

Bioinformatics: Life Sciences on Your Computer

BLAST

This is supplemental reading to the [YouTube video](#) on BLAST.

Speed, sensitivity, specificity

Most algorithms attempt to maximize three measures. It is hard to maximize all three, because tradeoffs usually come into play.

Sensitivity

Ability to find all positives

$100\% - x\% = \text{accuracy}$

$x\% = \% \text{ actual hits missed by algorithm}$

Specificity

Ability to avoid false positives

$100\% - y\%$ where $y\% = \text{false positives}$

Speed

How fast the program produces results. Often speed comes at the expense of sensitivity.

Discussion topic: Let's say you start with a query protein. In a database, because I am all knowing (in this case), there are exactly 80 homologs. You run a database search and find 55 hits. Of the 55 hits, 50 are really homologs. Calculate the sensitivity and specificity. Okay, now not being satisfied with your results, let's say we lower the bar and accept lower scored matches as positive hits. Now we have 75 hits, but only 63 are homologs. Now calculate the sensitivity and specificity. Do you see the tradeoff?

Local alignment

Most searches use local alignment, which helps find matches to protein domains. Domain matches do not always align for the whole length of the protein. The rigorous Smith-Waterman algorithm is not practical for database searches, due to too much computation for searching a large database. Heuristic modifications of Smith-Waterman are required.

Increasing the word length in BLAST allows for a faster program. That does sacrifice sensitivity.

BLAST

Stephen Altschul developed BLAST in 1990 at NCBI. BLAST was faster than FASTA, which had been the best search algorithm. BLAST seeks high-scoring segment pairs (HSPs), which is a pair of matching sequences aligned with no gaps.

BLAST uses longer word length than FASTA. A longer word length usually increases speed but sacrifices sensitivity. In order to regain that lost sensitivity, BLAST allows for "misspellings." The algorithm uses "neighborhood words," measured by BLOSUM 62. Any three letter word scoring higher than a predetermined threshold is considered a match.

Phase 1: compile a list of words (w = 3) above threshold T

- Query sequence: human beta globin NP_000509 (includes ... VTALWGKVNVD...)

- words derived from query sequence (HBB):

VTALWGKVNVD

- generate a list of words matching query

(both above and below T). Consider **LWG** in the query and the scores (derived from a BLOSUM62 matrix) for various words:

	LWG	4+11+6=21
	IWG	2+11+6=19
	MWG	2+11+6=19
	VWG	1+11+6=18
examples of	FWG	0+11+6=17
words above	AWG	0+11+6=17
threshold 11	LWS	4+11+0=15
	LWN	4+11+0=15
	LWA	4+11+0=15
	LYG	4+ 2+6=12
	LFG	4+ 1+6=11
examples of	FWS	0+11+0=11
words below	AWS	-1+11+0=10
threshold 11	CWS	-1+11+0=10
	IWC	2+11-3=10

threshold

BLAST extensions

BLAST extends matching words both directions, keeping track of the alignment score. An extension will continue until the score drops below a threshold of 22 for proteins, 20 for DNA. It stops at a high-scoring segment pair (HSP), and if the score is high enough, that would trigger a gapped extension.

Phase 3: extend the hits in either direction. Stop when the score drops.

LTPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKV HBB
 L+P +K+ V A WGKV + E G EAL R+ + +P T+ +F F D G+ +V
 LSPADKTNVKAAGKVGAGHAGEYGAEALERMFLSFPTTKTYFPHF-----DLSHGSAQV HBA

extension word pair from extension
 first phases of search
 "hits" alpha globin,
 triggers extension

A major modification happened in 1997. Hits were extended in either direction in the original (1990) implementation of BLAST. In the refinement, two independent hits required, and the hits must occur near each other. With the modification only one seventh as many extensions occur, greatly reducing the time required for a search.

Large genomic sequences

Any search algorithm takes a very long time to run if the query sequence is very long. This most commonly occurs with genomic DNA sequences. There are two specialized forms of BLAST that are used to help cut down the computation time. **MegaBLAST** uses a very high word length. While basic blastn uses a word length of 11, MegaBLAST uses a word length of 28. It's good for finding VERY similar and nearly identical sequences and that can be useful in the right situation.

Because MegaBLAST sacrifices sensitivity with a word length of 28, a version was developed to help

find sequences that are slightly divergent from the query. **Discontinuous MegaBLAST** allows for gaps and small changes. It runs a little slower than MegaBLAST but is a bit more sensitive.

Be careful when using nucleotide BLAST. Currently, the default is MegaBLAST, not blastn. To run blastn, click the radiobutton for blastn.