

We were talking about
similarity,
sequence comparison
and alignment.

HOW DOES IT WORK ?

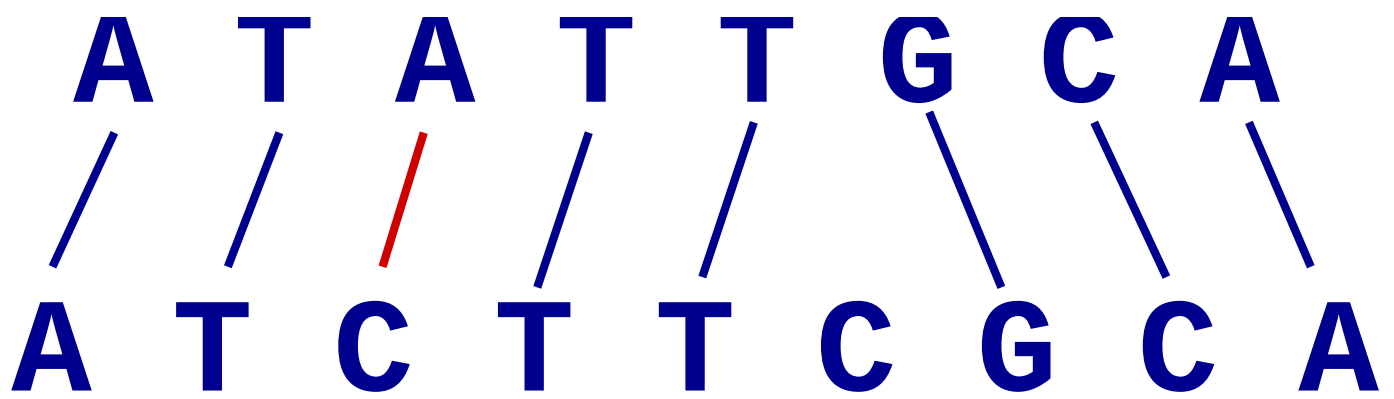
The high end solution

Use the most sensible,
most powerful,
and best trainable tool available ...

. . . your eyes

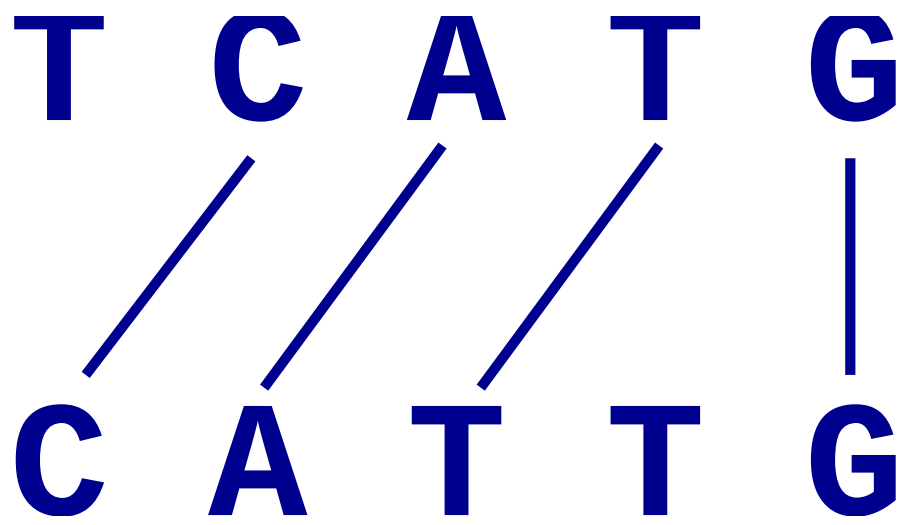
A T A T T G C A

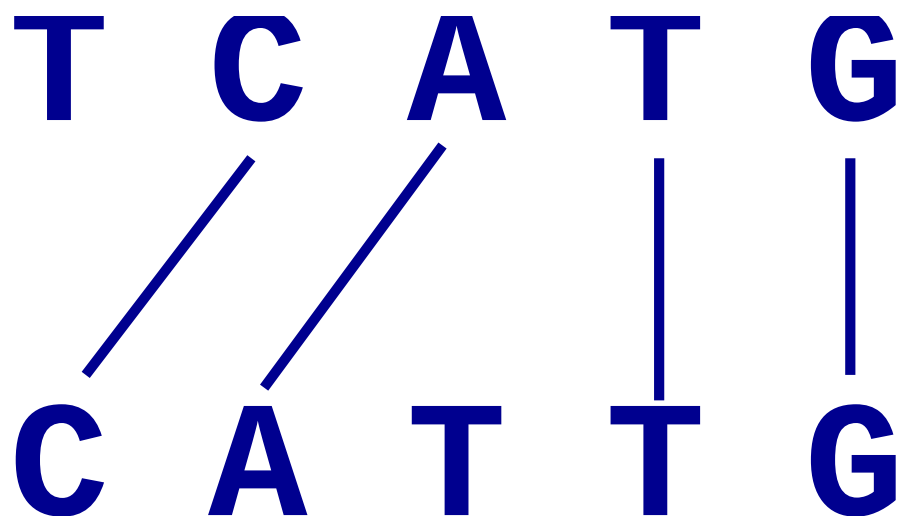
A T C T T C G C A



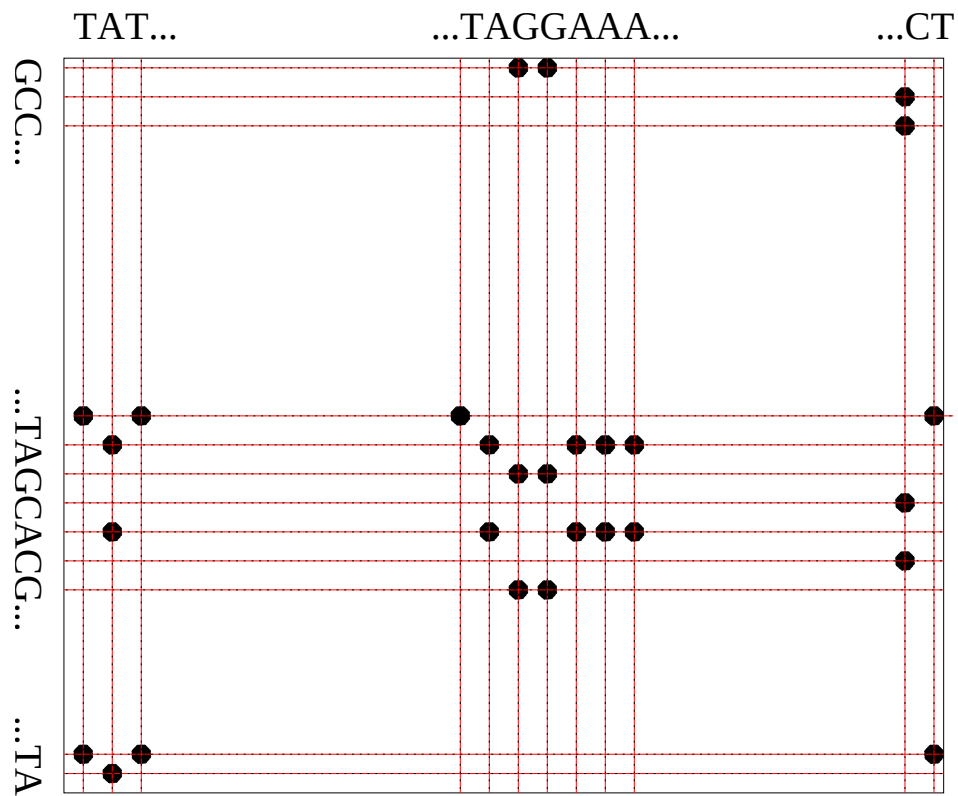
T C A T G

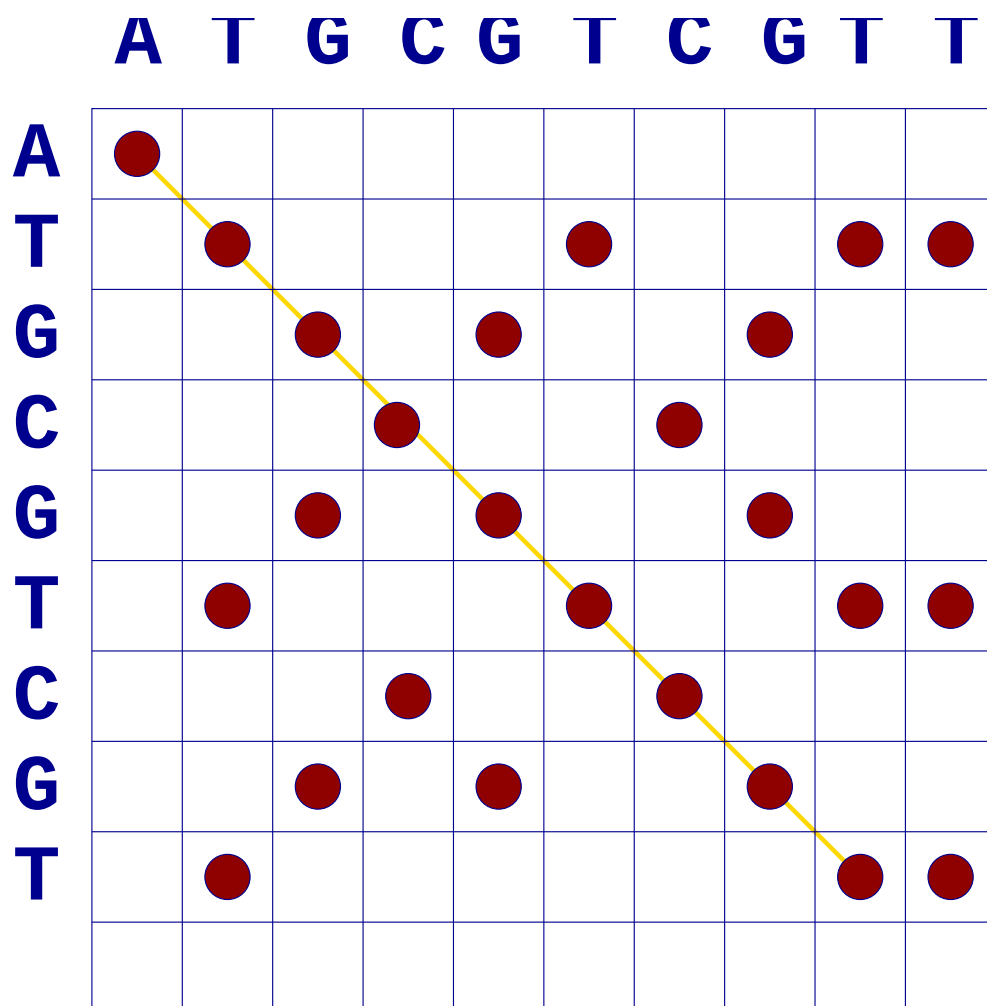
C A T T G

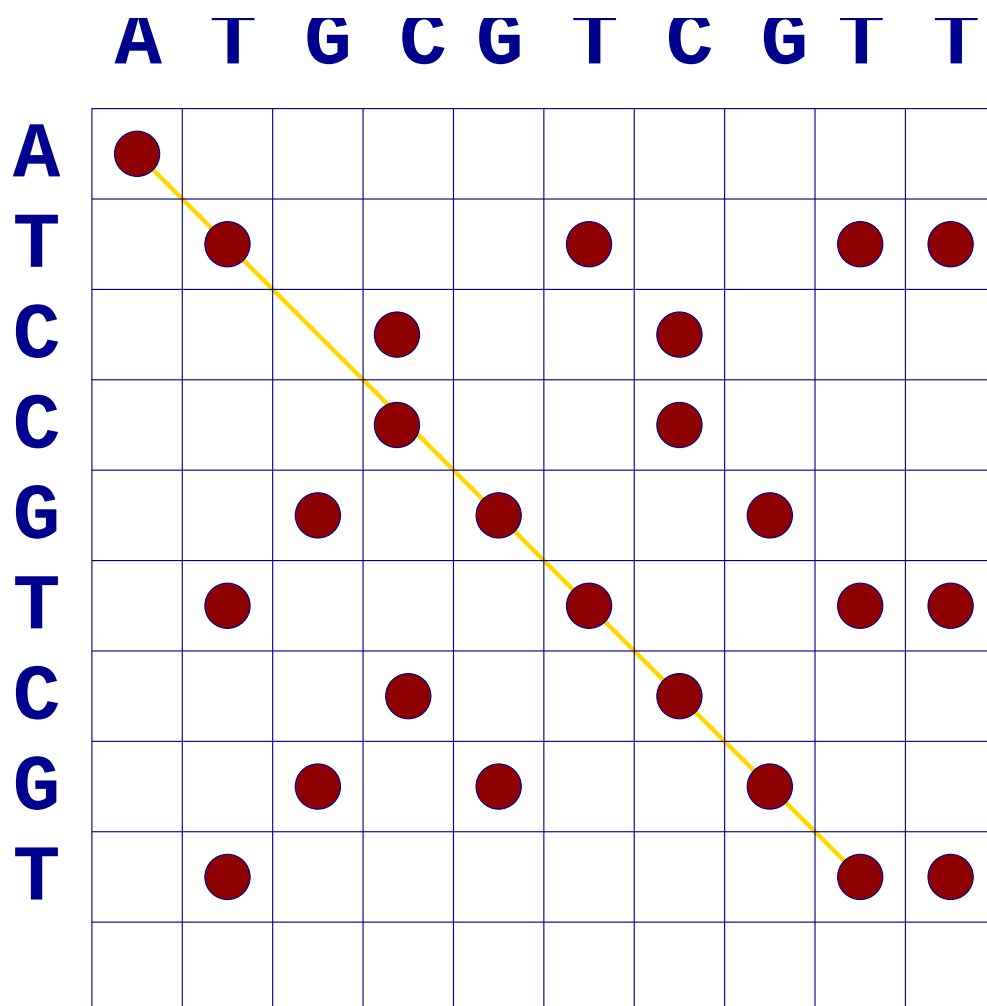


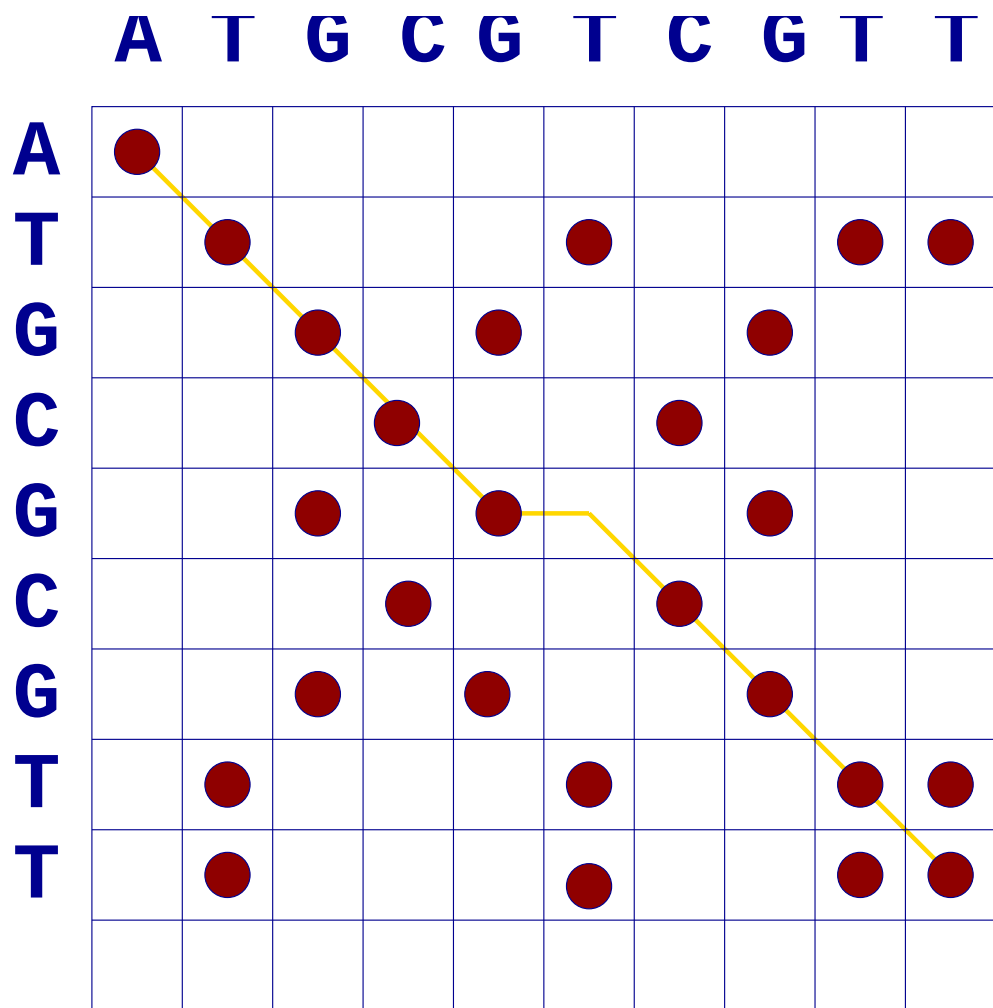


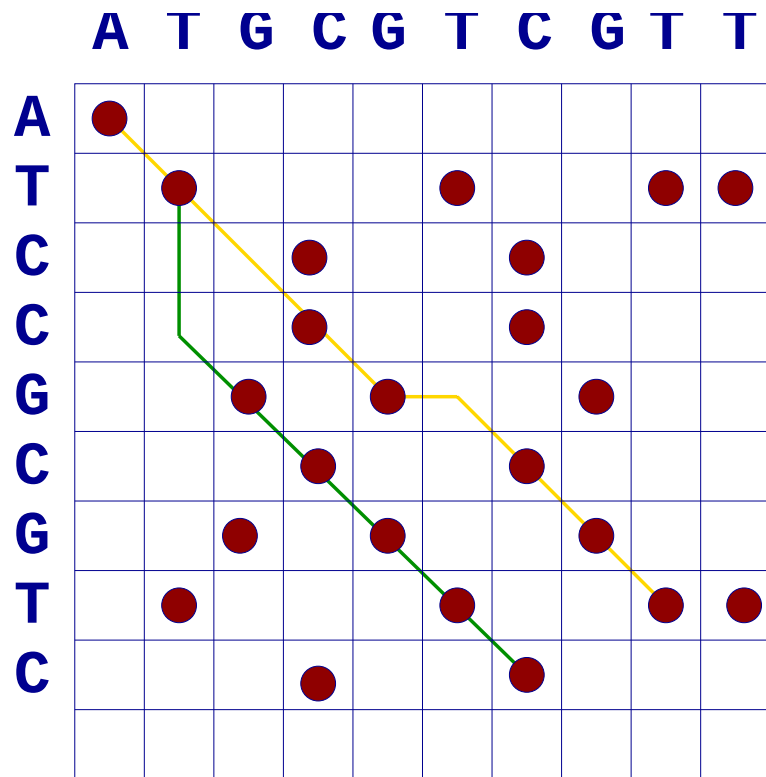
DOTPLOTS







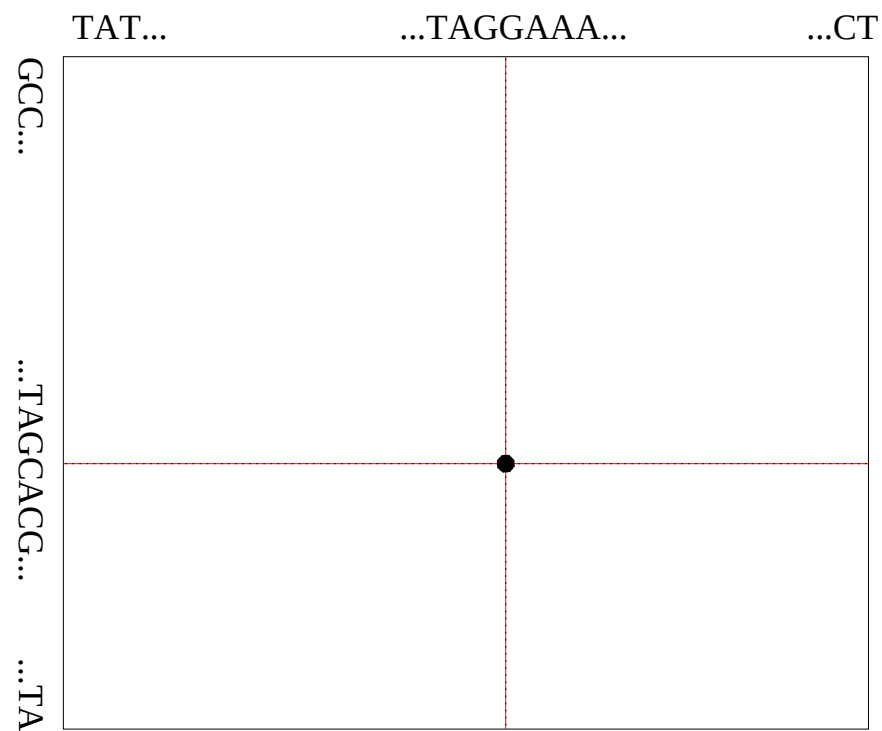


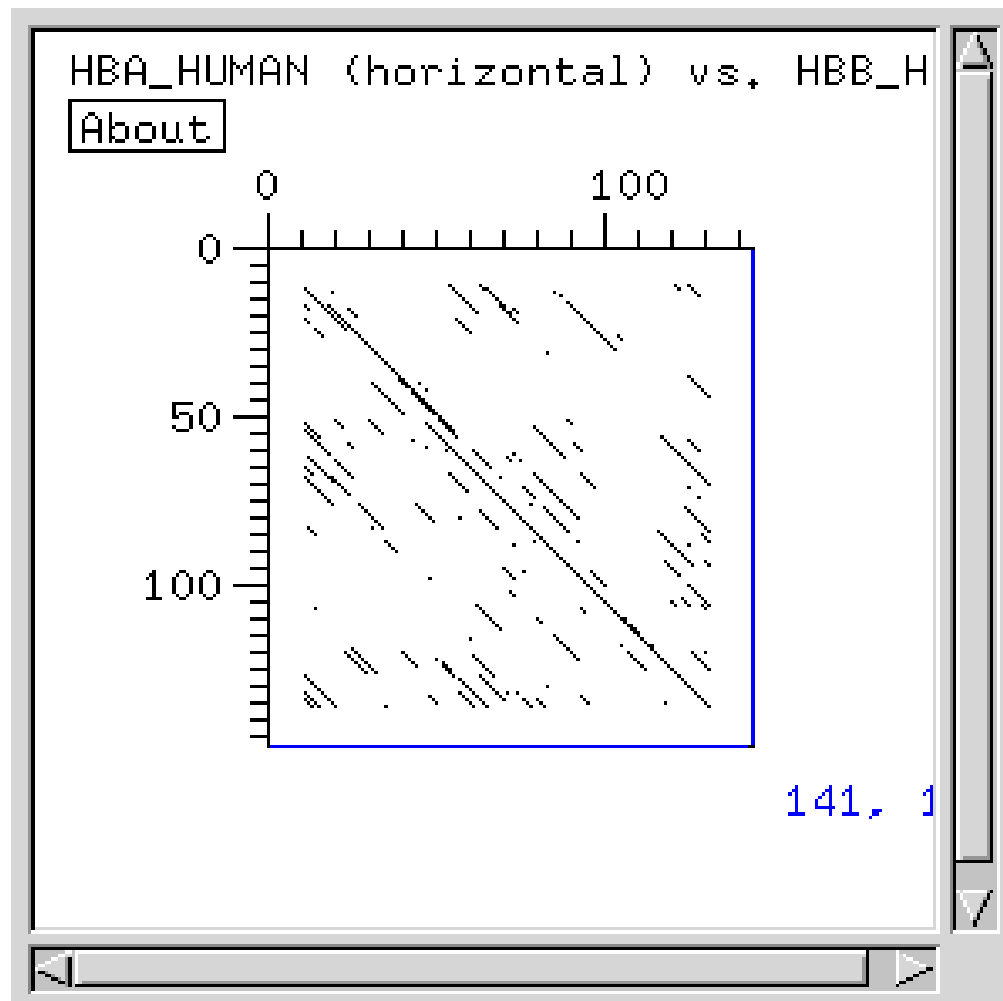


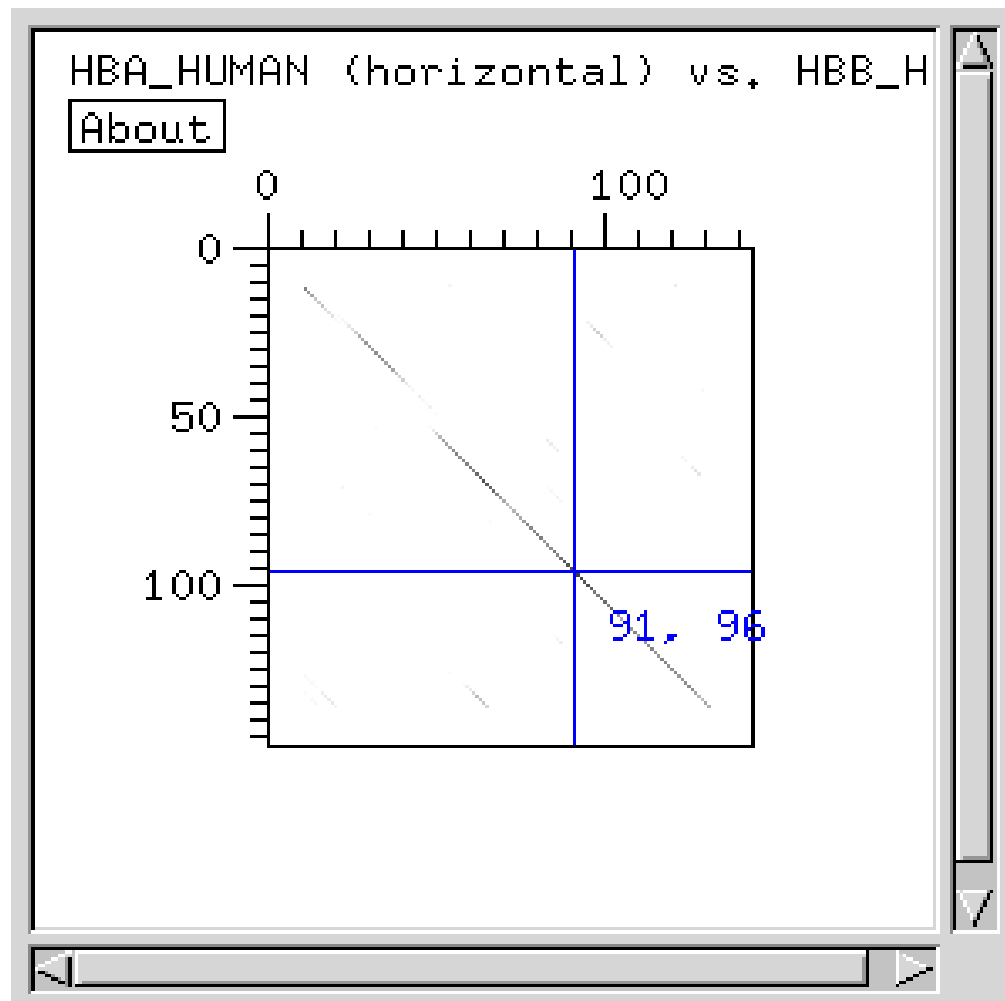
AT--GCGTCGTT
ATCCGCGTC---

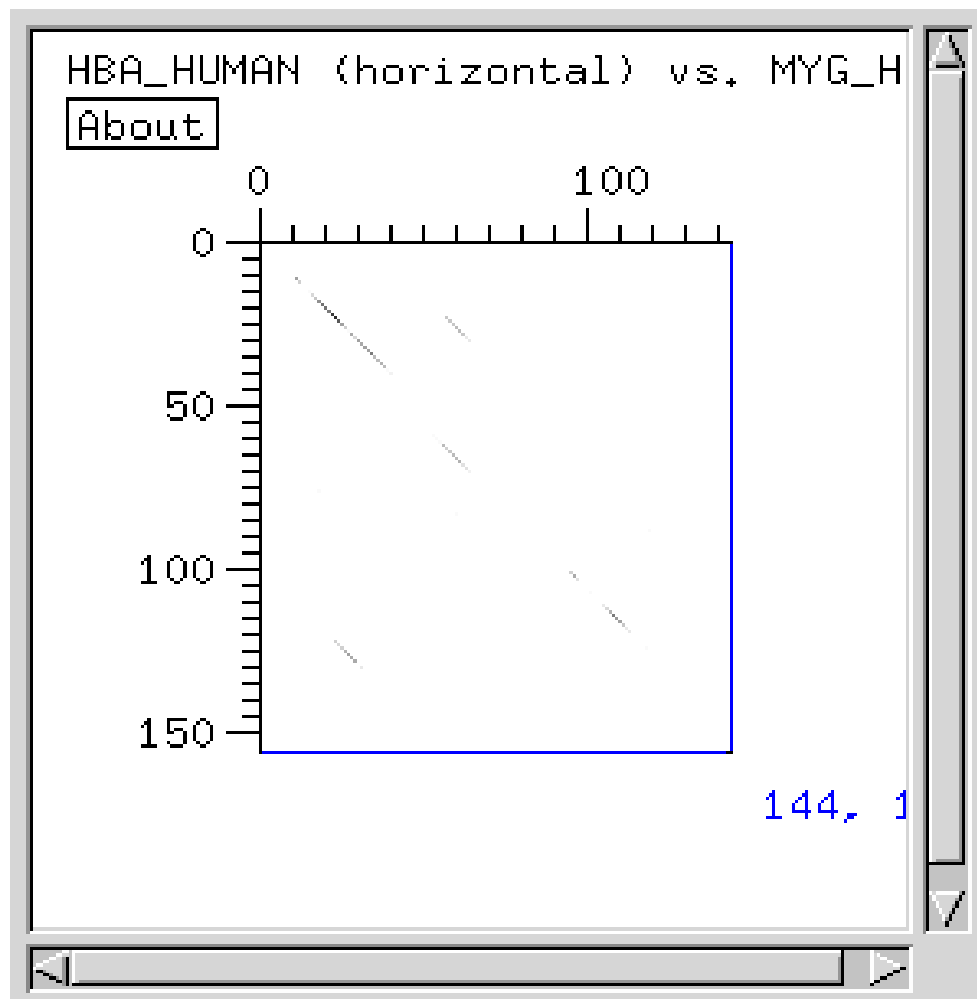
ATGCGTCGTT
ATCG-CGTC

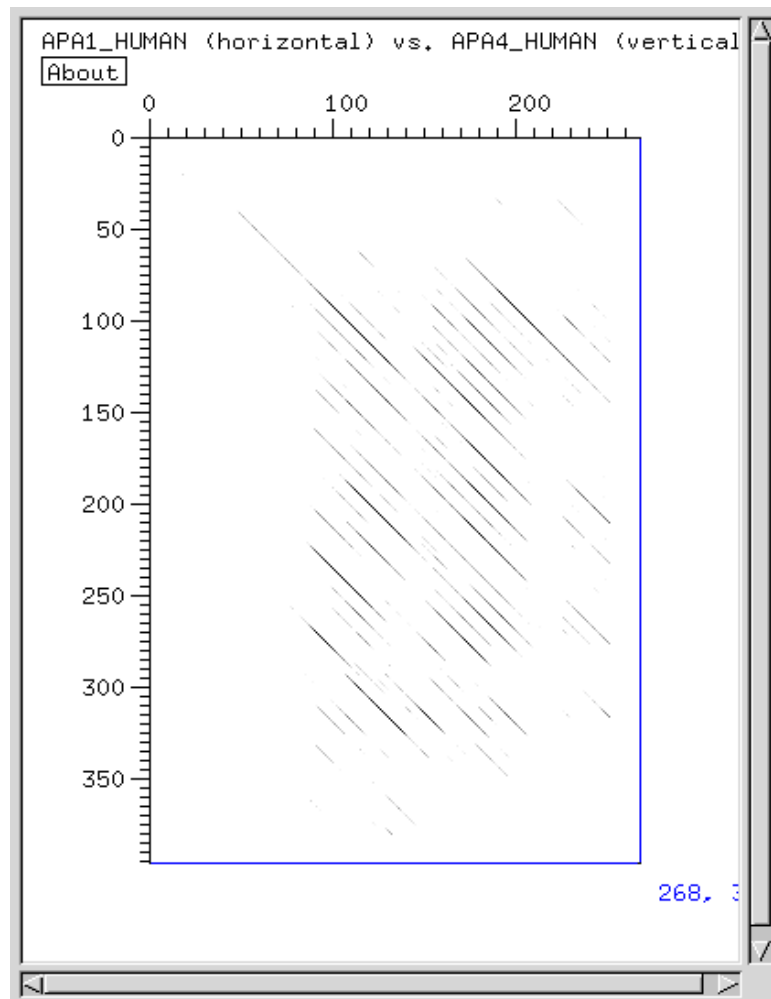
TATAGCGTCATGCGTACCCCCCTAGGAAAGGATCAGCCCTATATCT
 GCCTAAACCACTGTGTCTCTTTAGCACGGGGTATCCATA

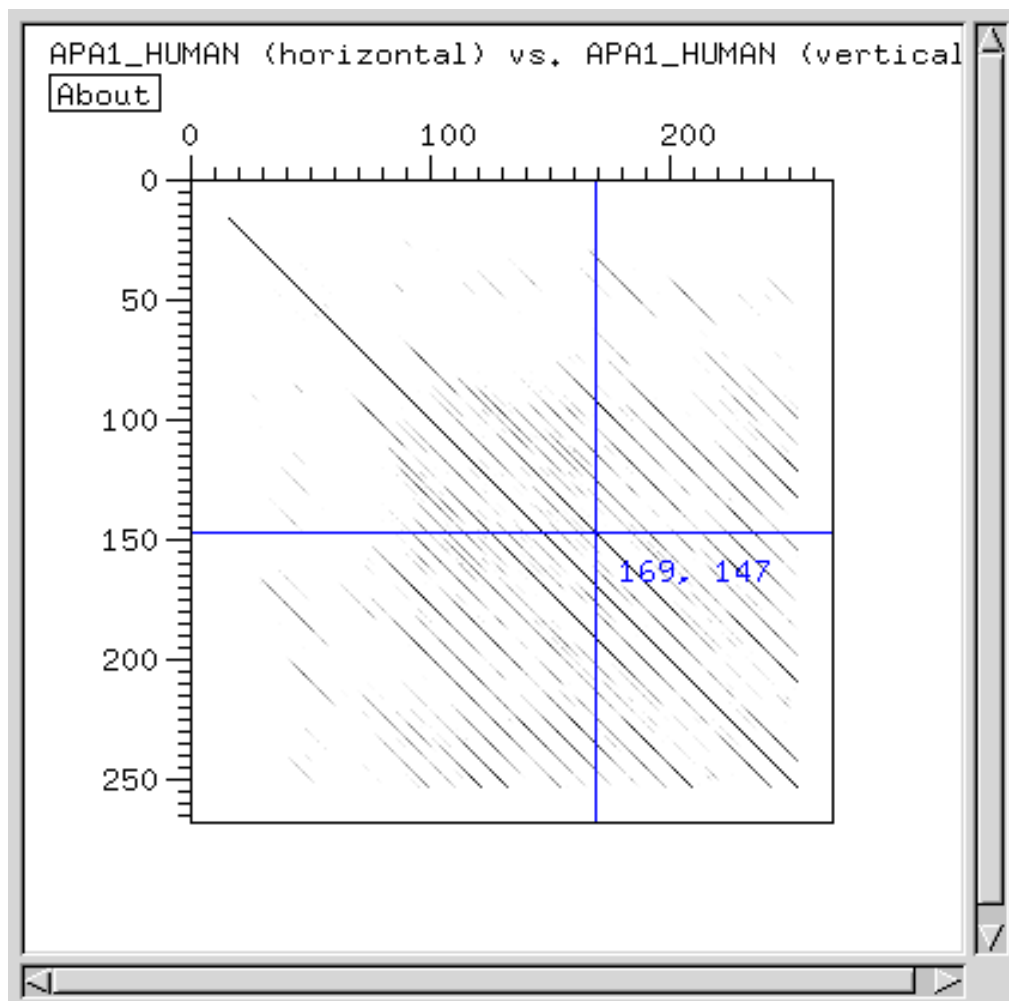












Dotplots ...

- ... detect both global and local similarity**
- ... detect internal repeats**
- ... detect multiple domain structure**

Dotplots ...

... rely on the power of human cognition

... are qualitative and not quantitative

Definition (global alignment)

A global alignment between two sequences S_1 and S_2 is obtained by first inserting chosen spaces, either into or at the end of S_1 and S_2 , and then placing the resulting strings one above the other so that every space or character in either sequence is opposite a unique character or a unique space in the other string. Matching spaces are not allowed

Editing

Given two sequences:
Edit the first sequence such that
it is identical to the second.

Edit operations:

	SHORT
(1) Replacements: R(A→T)	R
(2) Deletions: D(A)	D
(3) Insertions: I(T)	I
(4) Do Nothing: N	N

Only the first sequence is edited !!!

Example

Edit Script:

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

1: N
2: R(T→C)
3: N
4: D(G)
5: N
6: N
7: N
8: I(T)
9: N
10: N

**Given the first sequence
and an Edit script
we can reconstruct the second
sequence and the alignment.**

**First Sequence: C C A T
Script: N D(C) N I(T) N**

Alignment:

**C C A - T
C - A T T**

Given both sequences
and the short version of an edit script
we can reconstruct the alignment

First Sequence: C C A T
Second Sequence: C A T T



Short Script: N D N I N

C C A - T
C - A T T

Every alignment is equivalent to a
string on the alphabet {R D I N}

Definition (Edit Distance)

The edit distance between two sequences is the minimum number of edit operations {R I D} needed to transform the first sequence into the second.

Note that {N} operations are not counted

In order to calculate the edit distance of two sequences we need to **solve an optimization problem:**

Given two sequences: What is the shortest edit script that transforms the first sequence into the second.

The length of the script is the number of {R I D} in it.

Let S_1 be a sequence of length n_1 and
let S_2 be a sequence of length n_2 .

There are at least $\binom{n_1+n_2}{n_1}$

different global alignments
between S_1 and S_2

A PROOF IN RED AND BLUE

C A A G T - -
C A - T G C A

C C A A A G T T G C A
R B R B R R B R B B B

There are $\binom{n_1+n_2}{n_1}$ ways
to place the n_1 blue Bs in this
string of length n_1+n_2

A T A T T G C A
A T C T T C G C A

24310 different alignments

Two sequences of length 500:
2.7029e+299 different alignments

Divide and conquer

Subdivide a problem that is too large to be computed, into smaller problems that may be efficiently computed. Then assemble the answers to give a solution to the large problem.

Dynamic Programming

Recursively subdivide a large problem into subproblems of the same type.

Subproblems should share subproblems.

Calculate the solution of **all** the subproblems just **once**.

Save the answer in a **table**, thereby avoiding the work of recomputing the answers everytime the subproblem is encountered.

The 3 Steps of a dynamic programming algorithm.

- (1) The recurrence relation**
- (2) A tabular computation scheme**
- (3) The traceback**

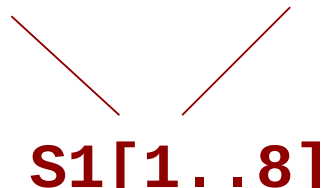
Notation:

Let $S1$ and $S2$ be two sequences.

$S1[1..i]$ and $S2[1..j]$ are the first i resp. j characters of the sequences.

$D(i, j)$ denotes the edit distance of $S1[1..i]$ and $S2[1..j]$

S1: TAGGTCAT CCATATAATA



S1[1..8]

Problem:

Calculate the minimal edit distance of 2 sequences and the corresponding global alignment.

Observation:

That is easier for short sequences.

Strategy:

Solve the problem for all $S1[1..i]$ and $S2[1..j]$.



shorter sequences

An alignment ends either with
(1) a match/mismatch
(2) a gap in the first sequence
(3) a gap in the second sequence

S1: ATCGCTGGCATAC
S2: TTCCTAGCCTAC

ATCGCT
TTCCTA



use the opt.
alignment of
S1[1..5] and
S2[1..5].

ATCGCT-
-TTCCTA



use the opt.
alignment of
S1[1..6] and
S2[1..5].

ATCGC-T
TTCCTA-



use the opt.
alignment of
S1[1..5] and
S2[1..6].

One of the alignments is optimal !

The recurrence relation

	ATCGCT TTCCTA	ATCGCT- -TTCCTA	ATCGC-T TTCCTA-
Edit steps	$D(5, 5) + 1$	$D(6, 5) + 1$	$D(5, 6) + 1$

$$D(6, 6) = \min \begin{bmatrix} D(5, 5) + 1 \\ D(6, 5) + 1 \\ D(5, 6) + 1 \end{bmatrix}$$

The general recurrence relation

$$D(i, j) = \min \begin{cases} D(i-1, j-1) + t(i, j) \\ D(i, j-1) + 1 \\ D(i-1, j) + 1 \end{cases}$$

$t(i, j) = 0$ if $S1(i) = S2(j)$ "match"
 $t(i, j) = 1$ if $S1(i) \neq S2(j)$ "mismatch"

"Calculate $D(3,4)$ " is a subproblem of
"calculate $D(5,5)$ "

"Calculate $D(3,4)$ " is also a subproblem
of "calculate $D(12,15)$ "

Idea:

We solve "calculate $D(3,4)$ " only once

We start with solving easy problems
like "calculate $D(1,1)$ " or even

"calculate $D(0,0), D(0,1), D(1,0) \dots$ "

BOTTOM-UP COMPUTATION

INITIALIZATION

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
V	1	1							
I	2	2							
N	3	3							
T	4	4							
N	5	5							
E	6	6							
R	7	7							

Align the first 0
characters of S1
to the first 2
characters of S2:

S1: **WRITERS**

S2: **VINTERS**

VI...

--...

This results in
2 insertions.

Tabular calculation

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
V	1	1	1	2	3	4	5	6	7
I	2	2	2	2	2	3	4	5	6
N	3	3	3	3	3	3	4	5	6
T	4	4	4	4	4	?			
N	5	5							
E	6	6							
R	7	7							

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
V	1	1	1	2	3	4	5	6	7
I	2	2	2	2	2	3	4	5	6
N	3	3	3	3	3	3	4	5	6
T	4	4	4	4	4	3	4	5	6
N	5	5	5	5	5	4	4	5	6
E	6	6	6	6	6	5	4	5	6
R	7	7	7	6	7	6	5	4	5


 Edit distance
of S1 and S2

THE TRACEBACK

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
V	1	↑1	↑1	↖2	↖3	↖4	↖5	↖6	↖7
I	2	↑2	↑2	↖2	↖2	↖3	↖4	↖5	↖6
N	3	↑3	↑3	↑3	↖3	↖3	↖4	↖5	↖6
T	4	↑4	↑4	↑4	↖4	↖3	↖4	↖5	↖6
N	5	↑5	↑5	↑5	↖5	↖4	↖4	↖5	↖6
E	6	↑6	↑6	↑6	↖6	↖5	↖4	↖5	↖6
R	7	↑7	↑7	↖6	↖7	↖6	↖5	↖4	↖5

RETRIEVING COOPTIMAL ALIGNMENTS

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
V	1	1	1	2	3	4	5	6	7
I	2	2	2	2	2	3	4	5	6
N	3	3	3	3	3	3	4	5	6
T	4	4	4	4	4	3	4	5	6
N	5	5	5	5	5	4	4	5	6
E	6	6	6	6	6	5	4	5	6
R	7	7	7	6	7	6	5	4	5

WRIT-ERS
VINTNER-
*** * *

WRI-T-ERS
V-INTNER-
** * * *

WRI-T-ERS
-VINTNER-
** * * *