Sequence Alignment

Why?

- Genome sequencing gives us new gene sequences
- Network biology gives us functional information on genes/proteins
- Analysis of mutants links unknown genes to diseases
- *Can we learn anything from other known sequences about our new gene/protein?*

BBC NEWS | Health | Autism gene breakthrough hailed
http://news.bbc.co.uk/1/hi/health/6369347.stm
autism
ntlworld Pubmed Google informatics JeS1 – Login News Apple Rumors
bbc.co.uk
Home TV Radio Talk Where I Live A-Z Index Search
UK version International version | About the versions
Low graphics | Accessibility help
BBC NEWS
WATCH The News in 2 minutes
News services Your news when you want it
News Front Page
World
UK
England
Northern Ireland
Scotland
Wales
Business
Politics
Health
Medical notes
Education
Science/Nature
Technology
Entertainment
Video and Audio
Have Your Say
Magazine
In Pictures
Country Profiles
Special Reports
RELATED BBC SITES
SPORT
WEATHER
CBBC NEWSROUND
E-mail this to a friend
Printable version

Last Updated: Monday, 19 February 2007, 03:15 GMT

Autism gene breakthrough hailed

Scientists have found new autism genes by scanning the largest collection of families with multiple cases of autism ever assembled.

The monumental task of studying the 1,200 families took more than 120 scientists from more than 50 institutions across 19 countries.

Chromosome 11 was identified as one of the culprits

The work, described in Nature Genetics, implicates a region of chromosome 11 and a specific gene called neurexin 1.

Experts say the findings should help with finding new autism treatments.

Autism is a complex brain disorder that inhibits a person's ability to communicate and develop social relationships, and is often accompanied by extreme behavioural challenges.

"These exciting results may represent a step on the way to further new treatments"
Child psychiatrist Professor Jonathan Green

SEE ALSO
- Autism-like disorder 'reversible' 08 Feb 07 | Health
- Gene flaw increases autism risk 28 Oct 06 | Health
- Cartoons to aid autistic children 09 Jan 07 | Health

RELATED INTERNET LINKS
- Nature Genetics
- Autism Speaks

The BBC is not responsible for the content of external internet sites

TOP HEALTH STORIES
- Doctor regulation shake-up plan
- Natural family planning 'effective'
- Egg donation decision due

| News feeds

MOST POPULAR STORIES NOW

MOST E-MAILED | MOST READ

1. Britney Spears 'back into rehab'
2. Celebrities raise $1.3m for Obama
3. PM denies road toll 'stealth tax'

What is it?

```
ACCGGTATCCTAGGAC
ACCTATCTTAGGAC
```

Are these two sequences related?

How similar (or dissimilar) are they?

What is it?

```
ACCGGTATCCTAGGAC
|||  |||| ||||||
ACC--TATCTTAGGAC
```

- Match the two sequences as closely as possible = aligned
- Therefore, alignments need a score

Why do we care?

- DNA and Proteins are based on linear sequences
- Information is encoded in these sequences
- All bioinformatics at some level comes back to matching sequences that might have some noise or variability

Alignment Types

- Global: used to compare to similar sized sequences.
  - Compare closely related genes
  - Search for mutations or polymorphisms in a sequence compared to a reference.

Armstrong, 2008

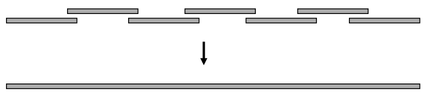
Alignment Types

- Local: used to find shared subsequences.
  - Search for protein domains
  - Find gene regulatory elements
  - Locate a similar gene in a genome sequence.

Armstrong, 2008

Alignment Types

- Ends Free: used to find joins/overlaps.
  - Align the sequences from adjacent sequencing primers.



Armstrong, 2008

How do we score alignments?

ACCGG**TATC**C**TAGGAC**

```
|||  |||| ||||||
```

ACC--TATCT**TAGGAC**

- Assign a score for each match along the sequence.

Armstrong, 2008

How do we score alignments?

ACCGGTATC**C**TAGGAC

```
|||  |||| ||||||
```

ACC--TATC**T**TAGGAC

- Assign a score (or penalty) for each substitution.

Armstrong, 2008

How do we score alignments?

ACC**GG**TATCCTAGGAC

```
|||  |||| ||||||
```

ACC**--**TATCTTAGGAC

- Assign a score (or penalty) for each insertion or deletion.
- insertions/deletions otherwise known as indels

Armstrong, 2008

How do we score alignments?

```
ACCGGTATCCTAGGAC
|||  |||| ||||||
ACC--TATCTTAGGAC
```

- Matches and substitutions are ‘easy’ to deal with.
  - We’ll look at substitution matrices later.
- How do we score indels: gaps?

Armstrong, 2008

How do we score gaps?

```
ACCGGTATCC---GAC
|||  ||||     |||
ACC--TATCTTAGGAC
```

- A gap is a consecutive run of indels
- The gap length is the number of indels.
- The simple example here has two gaps of length 2 and 3

Armstrong, 2008

How do we score gaps?

```
ACCGGTATCC---GAC
|||  ||||     |||
ACC--TATCTTAGGAC
```

• Constant:	Length independent weight
• Affine:	*Open* and *Extend* weights.
• Convex:	Each additional gap contributes less
• Arbitrary:	Some arbitrary function on length

Armstrong, 2008

Choosing Gap Penalties

- The choice of Gap Scoring Penalty is very sensitive to the context in which it is applied:
  - introns vs exons
  - protein coding regions
  - mis-matches in PCR primers

Armstrong, 2008

Substitution Matrices

- Substitution matrices are used to score substitution events in alignments.
- Particularly important in Protein sequence alignments but relevant to DNA sequences as well.
- Each scoring matrix represents a particular theory of evolution

Armstrong, 2008

Similarity/Distance

- Distance is a measure of the cost or replacing one residue with another.
- Similarity is a measure of how similar a replacement is.

e.g. replacing a hydrophobic residue with a hydrophilic one.

- The logic behind both are the same and the scoring matrices are interchangeable.

Armstrong, 2008

DNA Matrices

Identity matrix

	A	C	G	T
A	1	0	0	0
C	0	1	0	0
G	0	0	1	0
T	0	0	0	1

BLAST

	A	C	G	T
A	5	-4	-4	-4
C	-4	5	-4	-4
G	-4	-4	5	-4
T	-4	-4	-4	5

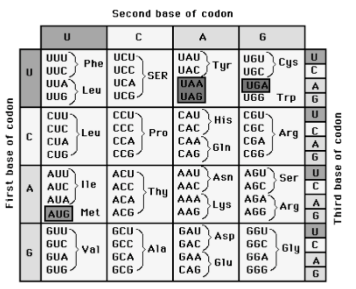
However, some changes are more likely to occur than others (even in DNA). When looking at distance, the ease of mutation is a factor. a.g. A-T and A-C replacements are rarer than A-G or C-T.

Protein Substitution Matrices

How can we score a substitution in an aligned sequence?

- Identity matrix like the simple DNA one.
- Genetic Code Matrix:

For this, the score is based upon the minimum number of DNA base changes required to convert one amino acid into the other.



Second base of codon

First base of codon	U	C	A	G	Third base of codon
U	UUU Phe	UCU SER	UAU Tyr	UGU Cys	U
	UUC Phe	UCC SER	UAC Tyr	UGC Cys	C
	UUA Leu	UCA SER	UAA	UGA	A
	UUG Leu	UCG SER	UAG	UGG Trp	G
C	CUU Leu	CCU Pro	CAU His	CGU Arg	U
	CUC Leu	CCC Pro	CAC His	CGC Arg	C
	CUA Leu	CCA Pro	CAA Gln	CGA Arg	A
	CUG Leu	CCG Pro	CAG Gln	CGG Arg	G
A	AUU Ile	ACU Thy	AAU Asn	AGU Ser	U
	AUC Ile	ACC Thy	AAC Asn	AGC Ser	C
	AUA Ile	ACA Thy	AAA Lys	AGA Arg	A
	AUG Met	ACG Thy	AAG Lys	AGG Arg	G
G	GUU Val	GCU Ala	GAU Asp	GGU Gly	U
	GUC Val	GCC Ala	GAC Asp	GGC Gly	C
	GUA Val	GCA Ala	GAA Glu	GGA Gly	A
	GUG Val	GCG Ala	CAG Glu	GGG Gly	G

The genetic code, written by convention in the form in which the Codons appear in mRNA. The three terminator codons, UAA, UAG, and UGA, are boxed in red; the AUG initiator codon is shown in green.

Protein Substitution Matrices

How can we score a substitution in an aligned sequence?

- Amino acid property matrix

Assign arbitrary values to the relatedness of different amino acids:

e.g. hydrophobicity , charge, pH, shape, size

Matrices based on Probability

$$S_{ij} = \log (q_{ij}/p_i p_j)$$

Sij is the log odds ratio of two probabilities: amino acids i and j are aligned by evolutionary descent and the probability that they aligned at random**.**

This is the basis for commonly used substitution matrices.

PAM matrices

Dayhoff, Schwarz and Orcutt 1978 took these into consideration when constructing the PAM matrices:

Took 71 protein families - where the sequences differed by no more than 15% of residues (i.e. 85% identical)

Aligned these proteins

Build a theoretical phylogenetic tree

Predicted the most likely residues in the ancestral sequence

PAM Matrices

- Ignore evolutionary direction
- Obtained frequencies for residue X being substituted by residue Y over time period Z
- Based on 1572 residue changes
- They defined a substitution matrix as 1 PAM (point accepted mutation) if the expected number of substitutions was 1% of the sequence length.

PAM Matrices

To increase the distance, they multiplied the the PAM1 matrix.

PAM250 is one of the most commonly used.

PAM - notes

The PAM matrices are rooted in the original datasets used to create the theoretical trees

They work well with closely related sequences

Based on data where substitutions are most likely to occur from single base changes in codons.

PAM - notes

Biased towards conservative mutations in the DNA sequence (rather than amino acid substitutions) that have little effect on function/structure.

Replacement at any site in the sequence depends only on the amino acid at that site and the probability given by the table.This does not represent evolutionary processes correctly. Distantly related sequences usually have regions of high conservation (blocks).

PAM - notes

36 residue pairs were not observed in the dataset used to create the original PAM matrix

A new version of PAM was created in 1992 using 59190 substitutions: Jones, Taylor and Thornton 1992 CAMBIOS 8 pp 275

BLOSUM matrices

Henikoff and Henikoff 1991

Took sets of aligned ungapped regions from protein families from the BLOCKS database.

The BLOCKS database contain short protein sequences of high similarity clustered together. These are found by applying the MOTIF algorithm to the SWISS-PROT and other databases. The current release has 8656 Blocks.

BLOSUM matrices

Sequences were clustered whenever the %identify exceeded some percentage level.

Calculated the frequency of any two residues being aligned in one cluster also being aligned in another

Correcting for the size of each cluster.

Armstrong, 2008

BLOSUM matrices

Resulted in the fraction of observed substitutions between any two residues over all observed substitutions.

The resulting matrices are numbered inversely from the PAM matrices so the BLOSUM50 matrix was based on clusters of sequence over 50% identity, and BLOSUM62 where the clusters were at least 62% identical.

Armstrong, 2008

BLOSUM 62 Matrix

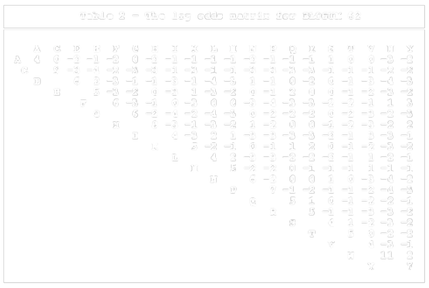


Table 2 – The log odds matrix for BLOSUM 62

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	4	0	−2	−1	−2	0	−2	−1	−1	−1	−1	−2	−1	−1	−1	1	0	0	−3	−2
C		9	−3	−4	−2	−3	−3	−1	−3	−1	−1	−3	−3	−3	−3	−1	−1	−1	−2	−2
D			6	2	−3	−1	−1	−3	−1	−4	−3	1	−1	0	−2	0	−1	−3	−4	−3
E				5	−3	−2	0	−3	1	−3	−2	0	−1	2	0	0	−1	−2	−3	−2
F					6	−3	−1	0	−3	0	0	−3	−4	−3	−3	−2	−2	−1	1	3
G						6	−2	−4	−2	−4	−3	0	−2	−2	−2	0	−2	−3	−2	−3
H							8	−3	−1	−3	−2	1	−2	0	0	−1	−2	−3	−2	2
I								4	−3	2	1	−3	−3	−3	−3	−2	−1	3	−3	−1
K									5	−2	−1	0	−1	1	2	0	−1	−2	−3	−2
L										4	2	−3	−3	−2	−2	−2	−1	1	−2	−1
M											5	−2	−2	0	−1	−1	−1	1	−1	−1
N												6	−2	0	0	1	0	−3	−4	−2
P													7	−1	−2	−1	−1	−2	−4	−3
Q														5	1	0	−1	−2	−2	−1
R															5	−1	−1	−3	−3	−2
S																4	1	−2	−3	−2
T																	5	0	−2	−2
V																		4	−3	−1
W																			11	2
Y																				7

Armstrong, 2008

Summary so far…

- Gaps
  - Indel operations
  - Gap scoring methods
- Substitution matrices
  - DNA largely simple matrices
  - Protein matrices are based on probability
  - PAM and BLOSUM

Armstrong, 2008

How do we do it?

- Like everything else there are several methods and choices of parameters
- The choice depends on the question being asked
  - What kind of alignment?
  - Which substitution matrix is appropriate?
  - What gap-penalty rules are appropriate?
  - Is a heuristic method good enough?

Armstrong, 2008

Working Parameters

- For proteins, using the affine gap penalty rule and a substitution matrix:

Query Length	Matrix	Gap (open/extend)
<35	PAM-30	9,1
35-50	PAM-70	10,1
50-85	BLOSUM-80	10,1
>85	BLOSUM-62	11,1

Armstrong, 2008

Alignment Types

- Global: used to compare to similar sized sequences.

- Local: used to find similar subsequences.

- Ends Free: used to find joins/overlaps.

Global Alignment

- Two sequences of similar length
- Finds the best alignment of the two sequences
- Finds the score of that alignment
- Includes **ALL** bases from both sequences in the alignment and the score.

- Needleman-Wunsch algorithm

Needleman-Wunsch algorithm

- Gaps are inserted into, or at the ends of each sequence.
- The sequence length (bases+gaps) are identical for each sequence
- Every base or gap in each sequence is aligned with a base or a gap in the other sequence

Needleman-Wunsch algorithm

- Consider 2 sequences *S* and *T*
- Sequence *S* has *n* elements
- Sequence *T* has *m* elements
- Gap penalty ?

How do we score gaps?

```
ACCGGTATCC---GAC
|||  ||||    |||
ACC--TATCTTAGGAC
```

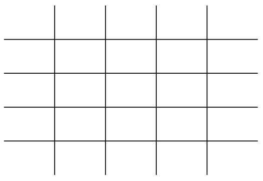
- Constant: Length independent weight
- Affine: *Open* and *Extend* weights.
- Convex: Each additional gap contributes less
- Arbitrary: Some arbitrary function on length
  - Lets score each gap as –1 times length

Needleman-Wunsch algorithm

- Consider 2 sequences *S* and *T*
- Sequence *S* has *n* elements
- Sequence *T* has *m* elements
- Gap penalty –1 per base (arbitrary gap penalty)
- An alignment between base *i* in *S* and a gap in *T* is represented: $(S_i,-)$
- The score for this is represented : $\sigma(S_i,-) = -1$

Needleman-Wunsch algorithm

- Substitution/Match matrix for a simple alignment
- Several models based on probability….



Needleman-Wunsch algorithm

- Substitution/Match matrix for a simple alignment
- Simple identify matrix (2 for match, -1 for mismatch)
- An alignment between base *i* in *S* and base *j* in *T* is represented: (S_i,T_j)
- The score for this occurring is represented: $\sigma(S_i,T_j)$

Needleman-Wunsch algorithm

- Set up a array V of size n+1 by m+1
- Row 0 and Column 0 represent the cost of adding gaps to either sequence at the start of the alignment
- Calculate the rest of the cells row by row by finding the optimal route from the surrounding cells that represent a gap or match/mismatch
  - This is easier to demonstrate than to explain

Needleman-Wunsch algorithm

– lets start by trying out a simple example alignment:

```
S = ACCGGTAT
T =  ACCTATC
```

Needleman-Wunsch algorithm

– Get lengths

```
S = ACCGGTAT
T =  ACCTATC
```

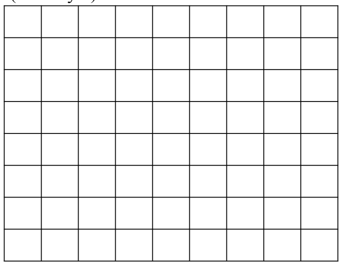
Length of *S* = *m* = 8

Length of T = *n* = 7

(lengths approx equal so OK for Global Alignment)

Create array m+1 by n+1

(i.e. 9 by 8)



Add on bases from each sequence

		A	C	C	G	G	T	A	T	(S)
A										
C										
C										
T										
A										
T										
C										
(T)										

Armstrong, 2008

Represent scores for gaps in row/ col 0

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2							
A										
C										
C										
T										
A										
T										
C										
(T)										

Armstrong, 2008

Represent scores for gaps in row/ col 0

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1									
C	-2									
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

Armstrong, 2008

For each cell consider the 'best' path

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	▨								
C	-2									
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

Armstrong, 2008

For each cell consider the 'best' path

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3						
A	-1									
C										
C										
T										
A										
T										
C										
(T)										

(S_1,T_0) & σ(-,T_1) = -1
Running total (-1+-1)=-2

Armstrong, 2008

For each cell consider the 'best' path

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3						
A	-1									
C										
C										
T										
A										
T										
C										
(T)										

(S_1,T_0) & σ(-,T_1) = -1
Running total (-1+-1)=-2

(S_0, T_1) & σ(S_1,-) = -1
Running total (-1+-1)=-2

Armstrong, 2008

For each cell consider the 'best' path

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3						
A	-1									
C										
C										
T										
A										
T										
C										
(T)										

(S_1,T_0) & σ(-,T_1) = -1
Running total (-1+-1)=-2

(S_0,T_0) & σ(S_1,T_1) = 2
Running total (0+2)=2

(S_0, T_1) & σ(S_1,-) = -1
Running total (-1+-1)=-2

Armstrong, 2008

Choose and record 'best' path

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3						
A	-1	2								
C										
C										
T										
A										
T										
C										
(T)										

Armstrong, 2008

Choose and record 'best' path

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3						
A	-1	2	1							
C										
C										
T										
A										
T										
C										
(T)										

(S_2,T_0) & σ(-,T_1)
Running total (-2+-1)=-3

(S_1,T_0) & σ(S_2,T_1)
Running total (-1+-1)=-2

(S_1,T_1) & σ(S_2,-)
Running total (2+-1)=1

Armstrong, 2008

Continue….

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4		
C	-2									
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

Armstrong, 2008

Continue….

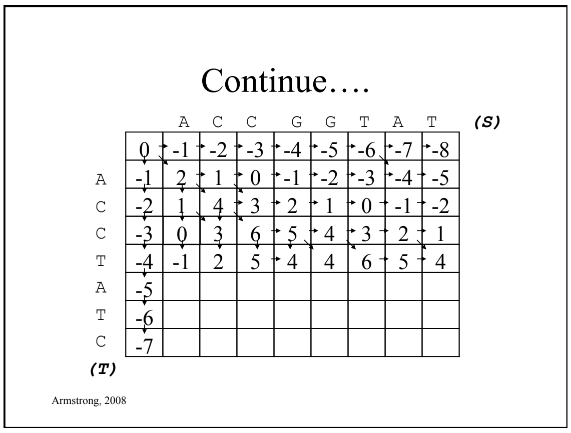
		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

Armstrong, 2008

Continue….

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

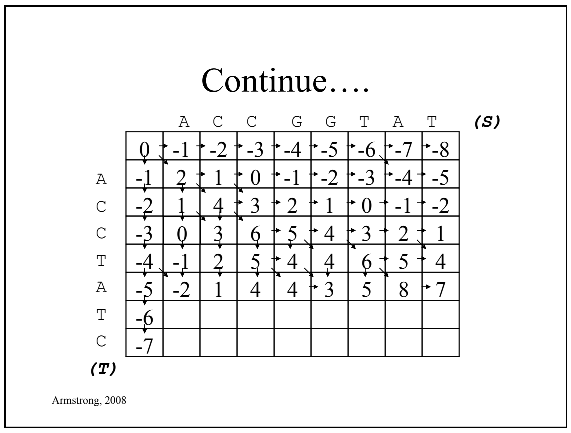
Armstrong, 2008



Continue….

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5									
T	-6									
C	-7									
(T)										

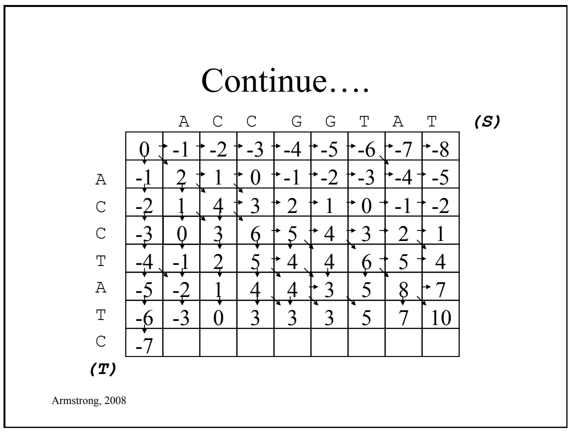
Armstrong, 2008



Continue….

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5	-2	1	4	4	3	5	8	7	
T	-6									
C	-7									
(T)										

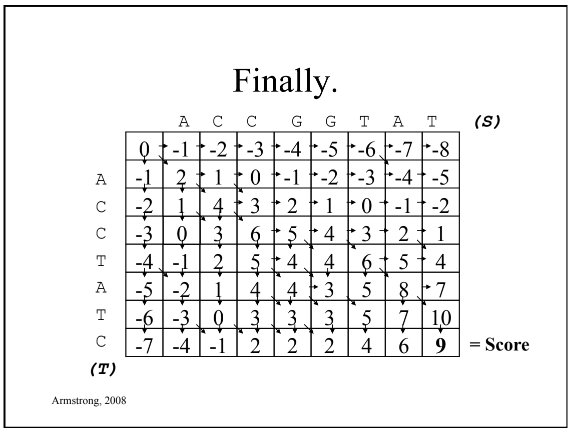
Armstrong, 2008



Continue….

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5	-2	1	4	4	3	5	8	7	
T	-6	-3	0	3	3	3	5	7	10	
C	-7									
(T)										

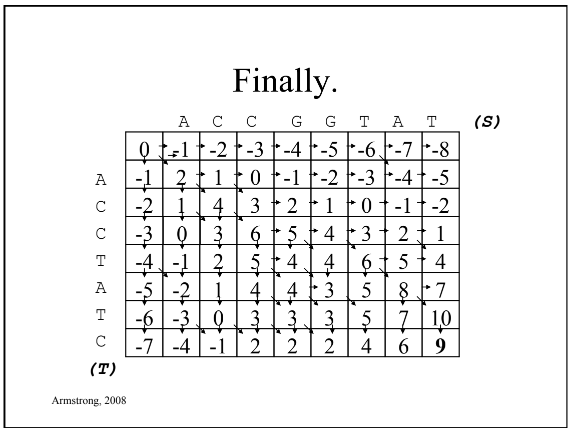
Armstrong, 2008



Finally.

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5	-2	1	4	4	3	5	8	7	
T	-6	-3	0	3	3	3	5	7	10	
C	-7	-4	-1	2	2	2	4	6	**9**	**= Score**
(T)										

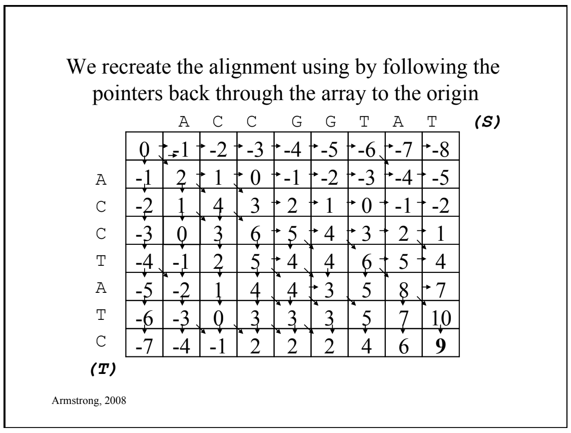
Armstrong, 2008



Finally.

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5	-2	1	4	4	3	5	8	7	
T	-6	-3	0	3	3	3	5	7	10	
C	-7	-4	-1	2	2	2	4	6	**9**	
(T)										

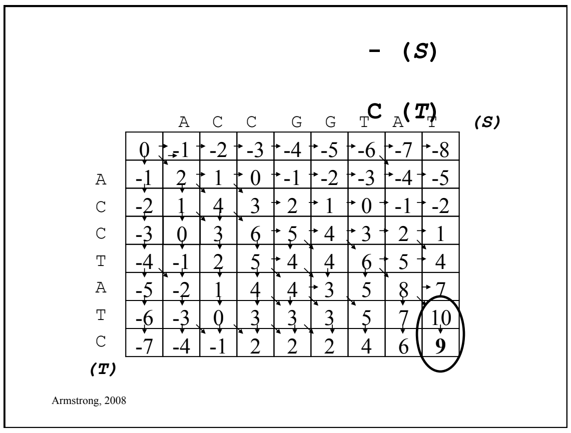
Armstrong, 2008



We recreate the alignment using by following the pointers back through the array to the origin

		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	4	4	6	5	4	
A	-5	-2	1	4	4	3	5	8	7	
T	-6	-3	0	3	3	3	5	7	10	
C	-7	-4	-1	2	2	2	4	6	**9**	
(T)										

Armstrong, 2008



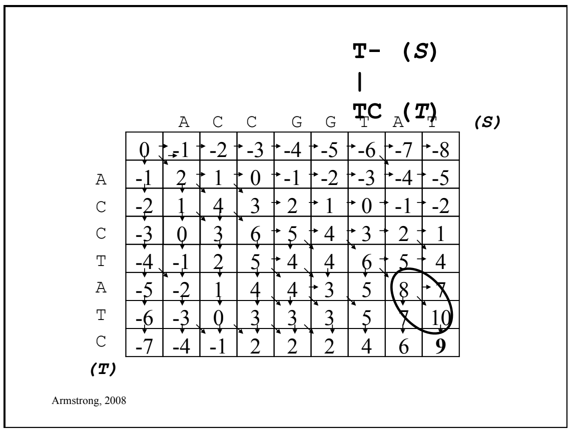
```
    - (S)

    C (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008



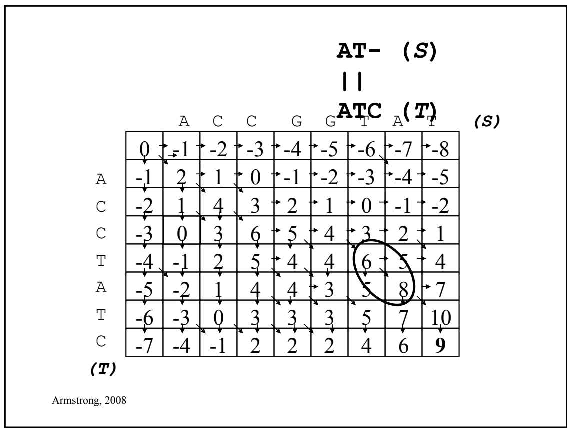
```
   T- (S)
   |
   TC (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008



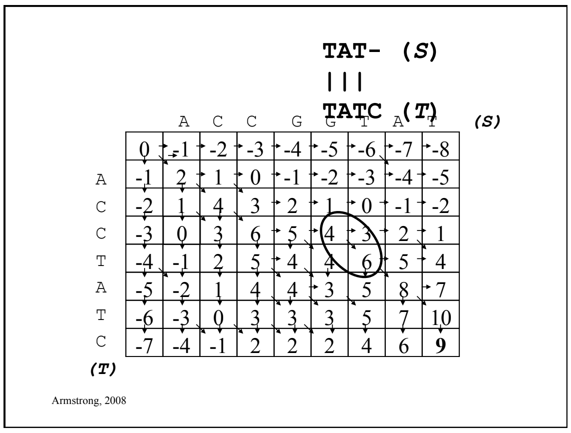
```
   AT- (S)
   ||
   ATC (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008



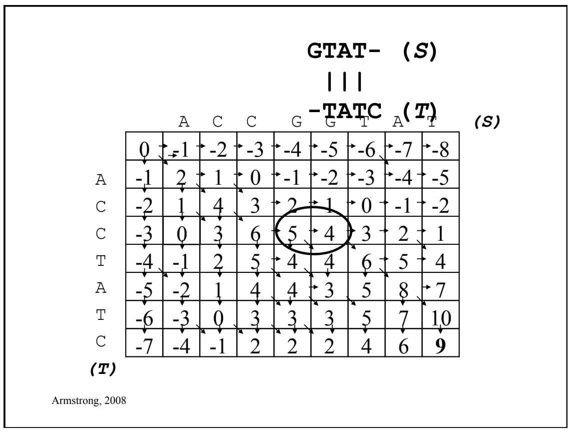
```
   TAT- (S)
   |||
   TATC (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008



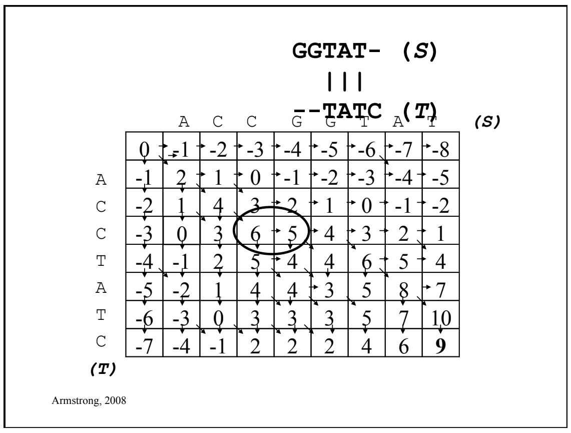
```
   GTAT- (S)
    |||
   -TATC (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008



```
   GGTAT- (S)
     |||
   --TATC (T)
```

(S)

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(T)

Armstrong, 2008

```
CGGTAT-  (S)
|   |||
C--TATC  (T)
```

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(S) *(T)*

```
CCGGTAT-  (S)
||   |||
CC--TATC  (T)
```

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(S) *(T)*

```
ACCGGTAT-  (S)
|||   |||
ACC--TATC  (T)
```

		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	4	4	6	5	4
A	-5	-2	1	4	4	3	5	8	7
T	-6	-3	0	3	3	3	5	7	10
C	-7	-4	-1	2	2	2	4	6	**9**

(S) *(T)*

Checking the result

```
ACCGGTAT-  (S)
|||   |||
ACC--TATC  (T)
```

- Our alignment considers ALL bases in each sequence
- 6 matches = 12 points, 3 gaps = -3 points
- Score = 9 confirmed.

A bit more formally..

Base conditions:

$$V(i,0) = \sum_{k=0}^{i} \sigma(S_k,-)$$

$$V(0,j) = \sum_{k=0}^{j} \sigma(-,T_k)$$

Recurrence relation: for $1 <= i <= n$, $1 <= j <= m$:

$$V(i,j) = \max \begin{cases} V(i-1,j-1) + \sigma(S_i,T_j) \\ V(i-1,j) + \sigma(S_i,-) \\ V(i,j-1) + \sigma(-,T_j) \end{cases}$$

Time Complexity

- Each cell is dependant on three others and the two relevant characters in each sequence
- Hence each cell takes a constant time
- (n+1) x (m+1) cells
- Complexity is therefore O(nm)

Space Complexity

- To calculate each row we need the current row and the row above only.
- Therefore to get the score, we need O(n+m) space
- However, if we need the pointers as well, this increases to O(nm) space
- This is a problem for very long sequences
  - think about the size of whole genomes

Global alignment in linear space

- Hirschberg 1977 applied a ‘divide and conquer’ algorithm to Global Alignment to solve the problem in linear space.
- Divide the problem into small manageable chunks
- The clever bit is finding the chunks

dividing...

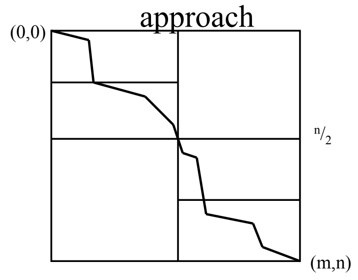
Compute matrix V(A,B) saving the values for $^n/_2$th row
- call this matrix F

Compute matrix V(A^r,B^r) saving the values for $^n/_2$th row
- call this matrix B

Find column *k* so that the crossing point ($^n/_2$,*k*) satisfies:
$F(^n/_2,k) + B(^n/_2,m-k) = F(n,m)$

Now we have two much smaller problems:
(0,0) -> ($^n/_2$,*k*) and (n,m) -> ($^n/_2$,*m-k*)

Hirschberg’s divide and conquer approach



Complexity

- After applying Hirschberg’s divide and conquer approach we get the following:
  - Complexity O(mn)
  - Space O(min(m,n))
- For the proofs, see D.S. Hirschberg. (1977) Algorithms for the longest common subsequence problem. J. A.C.M 24: 664-667

OK where are we?

- The Needleman-Wunsch algorithm finds the optimum alignment and the best score.
  - NW is a dynamic programming algorithm
- Space complexity is a problem with NW
- Addressed by a divide and conquer algorithm
- What about local and ends-free alignments?

Smith-Waterman algorithm

- Between two sequences, find the best two subsequences and their score.
- We want to ignore badly matched sequence
- Use the same types of substitution matrix and gap penalties
- Use a modification of the previous dynamic programming approach.

Smith-Waterman algorithm

- If S_i matches T_j then $\sigma(S_i,T_j) >= 0$
- If they do not match or represent a gap then <=0

- Lowest allowable value of any cell is 0
- Find the cell with the highest value (*i,j*) and extend the alignment back to the first zero value
- The score of the alignment is the value in that cell
- A quick example if best...

min value of any cell is 0

		A	C	C	G	G	T	A	T	*(S)*
	0	0	0	0	0	0	0	0	0	
T	0									
T	0									
G	0									
T	0									
A	0									
T	0									
C	0									
(T)										

min value of any cell is 0

		A	C	C	G	G	T	A	T	*(S)*
	0	0	0	0	0	0	0	0	0	
T	0	0	0	0	0	0	2	1	2	
T	0	0	0	0	0	0	2	1	3	
G	0									
T	0									
A	0									
T	0									
C	0									
(T)										

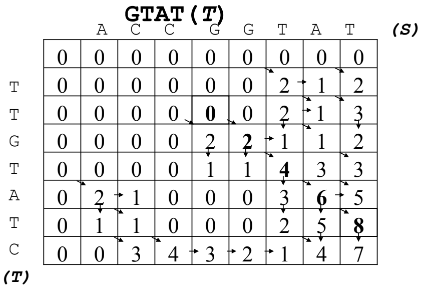
min value of any cell is 0

		A	C	C	G	G	T	A	T	*(S)*
	0	0	0	0	0	0	0	0	0	
T	0	0	0	0	0	0	2	1	2	
T	0	0	0	0	0	0	2	1	3	
G	0	0	0	0	2	2	1	1	2	
T	0	0	0	0	1	1	4	3	3	
A	0	2	1	0	0	0	3	6	5	
T	0	1	1	0	0	0	2	5	8	
C	0	0	3	4	3	2	1	4	7	
(T)										

Find biggest cell and map alignment from there

		A	C	C	G	G	T	A	T	*(S)*
	0	0	0	0	0	0	0	0	0	
T	0	0	0	0	0	0	2	1	2	
T	0	0	0	0	**0**	0	2	1	3	
G	0	0	0	0	2	**2**	1	1	2	
T	0	0	0	0	1	1	**4**	3	3	
A	0	2	1	0	0	0	3	**6**	5	
T	0	1	1	0	0	0	2	5	**8**	
C	0	0	3	4	3	2	1	4	7	
(T)										

```
GTAT(S)
||||
GTAT(T)
```



		A	C	C	G	G	T	A	T
	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	2	1	2
T	0	0	0	0	**0**	0	2	1	3
G	0	0	0	0	2	**2**	1	1	2
T	0	0	0	0	1	1	**4**	3	3
A	0	2	1	0	0	0	3	**6**	5
T	0	1	1	0	0	0	2	5	**8**
C	0	0	3	4	3	2	1	4	7

(S)

(T)

Smith-Waterman cont’d

- Complexity
  - Time is O(nm) as in global alignments
  - Space is O(nm) as in global alignments
  - A mod of Hirschbergs algorithm allows O(n+m) (n +m) as two rows need to be stored at a time instead of one as in the global alignment.

A bit more formally..

Base conditions: $\forall i,j.\ V(i,0) = 0,\ V(0,j) = 0$

Recurrence relation: for $1 <= i <= n$, $1 <= j <= m$:

$$V(i,j) = \max \begin{cases} 0 \\ V(i-1,j-1) + \sigma(S_i,T_j) \\ V(i-1,j) + \sigma(S_i,-) \\ V(i,j-1) + \sigma(-,T_j) \end{cases}$$

Compute i^* and j^* $V(i^*,j^*) = \max_{1<=i<=n,1<=j<=m} V(i,j)$

Ends-free alignment

- Find the overlap between two sequences such start the start of one overlaps is in the alignment and the end of the other is in the alignment.
- Essential to DNA sequencing strategies.
  - Building genome fragments out of shorter sequencing data.
- Another variant of the Global Alignment Problem

Ends-free alignment

- Set the initial conditions to zero weight
  - allow indels/gaps at the ends without penalty
- Fill the array/table using the same recursion model used in global/local alignment
- Find the best alignment that ends in one row or column
  - trace this back

min value row0 & col0 is 0

		G	T	T	A	C	T	G	T
	0	0	0	0	0	0	0	0	0
C	0	-1	-1	-1	-1	2	1	0	-1
T	0	-1	1	1	0	1	4	3	2
G	0	2	1	0	0	0	3	6	5
T	0	1	4	3	2	1	2	5	8
A	0	0	3	3	5	4	3	4	7
T	0	-1	2	5	4	4	6	5	6
C	0	0	1	4	4	6	5	5	5

(S)

(T)

Find the best 'end' point in an end col or row

		G	T	T	A	C	T	G	T	(S)
	0	0	0	0	0	0	0	0	0	
C	0	-1	-1	-1	-1	2	1	0	-1	
T	0	-1	1	1	0	1	4	3	2	
G	0	2	1	0	0	0	3	6	5	
T	0	1	4	3	2	1	2	5	8	
A	0	0	3	3	5	4	3	4	7	
T	0	-1	2	5	4	4	6	5	6	
C	0	0	1	4	4	6	5	5	5	
(T)										

Trace the best route from there to the origin and end

		G	T	T	A	C	T	G	T	(S)
	0	0	0	0	0	0	0	0	0	
C	0	-1	-1	-1	-1	2	1	0	-1	
T	0	-1	1	1	0	1	4	3	2	
G	0	2	1	0	0	0	3	6	5	
T	0	1	4	3	2	1	2	5	8	
A	0	0	3	3	5	4	3	4	7	
T	0	-1	2	5	4	4	6	5	6	
C	0	0	1	4	4	6	5	5	5	
(T)										

```
GTTACTGT---(S)
    ||||
----CTGTATC(T)
```

		G	T	T	A	C	T	G	T	(S)
	0	0	0	0	0	0	0	0	0	
C	0	-1	-1	-1	-1	2	1	0	-1	
T	0	-1	1	1	0	1	4	3	2	
G	0	2	1	0	0	0	3	6	5	
T	0	1	4	3	2	1	2	5	8	
A	0	0	3	3	5	4	3	4	7	
T	0	-1	2	5	4	4	6	5	6	
C	0	0	1	4	4	6	5	5	5	
(T)										

A bit more formally..

Base conditions: $\forall i,j.\ V(i,0) = 0,\ V(0,j) = 0$

Recurrence relation: for $1<=i <= n$, $1<=j<=m$:

$$V(i,j) = \max \begin{cases} V(i\text{-}1,j\text{-}1) + \sigma(S_i,T_j) \\ V(i\text{-}1,j) + \sigma(S_i,\text{-}) \\ V(i,j\text{-}1) + \sigma(\text{-},T_j) \end{cases}$$

Search for i^* such that: $V(i^*,m)=\max_{1<=i<=n,m} V(i,j)$
Search for j^* such that: $V(n,j^*)=\max_{1<=j<=n,m} V(i,j)$

Define alignment score $V(S,T) = \max \begin{cases} V(n,j^*) \\ V(i^*,m) \end{cases}$

Summary so far...

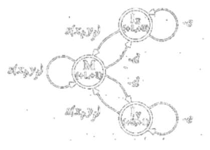
- Dynamic programming algorithms can solve global, local and ends-free alignment
- They give the optimum score and alignment using the parameters given
- Divide and conquer approaches make the space complexity manageable for small-medium sized sequences

Dynamic Programming Issues

- For huge sequences, even linear space constraints are a problem.
- We used a very simple gap penalty
- The Affine Gap penalty is most commonly used.
  - Cost to open a gap
  - Cost to extend an open gap
- Need to track and evaluate the 'gap' state in the array

Tracking the gap state

- We can model the matches and gap insertions as a finite state machine:

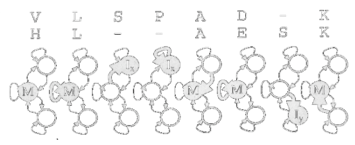


Taken from Durbin, chapter 2.4

Armstrong, 2008

Tracking the gap state

- Working along the alignment process...



Taken from Durbin, chapter 2.4

Armstrong, 2008

Real Life Sequence Alignment

- When searching multiple genomes, the sizes still get too big!
- Several approaches have been tried:
- Use huge parallel hardware:
  - Distribute the problem over many CPUs
  - Very expensive
- Implement in Hardware
  - Cost of specialist boards is high
  - Has been done for Smith-Waterman on SUN

Armstrong, 2008

Real Life Sequence Alignment

- Use a Heuristic Method
  - Faster than ‘exact’ algorithms
  - Give an approximate solution
  - Software based therefore cheap

- Based on a number of assumptions:

Armstrong, 2008

Assumptions for Heuristic Approaches

- Even linear time complexity is a problem for large genomes
- Databases can often be pre-processed to a degree
- Substitutions more likely than gaps
- Homologous sequences contain a lot of substitutions without gaps which can be used to help find start points in alignments

Armstrong, 2008

Conclusions

- Dynamic programming algorithms are expensive but they give you the optimum alignment and exact score
- Choice of GAP penalty and substitution matrix are critically important
- Heuristic approaches are generally required for high throughput or very large alignments

Armstrong, 2008

Heuristic Methods

- FASTA
- BLAST
- Gapped BLAST
- PSI-BLAST

Assumptions for Heuristic Approaches

- Even linear time complexity is a problem for large genomes
- Databases can often be pre-processed to a degree
- Substitutions more likely than gaps
- Homologous sequences contain a lot of substitutions without gaps which can be used to help find start points in alignments

BLAST

Altschul, Gish, Miller, Myers and Lipman (1990) Basic local alignment search tool. J Mol Biol 215:403-410

- Developed on the ideas of FASTA
  - uses short identical matches to reduce search = hotspot
- Integrates the substitution matrix in the first stage of finding the *hot spots*
- Faster *hot spot* finding

BLAST definitions

- Given two strings S_1 and S_2
- A *segment pair* is a pair of equal lengths substrings of S_1 and S_2 aligned without gaps
- A *locally maximal segment* is a segment whose alignment score (without gaps) cannot be improved by extending or shortening it.
- A *maximum segment pair* (*MSP*) in S_1 and S_2 is a segment pair with the maximum score over all segment pairs.

BLAST Process

- Parameters:
  - *w*: word length (substrings)
  - *t*: threshold for selecting interesting alignment scores

BLAST Process

- 1. Find all the *w*-length substrings from the database with an alignment score >*t*
  - Each of these (similar to a hot spot in FASTA) is called a *hit*
  - Does not have to be identical
  - Scored using substitution matrix and score compared to the threshold *t* (which determines number found)
  - Words size can therefore be longer without losing sensitivity: AA - 3-7 and DNA ~12

BLAST Process

- 2. Extend hits:
  - extend each hit to a local maximal segment
  - extension of initial w size hit may increase or decrease the score
  - terminate extension when a threshold is exceeded
  - find the best ones (HSP)
- This first version of Blast did not allow gaps….

(Improved) BLAST

Altshul, Madden, Schaffer, Zhang, Zhang, Miller & Lipman (1997) Gapped BLAST and PSI-BLAST:a new generation of protein database search programs. Nucleic Acids Research 25:3389-3402

- Improved algorithms allowing gaps
  - these have superceded the older version of BLAST
  - two versions: Gapped and PSI BLAST

(Improved) BLAST Process

- Find words or hot-spots
  - search each diagonal for two w length words such that score >=*t*
  - future expansion is restricted to just these initial words
  - we reduce the threshold t to allow more initial words to progress to the next stage

(Improved) BLAST Process

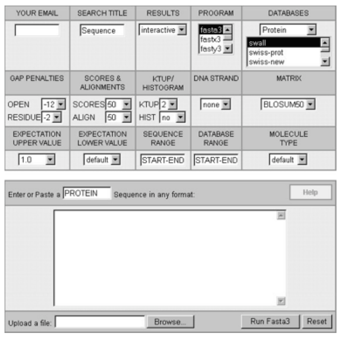
- Allow local alignments with gaps
  - allow the words to merge by introducing gaps
  - each new alignment comprises two words with a number of gaps
  - unlike FASTA does not restrict the search to a narrow band
  - as only two word hits are expanded this makes the new blast about 3x faster

PSI-BLAST

- Iterative version of BLAST for searching for protein domains
  - Uses a dynamic substitution matrix
  - Start with a normal blast
  - Take the results and use these to ‘tweak’ the matrix
  - Re-run the blast search until no new matches occur
- Good for finding distantly related sequences but high frequency of false-positive hits

BLAST Programs

- blastp compares an amino acid query sequence against a protein sequence database.
- blastn compares a nucleotide query sequence against a nucleotide sequence database.
- blastx compares a nucleotide query sequence translated in all reading frames against a protein sequence database.
- tblastn compares a protein query sequence against a nucleotide sequence database dynamically translated in all reading frames.
- tblastx compares the six-frame translations of a nucleotide query sequence against the six-frame translations of a nucleotide sequence database. (SLOW)



YOUR EMAIL	SEARCH TITLE	RESULTS	PROGRAM	DATABASES
	Sequence	interactive	fasta3 / fastx3 / fasty3	Protein; swall / swiss-prot / swiss-new
GAP PENALTIES	SCORES & ALIGNMENTS	KTUP/ HISTOGRAM	DNA STRAND	MATRIX
OPEN -12 RESIDUE -2	SCORES 50 ALIGN 50	KTUP 2 HIST no	none	BLOSUM50
EXPECTATION UPPER VALUE	EXPECTATION LOWER VALUE	SEQUENCE RANGE	DATABASE RANGE	MOLECULE TYPE
1.0	default	START-END	START-END	default

Enter or Paste a PROTEIN Sequence in any format:

Alignment Heuristics

- Dynamic Programming is better but too slow
- BLAST (and FASTA) based on several assumptions about good alignments
  - substitutions more likely than gaps
  - good alignments have runs of identical matches
- FASTA good for DNA sequences but slower
- BLAST better for amino acid sequences, pretty good for DNA, fastest, now dominant.

Biological Databases (sequences)

Biological Databases

- Introduction to Sequence Databases
- Overview of primary query tools and the databases they use (e.g. databases used by BLAST and FASTA)
- Demonstration of common queries
- Interpreting the results
- Overview of annotated 'meta' or 'curated' databases

DNA Sequence Databases

- Raw DNA (and RNA) sequence
- Submitted by Authors
- Patent, EST, Gemomic sequences
- Large degree of redundancy
- Little annotation
- Annotation and Sequence errors!

Main DNA DBs

• Genbank	US
• EMBL	EU
• DDBJ	Japan
• Celera genomics	Commercial DB

EMBL

- Sources for sequence include:
  - Direct submission - on-line submission tools
  - Genome sequencing projects
  - Scientific Literature - DB curators and editorial imposed submission
  - Patent applications
  - Other Genomic Databases, esp Genbank

International Nucleotide Sequence Database Collaboration

- Partners are EMBL, Genbank & DDBJ
- Each collects sequence from a variety of sources
- New additions to any of the three databases are shared to the others on a daily basis.

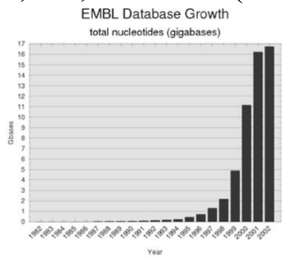
Limited annotation

- Unique accession number
- Submitting author(s)
- Brief annotation if available
- Source (cDNA, EST, genomic etc)
- Species
- Reference or Patent details

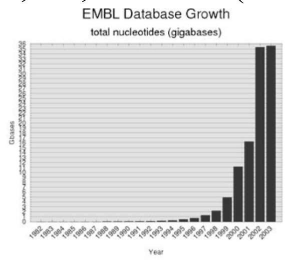
EMBL file tags

```
ID - identification              (begins each entry; 1 per entry)
AC - accession number            (>=1 per entry)
SV - new sequence identifier     (>=1 per entry)
DT - date                        (2 per entry)
DE - description                 (>=1 per entry)
KW - keyword                     (>=1 per entry)
OS - organism species            (>=1 per entry)
OC - organism classification     (>=1 per entry)
OG - organelle                   (0 or 1 per entry)
RN - reference number            (>=1 per entry)
RC - reference comment           (>=0 per entry)
RP - reference positions         (>=1 per entry)
RX - reference cross-reference   (>=0 per entry)
RA - reference author(s)         (>=1 per entry)
RT - reference title             (>=1 per entry)
RL - reference location          (>=1 per entry)
DR - database cross-reference    (>=0 per entry)
FH - feature table header        (0 or 2 per entry)
FT - feature table data          (>=0 per entry)
CC - comments or notes           (>=0 per entry)
XX - spacer line                 (many per entry)
SQ - sequence header             (1 per entry)
bb - (blanks) sequence data      (>=1 per entry)
// - termination line            (ends each entry; 1 per entry)
```

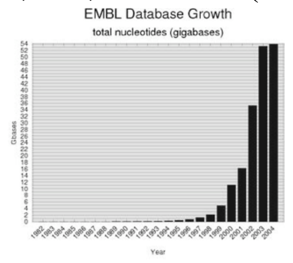
16,759,535,577 bases (27/1/02)



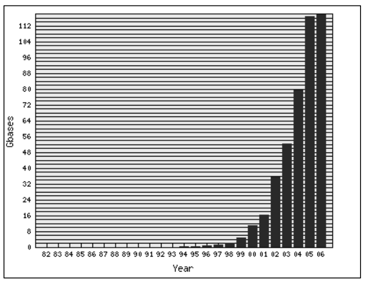
35,602,556,374 bases (17/1/03)



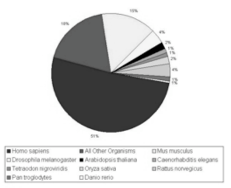
53,958,991,118 bases (24/1/04)



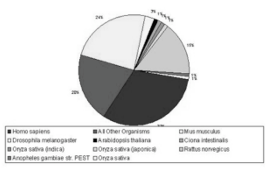
Jan '06 117,599,582,673bp



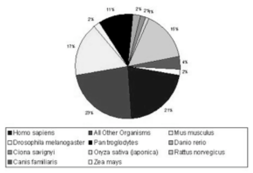
Bases by organism 02



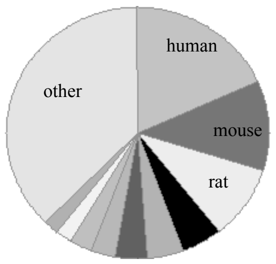
Bases by organism 03



Bases by organism 04



Bases by organism 06



http://www3.ebi.ac.uk/Services/DBStats/

17 Subdivisions

ESTs	EST
Bacteriophage	PHG
Fungi	FUN
Genome survey	GSS
High Throughput cDNA	HTC
High Throughput Genome	HTG
Human	HUM
Invertebrates	INV
Mus musculus	MUS
Organelles	ORG
Other Mammals	MAM
Other Vertebrates	VRT
Plants	PLN
Prokaryotes	PRO
Rodents	ROD
STSs	STS
Synthetic	SYN
Unclassified	UNC
Viruses	VRL

ESTs

- Expressed Sequence Tags
  - short mRNA samples from tissues
  - cloned and sequenced
  - single read
  - approx 1/3 of the database

HTG

- High throughput genomic sequences
  - Partial sequences obtained during genome sequencing.
  - Around 1/3 of the database

Specialist DNA Databases

- Usually focus on a single organism or small related group
- Much higher degree of annotation
- Linked more extensively to accessory data
  - Species specific:
    - Drosophila: FlyBase,
    - C. elegans: AceDB
  - Other examples include Mitochondrial DNA, Parasite Genome DB

FlyBase

flybase.bio.indiana.edu

- Includes the entire annotated genome searchable by BLAST or by text queries
- Also includes a detailed ontology or standard nomenclature for *Drosophila*
- Also provides information on all literature, researchers, mutations, genetic stocks and technical resources.
- Full mirror at EBI

Protein DBs

- Primary Sequence DBs
  - Swiss-Prot, TrEMBL, GenPept
- Protein Structure DBs
  - PDB, MSD
- Protein Domain Homology DBs
  - InterPro, CluSTr

UniProtKB/Swiss-Prot

- Consists of protein sequence entries
- Contains high-quality annotation
- Is non-redundant
- Cross-referenced to many other databases
- 104,559 sequences in Jan 02
- 120,960 sequences in Jan 03
- 194,317 sequences in Sep 05 (latest)

Swis-Prot by Species (‘03)

```
------  ---------  --------------------------------------------
Number  Frequency  Species
------  ---------  --------------------------------------------
     1       8950  Homo sapiens (Human)
     2~20%   6028  Mus musculus (Mouse)
     3       4891  Saccharomyces cerevisiae (Baker's yeast)
     4       4835  Escherichia coli
     5       3403  Rattus norvegicus (Rat)
     6       2385  Bacillus subtilis
     7       2286  Caenorhabditis elegans
     8       2106  Schizosaccharomyces pombe (Fission yeast)
     9       1836  Arabidopsis thaliana (Mouse-ear cress)
    10       1773  Haemophilus influenzae
    11       1730  Drosophila melanogaster (Fruit fly)
    12~13%   1528  Methanococcus jannaschii
    13       1471  Escherichia coli O157:H7
    14       1378  Bos taurus (Bovine)
    15       1370  Mycobacterium tuberculosis
```

Swis-Prot by Species (Oct ‘05)

```
------  ---------  --------------------------------------------
Number  Frequency  Species
------  ---------  --------------------------------------------
     1      12860  Homo sapiens (Human)
     2       9933  Mus musculus (Mouse)
     3       5139  Saccharomyces cerevisiae (Baker's yeast)
     4       4846  Escherichia coli
     5       4570  Rattus norvegicus (Rat)
     6       3609  Arabidopsis thaliana (Mouse-ear cress)
     7       2840  Schizosaccharomyces pombe (Fission yeast)
     8       2814  Bacillus subtilis
     9       2667  Caenorhabditis elegans
    10       2273  Drosophila melanogaster (Fruit fly)
    11       1782  Methanococcus jannaschii
    12       1772  Haemophilus influenzae
    13       1758  Escherichia coli O157:H7
    14       1653  Bos taurus (Bovine)
    15       1512  Salmonella typhimurium
```

UniProtKB/TrEMBL

- Computer annotated Protein DB
- Translations of all coding sequences in EMBL DNA Database
- Remove all sequences already in Swiss-Prot
- November 01: 636,825 peptides
- Jan 17th 2003: 728713 peptides
- TrEMBL new is a weekly update
- GenPept is the Genbank equivalent

SNPs

- Biggest growth area right now is in mutation databases
- www.ncbi.nlm.nih.gov/About/primer/snps.html
- Polymorphisms estimates at between 1:100 1:300 base pairs (normal human variation)
- Databases include true SNPs (single bases) and larger variations (microsatellites, small indels)

dbSNP

- “The database grows at 90 SNPs per month”
- 125 versions since start in 1998
- Currently 47 million SNPs in v125
- 15 million added between version 124 and 125

Database Search Methods

- Text based searching of annotations and related data: SRS, Entrez
- Sequence based searching: BLAST, FASTA, MPSearch

SRS

- Sequence Retrieval System
  - Powerful search of EMBL annotation
  - Linked to over 80 other data sources
  - Also includes results from automated searches

SRS data sources

- Primary Sequence: EMBL, SwissProt
- References/Literature: Medline
- Protein Homology: Prosite, Prints
- Sequence Related: Blocks, UTR, Taxonomy
- Transcription Factor: TFACTOR, TFSITE
- Search Results: BLAST, FASTA, CLUSTALW
- Protein Structure: PDB
- Also, Mutations, Pathways, other specialist DBs

Entrez

- Text based searching at NCBI’s Genbank
- Very simple and easy to use
- Not as flexible or extendable as SRS
- No user customisation

Sequence Based Searching

- Queries:

DNA query against DNA db
Translated DNA query against Protein db
Translated DNA query against translated DNA db
Translated Protein query against DNA db
Protein query against Protein db

- BLAST & FASTA

Secondary Databases

- PDB
- Pfam
- PRINTS
- PROSITE
- ProDom
- SMART
- TIGRFAMs

PDB

- Molecular Structure Database (EBI)
- Contains the 3D structure coordinates of ‘solved’ protein sequences
  - X-ray crystallography
  - NMR spectra
- 19749 protein structures

Armstrong, 2007 Bioinformatics 2

Multiple Sequence Alignment

- What and Why?
- Dynamic Programming Methods
- Heuristic Methods
- A further look at Protein Domains

Armstrong, 2007 Bioinformatics 2

Multiple Alignment

- Normally applied to proteins
- Can be used for DNA sequences
- Finds the common alignment of >2 sequences.
- Suggests a common evolutionary source between related sequences based on similarity
  - Can be used to identify sequencing errors

Armstrong, 2007 Bioinformatics 2

Multiple Alignment of DNA

- Take multiple sequencing runs
- Find overlaps
  - variation of ends-free alignment
- Locate cloning or sequencing errors
- Derive a consensus sequence
- Derive a confidence degree per base

Armstrong, 2007 Bioinformatics 2

Consensus Sequences

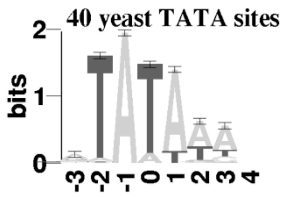
- Look at several aligned sequences and derive the most common base for each position.
  - Several ways of representing consensus sequences
  - Many consensus sequences fail to represent the variability at each base position.
  - Largely replaced by Sequence Logos but the term is often mis-applied

Armstrong, 2007 Bioinformatics 2

Sequence Logos

- Example, from an alignment of the TATA box in yeast genes:

We now have a confidence level for each base at each position



Armstrong, 2007 Bioinformatics 2

Multiple Alignment of Proteins

- Multiple Alignment of Proteins
- Identify Protein Families
- Find conserved Protein Domains
- Predict evolutionary precursor sequences
- Predict evolutionary trees

Protein Families

- Proteins are complex structures built from functional and structural sub-units
  - When studying protein families it is evident that some regions are more heavily conserved than others.
  - These regions are generally important for the structure or function of the protein
  - Multiple alignment can be used to find these regions
  - These regions can form a signature to be used in identifying the protein family or functional

Protein Domains

- Evolution conserves sequence patterns due to functional and structural constraints.
- Different methods have been applied to the analysis of these regions.
- Domains also known by a range of other names:

motifs patterns prints blocks

Multiple Alignment

- OK we now have an idea WHY we want to try and do this
- What does a multiple alignment look like?
- How could we do multiple alignments
- What are the practical implications

Multiple alignment table

```
dlg_CG1725-PH           ALFDYDPNRDDGLPSRGLPFKH
Sap97_dlgh1             ALFDYDKTKDSGLPSQGLNFRF
chapsyn-110_dlgh2       AMFDYDKSKDSGLPSQGLSFKY
Sap102_dlgh3            ALFDYDRTRDSCLPSQGLSFSY
PSD-95_dlgh4            ALFDYDKTKDCGFLSQALSFHF
                        *:**** .:*  : *:.* * .
```

A consensus character is the one that minimises the distance between it and all the other characters in the column

Conservatived or Identical residues are colour coded

Scoring Multiple Alignments

- We need to score on columns with more than 2 bases or residues:

$$\textit{ColumnCost}\begin{pmatrix} S \\ C \\ A \\ P \\ P \end{pmatrix} = 24$$

Multiple alignments are usually scored on cost/difference rather than similarity

Column Costs

- Several strategies exist for calculating the column cost in a multiple alignment
- Simplest is to sum the pairwise **costs** of each base/residue pair in the column using a matrix (e.g. PAM250).
- Gap scoring rules can be applied to these as well.

Scoring Multiple Alignments

- Score = (S,C)+(S,A)+(S,A)+(S,P)+(S,P)+ (C,A)+ (C,P)+(C,P)+(A,P)+(A,P)+(P,P)

$$\mathit{ColumnCost}\begin{pmatrix} S \\ C \\ A \\ P \\ P \end{pmatrix} = 24$$

Known as the sum-of-pairs scoring method

Sum-of-pairs cost method (SP)

- Score = (S,C)+(S,-)+(S,A)+(S,P)+(S,P)+ (-,A)+(-,P)+(-,P)+(A,P)+(A,P)+(P,P)

$$\mathit{ColumnCost}\begin{pmatrix} S \\ - \\ A \\ P \\ P \end{pmatrix} = 24$$

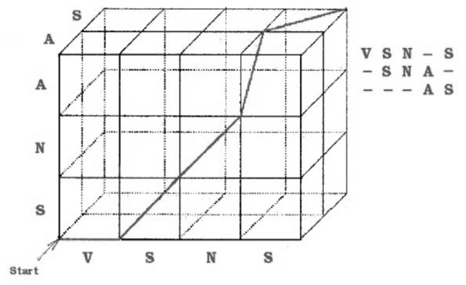
Still works with gaps using whatever gap penalty you want

Multiple Alignment Cost

- Sum of pairs is a simple method to get a score for each column in a multiple alignment
- Based on matrices and gap penalties used for pairwise sequence alignment
- The score of the alignment is the sum of each column

Optimal Multiple Alignment

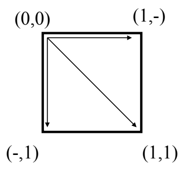
- The best alignment is generally the one with the lowest score (i.e. least difference)
  - depends on the scoring rules used.
- Like pairwise cases, each alignment represents a path through a matrix
- For multiple alignment, the matrix is *n*-dimensional
  - where *n*=number of sequences



(Murata, Richardson and Sussman 1999)

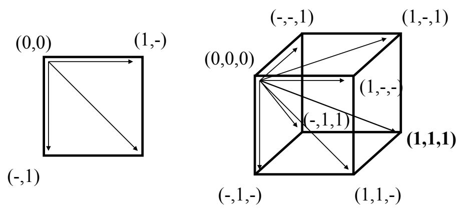
Contrasting pairwise and multiple alignments

Lets compare pairwise with three sequences.



Contrasting pairwise and multiple alignments

Lets compare pairwise with three sequences.



Multiple alignment table

```
dlg_CG1725-PH          ALFDYDPNRDDGLPSRGLPFKH
Sap97_dlgh1            ALFDYDKTKDSGLPSQGLNFRF
chapsyn-110_dlgh2      AMFDYDKSKDSGLPSQGLSFKY
Sap102_dlgh3           ALFDYDRTRDSCLPSQGLSFSY
PSD-95_dlgh4           ALFDYDKTKDCGFLSQALSFHF
                       *:**** .:*  : *:.* * .
```

The consensus character is the one that minimises the distance between it and all the other characters in the column