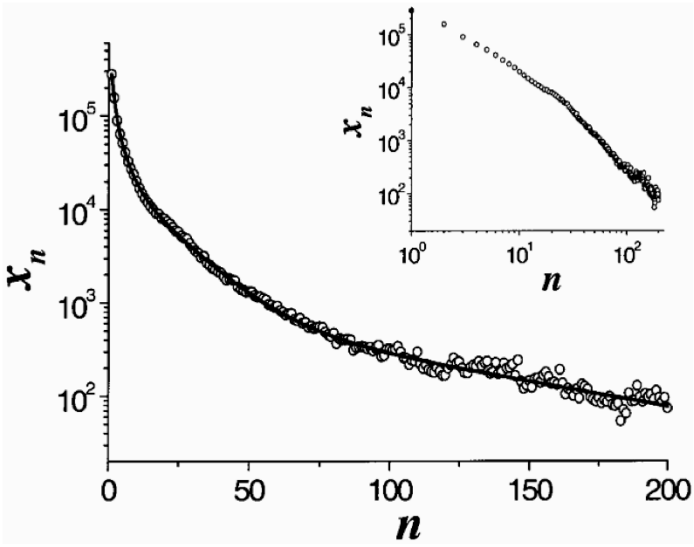



Fig. 1. Log plot and log-log plot of observed structure-based gap length distribution, compared with a quadruple-exponential fit (Eq. 8).

Note that discrete slopes in the log-log plot may indicate discrete molecular mechanisms for short- and long indels with characteristic length/frequency relationships.

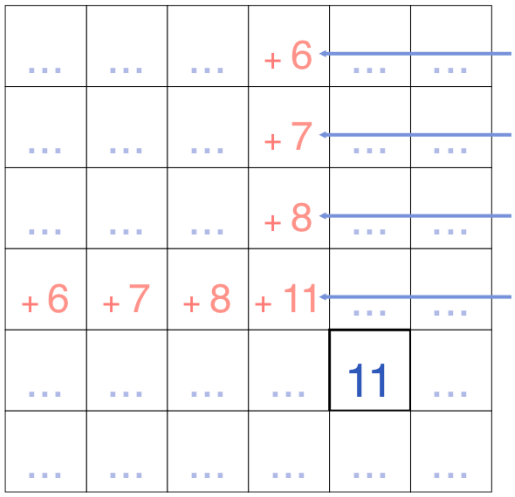
Qian and Goldstein (2001)[1] have shown that a log linear plot of gap probabilities in aligned sequences can be modeled by a sum of four exponential functions. This can be interpreted to mean that several molecular mechanisms could exist for the generation of indels, each with a distinct and characteristic probability of occurrence.

However, logarithmic gap penalties do not improve alignments (Cartwright, 2006)[2]. Recent developments focus on the inclusion of additional knowledge about the sequences, such as secondary-structure specific gap penalties, or using sequence profiles or multiple alignments, rather than aiming to further improve the gap parameters. The bottom line is: we have no good model for indels, but we have no significantly better model than the simple affine model.

[1] http://www.ncbi.nlm.nih.gov/pubmed/11536366

[2] http://www.ncbi.nlm.nih.gov/pubmed/17147805

Calculating **affine gap penalties** is computationally simple:
**reduce the score that is added to a cell according to the number of rows or columns that need to be skipped.**




Example parameters:
Gap insertion: -3
Gap extension: -1

Insert a gap,
extend by two: add score – 3 – 2

Insert a gap,
extend by one: add score – 3 – 1

Insert a gap: add score – 3

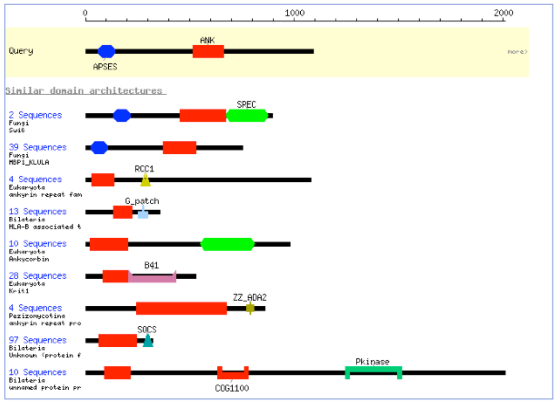
Ungapped continuation: add score to cell

## Pairwise Optimal alignment

- Reasonably fast for pairwise gene comparisons.
- Too slow / needs too much memory for database scans or whole genome alignments.
- Guaranteed to always give a mathematically optimal alignment.
- Alignment not guaranteed to be biologically correct or unique.
- Alignment will depend on scoring matrix.
- Alignment will strongly depend on (empirical !) gap insertion and extension parameters.

local and global ...

Often the score for an alignment between two substrings can be larger than the score for an alignment between two entire sequences. This is especially the case if a sequence has several domains.




The Smith-Waterman variation of the Needleman-Wunsch algorithm computes the highest scoring aligned *substrings*.

Always use local alignment -

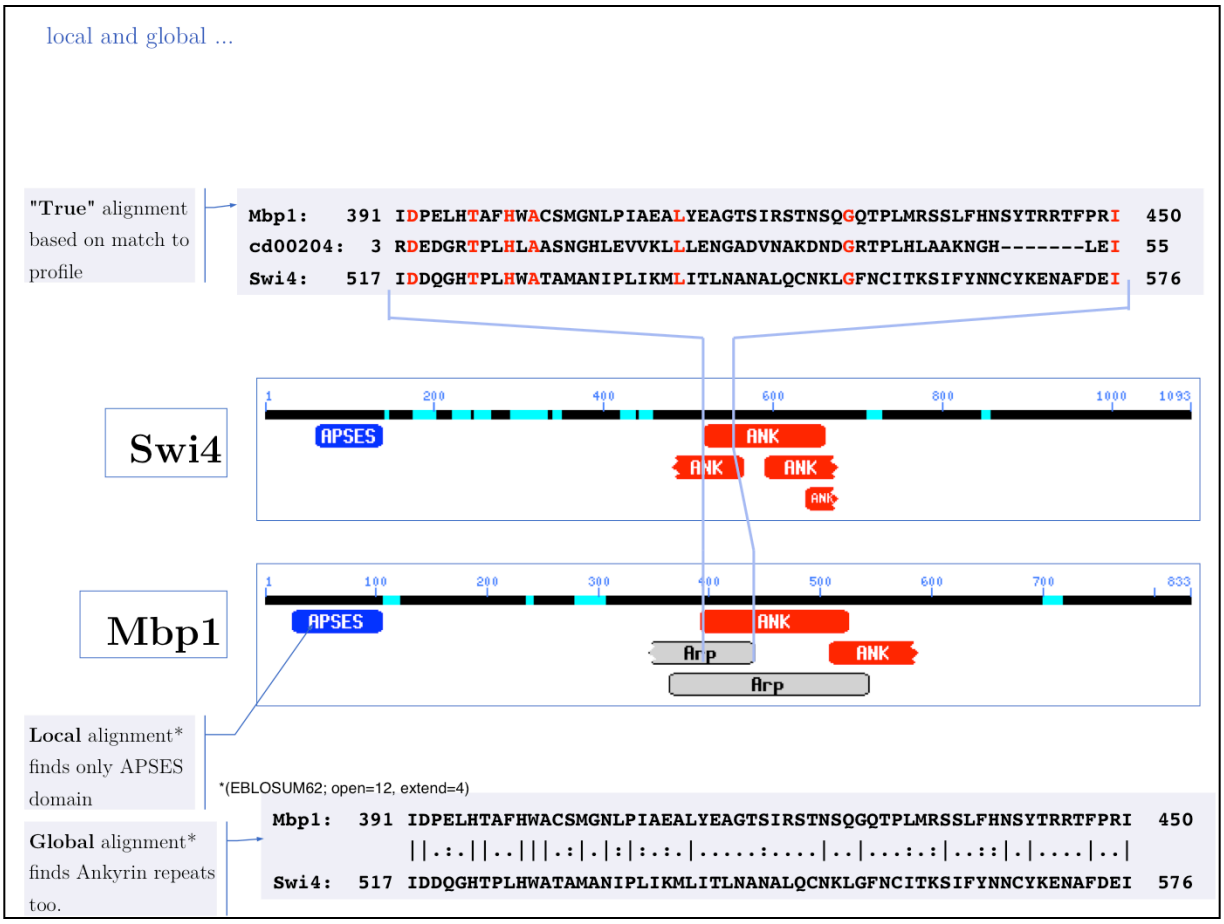
- when the sequences have very different lengths
- when the sequences are only related in domains or subdomains

In the example above, the ankyrin domain repeats of the yeast transcription factor Mbp1 are shown as a red box in this graphic of domains in sequence families, compiled in the **CDART database**[1]. This domain is found in many other proteins, but some of them do not share the other sequence elements found in Mbp1 - they are only partially related. Attempting a global sequence alignment with such sequences would attempt to align sequences that are actually not homologous, leading to inappropriately low scores and the danger of spurios results.

Temple Smith and Michael Waterman[2] have slightly modified the Needleman-Wunsch algorithm, 11 years after its publication, to find the highest scoring **local** alignment: this is the highest match in the matrix, tracked back to the point where the pathscore drops below zero. The rest of the algorithm works in exactly the same way. There is only one detail that needs to be considered: the substitution matrix must yield a negative expectation value for random alignments. If this were not the case, random pairs could extend the locally high-scoring alignment unreasonably.

[1] http://www.ncbi.nlm.nih.gov/pubmed/12368255

[2] http://en.wikipedia.org/wiki/Smith_Waterman_algorithm




In this example, Smith-Waterman local sequence alignment detects only the high-scoring similarity between Mbp1 and Swi4s APSES domains. The lower scoring, more highly diverged ankyrin repeats are missed by the algorithm. The Needleman-Wunsch alignment finds both sets of sequences, albeit there are segments in between that don't align well at all.

In this case, one **could** do a local alignment, remove the matching segments from the input sequences and then redo the alignment to see if any other significantly similar segments are found.

# When to use ...

## No alignment ...

Annotations of functional elements or domains may be conserved (e.g. TM-helices, phosphorylation sites, 2° structure, disordered segments ...). Especially significant if sequence divergence is otherwise large.

## Local alignment ...

Alignment in parts. Appropriate if sequences are homologous only in part, or if parts of the sequence are structurally dissimilar, or if inserted domains would create unrealistically large gap penalties. May need to be iterated.

## Global alignment ...

Appropriate if sequences are homologous over their whole length, especially to bridge segments of high divergence, and to discover islands of high similarity.

# What to do ...

## Rule 1: Only align sequence that is *homologous*

Always align domains (if known) separately.

## Rule 2: Only align sequence that is *conserved*.

Always align translated amino acid sequence–never nucleotide sequence–unless you are studying nucleotide variation.

Don't waste your time aligning gapped regions !

Of course the algorithms will optimally align anything you feed them, but for anything but homologous sequence **the alignment will be meaningless**. Aligning non-homologous sequences is a nice example of cargo-cult bioinformatics.

Therefore: if you already know that your proteins are multi-domain, separate out the domains before aligning. If you don't know, critically look at the results, generate a hypothesis about the domain structure and rerun your alignment on the domains separately. The exception, of course: is if you know (or believe) your two proteins comprise homologous domains in the same order.

Amino acid sequences are much more highly conserved then genomic sequence and even if you have nucleotide sequences to start from, you should always **translate** them before aligning. In general, many more matches are required to make nucleotide sequence matches significant, since the alphabet is much smaller. Also, there is no good notion of "similarity" or "conservative mutation" at the nucleotide level[1].

The only reasons to align nucleotides are:

- if you are actually interested in the **number and type of nucleotide exchanges**, such as in gene assembly and EST clustering, studies of SNPs, in comparative genomics, phylogenetic studies of closely related genes, or defining primer binding sites;
- if you are **aligning untranslated sequence**s; in particular if it is the nucleotide sequence itself that is conserved, such as in DNA binding sites or splice sites; or if you are studying RNA genes, such as tRNA or rRNA.

A corollary is that you should not try to align sequences in highly gapped regions. These residues have evolved in a non-comparable context, they cannot have been conserved by evolution for that reason and applying our scoring matrices cannot compare such residues in a meaningful way.

[1] However, transitions (conserving pyrimidines or purines) are more frequent than transversions. See http://en.wikipedia.org/wiki/Models_of_DNA_evolution for how this is modelled.

# How to set penalties ...

Higher opening penalties make gaps *less frequent*.

Higher extension penalties make gaps *shorter*.

The effect of the penalties depends on the scoring matrix!

Typical opening penalty: *2-3* times an *identity score*.

Typical extension penalty: $^{1}/_{5}$ to $^{1}/_{10}$ of an *opening penalty*.

Default penalties for BLOSUM62: -11 and -1 at *NCBI* (BLAST)
-10 and -0.5 at *EMBOSS* (Needle, Water)

# How to report results ...

The alignment score is a single number that measures the quality of the alignment. Scores depend on:

- the matrix
- the gap insertion penalty
- the gap extension penalty
- the end-gap penalty
- the algorithm (local or global, optimal or heuristic)

Therefore, all these parameters need to be reported along with the alignment (similarity) score, otherwise the number is meaningless.

Alternatively: *report % identity*! This allows a certain degree of comparison between alignments.

Note that reporting %-identity is an objective metric, but it still depends on the exact alignment that has been produced and it does not capture the quality of gaps.

decision thresholds

# How to interpret ...

No clear threshold exists for homology.

Homologous proteins can have as little as < 10% identity. (This is a problem).

Non-homologous proteins can have as much as > 50% identity over stretches of their alignment. (This is also a problem).

## Rules of Thumb:

More than 25% sequence identity over an entire domain (i.e >100 residues) almost always means *homologous*.

More than one indel per 20 residues almost always means non-homologous.

A Rule of Thumb does not replace sound judgement! Corroborating evidence can come from shared annotated function, conservation of conspicuous features (eg. C, H, W residues), multiple alignments ... Always examine alignments carefully: what is conserved but would not need to be if the sequences were not homologues? What is not conserved but would be expected to be if the sequences were homologues?

Identities of 20 to 25% are also called the ''twilight zone'' - in which homology is likely but can't be confidently inferred from sequence similarity alone.

These thresholds are based on sequence similarity after optimal alignment. Additional supporting evidence for homology can be contributed from:

- simlar length;
- similar functional sequence patterns (e.g. cys/his clusters);
- similar number of transmembrane helices;
- similar conservation patterns or conserved motifs;
- similar amino acid frequencies or bias (eg. polyglutamine, polyproline);
- similar patterns of disordered sequence;
- similar structure;
- similar function;
- similar genomic context;
- similar interactors;
- similar subcellular localization;
- [...]

## **Needle** - for optimal global alignments

## **Water** - for optimal local alignments

**stretche**r - for long sequences: half as fast as NW but only linear to the shorter sequence in memory.

**matcher** - for long sequences: slower than SW, but only linear to the shorter sequence in memory; also gives suboptimal matches.

**supermatche**r - rough results for very long sequences; heuristics, based on word matches.

Needle

Output of EMBOSS package global sequence alignment program *Needle* with sequences for yeast Swi4 and Mbp1.
(Default parameters).

The first ~200 amino acids of the alignment are shown, these include the conserved DNA binding domain.

- | - identical amino acids
- : - highly similar pairs
- . - similar pairs.

```
#=======================================
# Aligned_sequences: 2
# 1: Mbp1
# 2: Swi4
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 1147
# Identity:     266/1147 (23.2%)
# Similarity:   414/1147 (36.1%)
# Gaps:         368/1147 (32.1%)
# Score: 640.0
#=======================================

Mbp1     1 ------------------------------MSNQ--IYSARYSGVDVYE-  17
                                         .||.  |..|.||..||||
Swi4     1 MPFDVLISNQKDNTNHQNITPISKSVLLAPHSNHPVIEIATYSETDVYEC  50

Mbp1    18 FIH--STGSIMKRKKDDWVNATHILKAANFAKAKRTRILEKEVLKETHEK  65
           :|.  .|..:|:|.||||:|.|.:.|.|.|:|.|||:|||||.....|||
Swi4    51 YIRGFETKIVMRRTKDDWINITQVFKIAQFSKTKRTKILEKESNDMQHEK 100

Mbp1    66 VQGGFGKYQGTWVPLNIAKQLAEKFSVYDQLKPLFDFTQTDGSASPPPAP 115
           ||||:|::|||||:||:.||.|..|:.:.|.:.......|.|.:..||...
Swi4   101 VQGGYGRFQGTWIPLDSAKFLVNKYEIIDPVVNSILTFQFDPNNPPPKRS 150

Mbp1   116 KH---------------HHASKVDRKKAIRSASTSAIMETKRNNKKAEEN 150
           |:               ...:|..|||. .|:||||  .|...|||.::|
Swi4   151 KNSILRKTSPGTKITSPSSYNKTPRKKN-SSSSTSA--TTTAANKKGKKN 197

Mbp1   151 --------------------QFQ---SSKILGNPTAAPRKRGRPVGSTRG 177
                               |||   |..|:.|     ......:.....
Swi4   198 ASINQPNPSPLQNLVFQTPQQFQVNSSMNIMNN-----NDNHTTMNFNND 242
```

Water

Output of EMBOSS package local sequence alignment program *Water* with sequences for yeast Swi4 and Mbp1.
(Default parameters).

The first ~200 amino acids of the alignment are shown, these include the conserved DNA binding domain.

- | - identical amino acids
- : - highly similar pairs
- . - similar pairs.

```
#=======================================
# Aligned_sequences: 2
# 1: Mbp1
# 2: Swi4
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 1088
# Identity:     263/1088 (24.2%)
# Similarity:   411/1088 (37.8%)
# Gaps:         317/1088 (29.1%)
# Score: 644.5
#=======================================

Mbp1      8 ARYSGVDVYE-FIH--STGSIMKRKKDDWVNATHILKAANFAKAKRTRIL  54
            |.||..|||| :|.  .|..:|:|.||||:|.|.:.|.|.|:|.|||:||
Swi4     40 ATYSETDVYECYIRGFETKIVMRRTKDDWINITQVFKIAQFSKTKRTKIL  89

Mbp1     55 EKEVLKETHEKVQGGFGKYQGTWVPLNIAKQLAEKFSVYDQLKPLFDFTQ 104
            |||.....|||||||:|::||||:||:.||.|..|:.:.|.:.......|
Swi4     90 EKESNDMQHEKVQGGYGRFQGTWIPLDSAKFLVNKYEIIDPVVNSILTFQ 139

Mbp1    105 TDGSASPPPAPKH---------------HHASKVDRKKAIRSASTSAIME 139
            .|.:..||...|:               ...:|..|||. .|:||||  .
Swi4    140 FDPNNPPPKRSKNSILRKTSPGTKITSPSSYNKTPRKKN-SSSSTSA--T 186

Mbp1    140 TKRNNKKAEEN--------------------QFQ---SSKILGNPTAAPR 166
            |...|||.::|                    |||   |..|:.|     .
Swi4    187 TTAANKKGKKNASINQPNPSPLQNLVFQTPQQFQVNSSMNIMNN-----N 231

Mbp1    167 KRGRPVGSTRGSRRKLGVNL------------QRSQSDMGFPRPAIPNSS 204
            .....:.....:|..|..|:            |:|..:..|      |::
Swi4    232 DNHTTMNFNNDTRHNLINNISNNSNQSTIIQQQKSIHENSF------NNN 275
```

http://steipe.biochemistry.utoronto.ca/abc

B O R I S . S T E I P E @ U T O R O N T O . C A

DEPARTMENT OF BIOCHEMISTRY & DEPARTMENT OF MOLECULAR GENETICS
UNIVERSITY OF TORONTO, CANADA