

Building an Automated Scientist: Using Machine Learning to Configure Algorithms

Modern science is computational

Modern science is increasingly **computational**.

- Particularly in genomics, where experiments have multiple computational steps.
- Domain problems have in turn lead to algorithmic advances.

More people are **relying** on computational tools.

Parameter Advising for Bioinformatics

Bioinformatics software

Common themes arise in bioinformatics (and many other domain) problems.

- Many are **computationally inefficient** to solve exactly.
- **Many tools** developed for these problems.
- Each tool has many **parameters** whose values have an impact on the output.

Tunable parameters

Quant

=====


Perform dual-phase, mapping-based estimation of transcript abundance from RNA-seq reads

salmon quant options:

basic options:

<code>-v [--version]</code>	print version string
<code>-h [--help]</code>	produce help message
<code>-i [--index] arg</code>	Salmon index
<code>-l [--libType] arg</code>	Format string describing the library type
<code>-r [--unmatedReads] arg</code>	List of files containing unmated reads of (e.g. single-end reads)
<code>-1 [--mates1] arg</code>	File containing the #1 mates
<code>-2 [--mates2] arg</code>	File containing the #2 mates
<code>-o [--output] arg</code>	Output quantification file.
<code>--discardOrphansQuasi</code>	[Quasi-mapping mode only] : Discard orphan mappings in quasi-mapping mode. If this flag is passed then only paired mappings will be considered toward quantification estimates. The default behavior is to consider orphan mappings if no valid paired mappings exist. This flag is independent of the option to write the orphaned mappings to file (<code>--writeOrphanLinks</code>).
<code>--allowOrphansFMD</code>	[FMD-mapping mode only] : Consider orphaned reads as valid hits when performing lightweight-alignment. This option will increase sensitivity (allow more reads to map and more transcripts to be detected), but may decrease specificity as orphaned alignments are more likely to be spurious.
<code>--seqBias</code>	Perform sequence-specific bias correction.
<code>--gcBias</code>	[beta for single-end reads] Perform fragment GC bias correction
<code>-p [--threads] arg</code>	The number of threads to use concurrently.
<code>--incompatPrior arg</code>	This option sets the prior probability that an alignment that disagrees with the specified library type (<code>--libType</code>) results from the true fragment origin. Setting this to 0 specifies that alignments that disagree with the library type should be "impossible", while setting it to 1 says that alignments that disagree with the library type are no less likely than those that do
<code>-g [--geneMap] arg</code>	File containing a mapping of transcripts to genes. If this file is provided Salmon will output both <code>quant.sf</code> and <code>quant.genes.sf</code> files, where the latter contains aggregated gene-level abundance estimates. The transcript to gene mapping should be provided as either a GTF file, or a in a simple tab-delimited format where each line contains the name of a transcript and the gene to which it belongs separated by a tab. The extension of the file is used to determine how the file should be parsed. Files ending in <code>'.gtf'</code> , <code>'.gff'</code> or <code>'.gff3'</code> are assumed to be in GTF format; files with any other extension are assumed to be in the simple format. In GTF / GFF format, the <code>"transcript_id"</code> is assumed to contain the transcript identifier and the <code>"gene_id"</code> is assumed to contain the corresponding gene identifier.
<code>-z [--writeMappings] [=arg(=)]</code>	If this option is provided, then the quasi-mapping results will be written out in SAM-compatible format. By default, output will be directed to stdout, but an alternative file name can be provided instead.
<code>--meta</code>	If you're using Salmon on a metagenomic dataset consider setting this flag to disable parts of the abundance estimation model

Tunable parameters



Community Log In Sign Up

Question: Salmon libtype - Does it matter?


Hi,

I was wondering what happens if you have the libtype wrong for Salmon? Or why it requires wasn't sure if the first read of my paired-end reads was forward (ISF) or reverse (ISR), so I ran the mapping efficiencies were identical between the runs.

Thanks, Matt

salmon alignment

ADD COMMENT link




Welcome to Biostar! Community Log In Sign Up

Question: (Closed) Questions about Salmon

Hello All! I had some questions involving alignment based v alignment free mapping done by Salmon as well as some other general questions pertaining specifically to our experiments. Please excuse any perceived ignorance as I'm more of a molecular than computation biologist and

1. If one will probably map the reads in order to visualize alignment based mapping so that Salmon agrees with
2. Our experiments involve looking at ribosome profiling read counts. We also have the corresponding RNA-seq one/both data sets? My idea was to not use either of ribosome profiling data set.
3. Could someone provide a better explanation for finding changing the default settings be more useful? I had



Welcome to Biostar! Community Log In Sign Up

Question: Salmon unmapped reads


Hi All

I have some samples with low mapping in Salmon (40% and less) that have higher alignments in Tophat, and I'm trying to troubleshoot.

I picked some of the unmapped reads (from writeunmapped salmon parameter) and Blat them to human.

Some have 2 or more matches with identity 99% to 100% And some have many many matches, I need to scroll the page down too much. Many of these matches are 100% and some range between 85% to 100% identity.

12 weeks ago by Sharon · 150



Welcome to Biostar! Community Log In Sign Up

Question: RNA-Seq analysis using STAR and Salmon


Hello! I am having some trouble figuring out how to use Salmon. I have around 30 different samples which I trimmed using bmap then aligned them using STAR. I have all of the BAM files from this alignment. Should I

low mapping rate ? #160

Open atasub opened this issue on Oct 6, 2017 · 7 comments

atasub commented on Oct 6, 2017 · edited by rob-p

I recently ran Salmon by quasi-mapping-based mode and when I checked the salmon_quant.log file, saw that mapping rate was around ~%65-68 for all of the samples. Do you have any suggestions to improve the mapping rate? I used "--libType A" to to infer the library type info and got a warning that "Greater than 5% of the fragments disagreed with the provided library typ", but I guess this is not an issue. This is an example for one of the "lib_format_counts.json" files:




Welcome to Biostar! Community Log In Sign Up

Question: Salmon: Optimal k-mer size (for indexing) for RNA-seq data alignment using reference genome

Dear all,

I am using salmon to evaluate how the L.donovani gene expression varies in the two different (promastigotes and amastigotes). For that I downloaded two RNA-seq data from SRA and reference L.donovani coding sequence coding sequences (CDS)

In order to quantify gene expression with salmon I have to index the CDS using a specific salmon manual they use a 31-mer for 75bp reads. My question is whether is reasonable to use a value i.e. 0,413reads length or if there's any other advisable value. I heard that some people use a length whereas I also found this post where is mentioned between 0.5 and 0.66*reads length



Welcome to Biostar! Community Log In Sign Up

Question: Salmon very low mapping

Hi All

Tunable parameters

Most users rely on the default parameter settings,

- which are meant to work well on average,
- but the most interesting examples are not typically "average".

default

```
... yl-lhqflspssnqrtdqyggsvlenrarlvlevvdavcnewsad-RIGIRVSPigtfq ...  
... kP-LGVKLPPyf--dlvhfdimaeilnqfpltyvsnv-nsig----nglfidpeaesv ...  
... yl-lnqfldphsnttrtdeyggsvlenrarftlevvdalveaighe-KVGLRLSPygvfn ...  
... yl-plqflnpyynkrtdkyggsvlenrarfwletlekvhavgsdcAIATRF---GVdt ...  
... kvPLYVKLSPnv-tdivpiakaveaagadgltmintl-----mgvrfdlkrqp ...
```

alternate

```
... gsvenrarlvlevvdavcnewsad-RIGIRVSPigtfqnvndngpnee--adalyl--- ...  
... ydfeatekllke----vftfftk-PLGVKLPPyf-----dlvhfdim ...  
... gsienrarftlevvdalveaighe-KVGLRLSPygvfnsmggaetgivaqyayvage ...  
... gslenrarfwletlekvhavgsdcAIATRFV-----dtvygpgg ...  
... tdpevaaalvka----ckavskv-PLYVKLSPnvt-----divpiaka ...
```

The default parameter choices misaligns this region of the sequences.

Tunable parameters

It's not just a problem in computational biology!

Journal of Artificial Intelligence Research 36 (2009) 267-306

Submitted 06/09; published 10/09

SATzilla: Portfolio-based Algorithm Selection for SAT

Lin Xu
Frank Hutter
Holger H. Hoos
Kevin Leyton-Brown
*Department of Computer Science
University of British Columbia
201-2366 Main Mall, BC V6T 1Z4, CANADA*

XULIN730@CS.UBC.CA
HUTTER@CS.UBC.CA
HOOS@CS.UBC.CA
KEVINLB@CS.UBC.CA

ParamILS: An Automatic Algorithm Configuration Framework

Frank Hutter
Holger H. Hoos
Kevin Leyton-Brown
*University of British Columbia, 2366 Main Mall
Vancouver, BC, V6T1Z4, Canada*

Thomas Stützle
*Université Libre de Bruxelles, CoDE, IRIDIA
Av. F. Roosevelt 50 B-1050 Brussels, Belgium*

HUTTER@CS.UBC.CA
HOOS@CS.UBC.CA
KEVINLB@CS.UBC.CA

STUETZLE@ULB.AC.BE

Swarm and Evolutionary Computation 1 (2011) 19-31



Home Solutions ▾ Blog

Concertio Launches Optimizer Studio to Help Performance Engineers and IT Professionals Achieve Peak System Performance

by admin | Feb 22, 2018 | News | 0 comments



Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo



Invited paper

Parameter tuning for configuring and analyzing evolutionary algorithms

A.E. Eiben*, S.K. Smit¹

Department of Computer Science, Vrije Universiteit Amsterdam De Boelelaan 1081a 1081 HV, Amsterdam, Netherlands

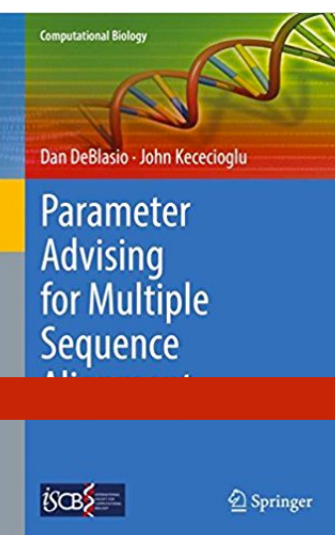
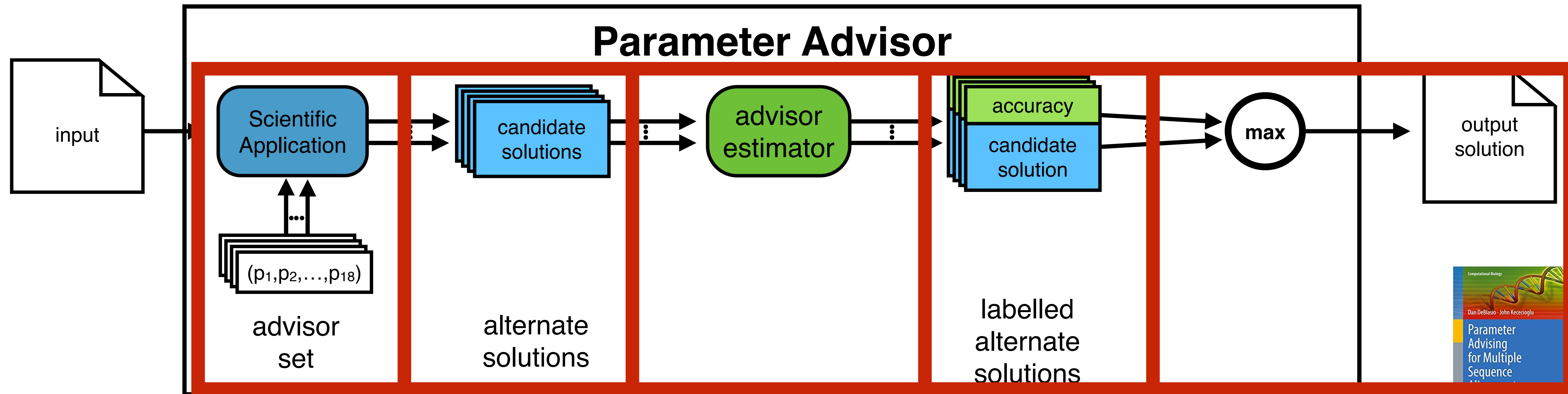
ARTICLE INFO

ABSTRACT

Parameter advising framework

Steps of advising:

- An **advisor set** of parameter choice vectors is used to obtain candidates.
- Solutions are ranked based on the **accuracy estimation**.
- The **highest ranked** candidate is returned.



Parameter advising framework

Steps of advising:

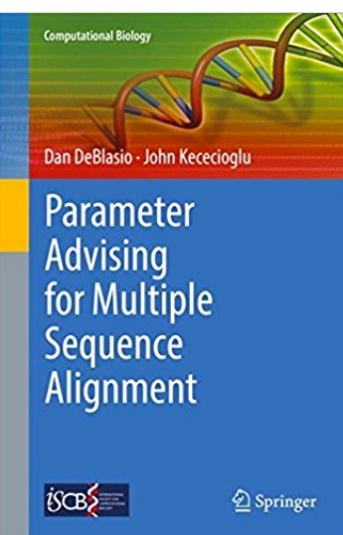
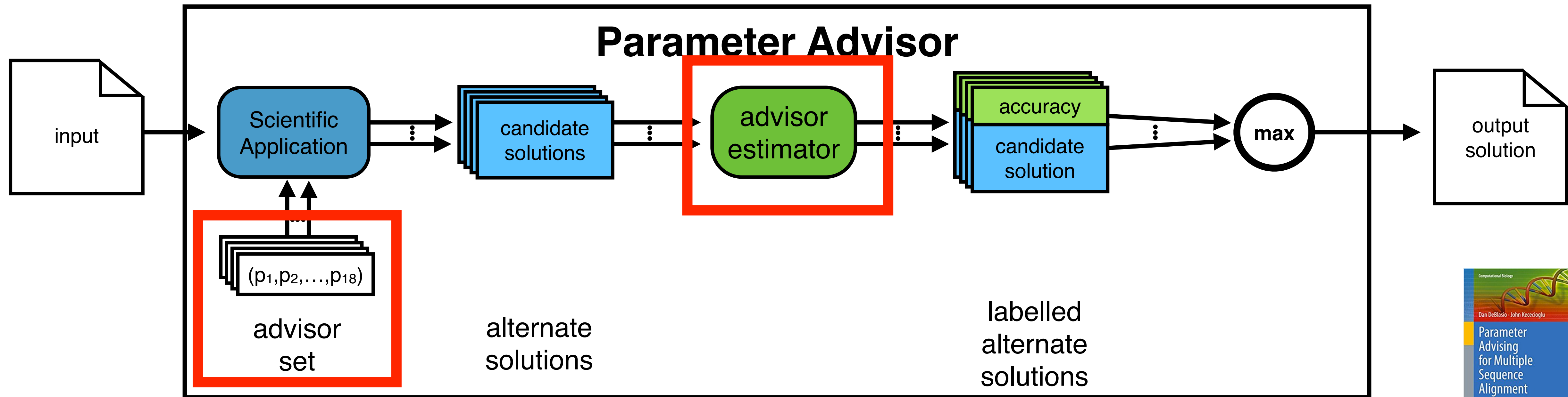
- An **advisor set** of parameter choice vectors is used to obtain candidates.
- Solutions are ranked based on the **accuracy estimation**.
- The **highest ranked** candidate is returned.



Parameter advising framework

Components of an advisor:

- An **advisor set** of parameter choice vectors.
- An **advisor estimator** to rank solutions.



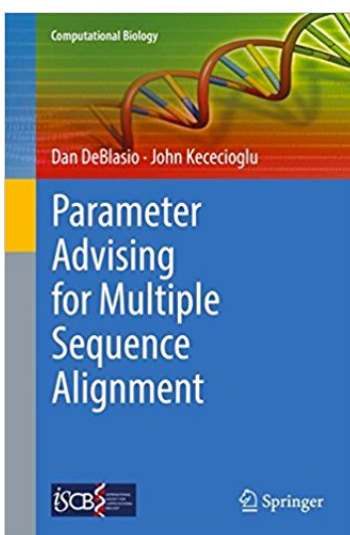
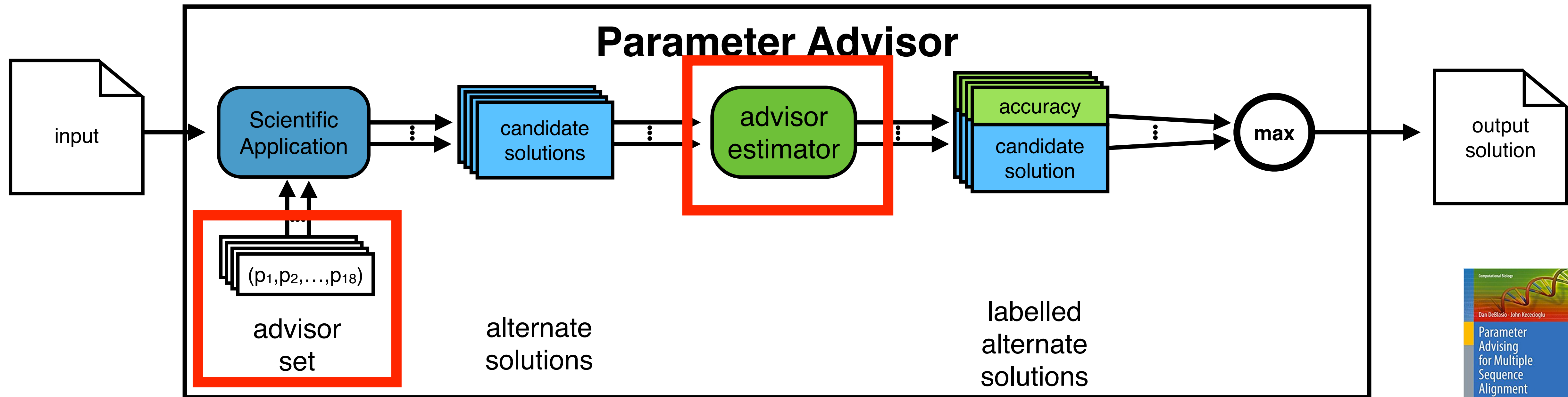
Parameter advising framework

Components of an advisor:

- An **advisor set** of parameter choice vectors.
- An **advisor estimator** to rank solutions.

A good advisor set:

- Small
- Representative



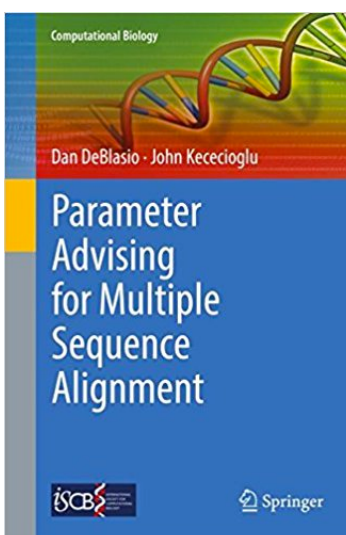
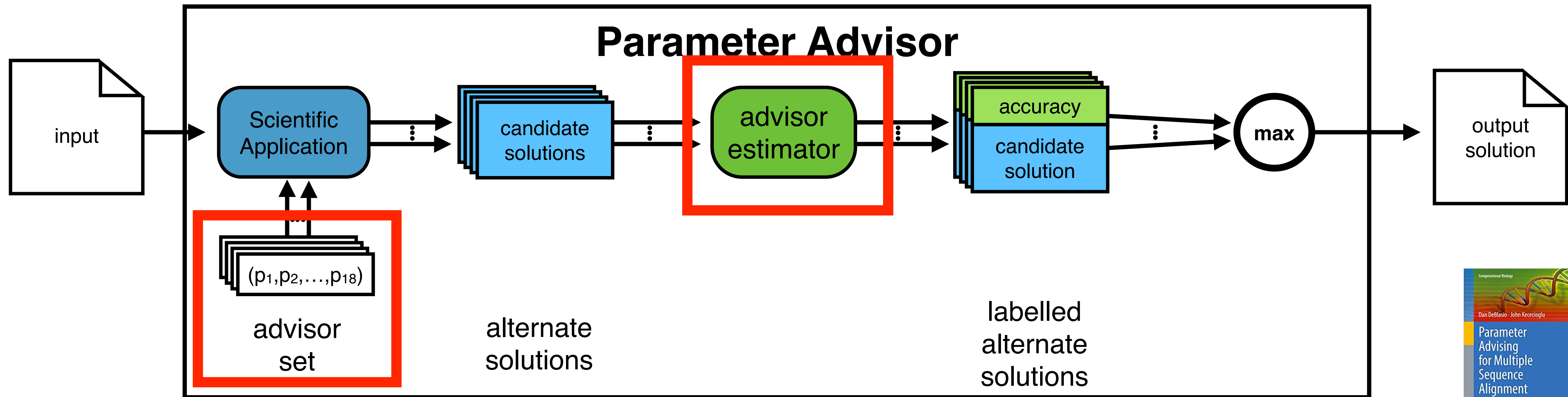
Parameter advising framework

Components of an advisor:

- An **advisor set** of parameter choice vectors.
- An **advisor estimator** to rank solutions.

A good advisor estimator:

- Efficient
- Rank Solutions Well



Background: Sequence Alignment

sequence alignment

Given

- a pair of sequences S_1, S_2 with lengths m and n , and
- an alignment objective function

find an $2 \times L$ matrix

- where $\max(m, n) < L < m + n$,
- each row represents one sequence from the set with inserted gaps, and
- is optimal under the objective function.



Edit Distance

A specific version of the the alignment problem

- the objective function is simply the count of operations:
 - *replace* one character with another -- R
 - *delete* a character from the first string -- D
 - *insert* a character from the second string -- I

Example: $S_1 = \mathbf{baseball}$ & $S_2 = \mathbf{ballcap}$.

RRR DR
baseball
ballcap

- 5 operations: change $s \rightarrow l$, $e \rightarrow l$, $b \rightarrow c$, delete l , $l \rightarrow p$

Alignment

We associate a **similarity score** with each pair of aligned characters:

- for characters $x, y \in \Sigma \cup \{-\}$
- define $\delta(x, y)$ to be the similarity of x and y

Let the score, Δ , of an **alignment** ($A = \{S'_1, S'_2\}$), be defined as

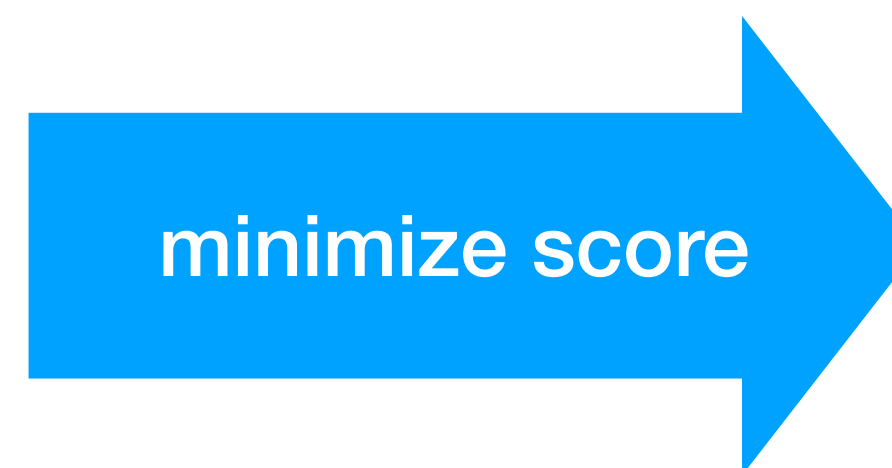
$$\Delta(A) =: \sum_{1 \leq i \leq |S'_1|} \delta(S'_1[i], S'_2[i])$$

Goal of alignment is to maximize that sum

Alignment

δ for edit distance

	-	a	b	c	e	l	p	s
-		-1	-1	-1	-1	-1	-1	-1
a	-1	0	-1	-1	-1	-1	-1	-1
b	-1	-1	0	-1	-1	-1	-1	-1
c	-1	-1	-1	0	-1	-1	-1	-1
e	-1	-1	-1	-1	0	-1	-1	-1
l	-1	-1	-1	-1	-1	0	-1	-1
p	-1	-1	-1	-1	-1	-1	0	-1
s	-1	-1	-1	-1	-1	-1	-1	0



baseball
ballca-p

Needleman-Wunsch

brute-force: compute all possible alignments and score them

- would take exponential time to compute the optimal alignment

using dynamic programming Needleman and Wunsch [1970] found that the optimal alignment can be computed in $O(mn)$ -time.

Needleman-Wunsch

Dynamic programming, generally, works by:

- solving **sub-problems**,
- **storing** the results, then
- **combine** the solutions to find the answer.

The sub-problem we will solve?

- **Given two strings $S[1...n]$ and $T[1...m]$, find the best alignment**

Needleman-Wunsch

Dynamic programming, generally, works by:

- solving **sub-problems**,
- **storing** the results, then
- **combine** the solutions to find the answer.

The sub-problem we will solve?

- **Given two strings $S[1...n]$ and $T[1...m]$, find the best alignment given the best alignments of:**
 - $S[1...(n-1)]$ and $T[1...m]$,
 - $S[1...n]$ and $T[1...(m-1)]$, and
 - $S[1...(n-1)]$ and $T[1...(m-1)]$

Needleman-Wunsch

Define an $n \times m$ array V

- the cell $V(i,j)$ will hold the score of the best sub alignments of $S[1...i]$ and $T[1...j]$

The recurrence relation (the base of any dynamic program)

$$V(i,j) = \max \begin{cases} V(i-1,j-1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i-1,j) + \delta(S[i], -) & \text{delete} \\ V(i,j-1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

The initialization is:

$$V(0,0) = 0$$

$$V(0,j) = V(0,j-1) + \delta(-, T[j])$$

$$V(i,0) = V(i-1,0) + \delta(S[i], -)$$

Optimal alignment score is in $V(n,m)$

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

	A	A	C	C	C	G
	0					
A	-1					
A						
G						
G						
C						
C						

The cost of the optimal alignment of "A" and ""

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

	A	A	C	C	C	G
	0					
A	-1					
A	-2					
G						
G						
C						
C						

The cost of the optimal alignment of "AA" and ""

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

		A	A	C	C	C	G
		0	-1				
A		-1					
A		-2					
G		-3					
G		-4					
C		-5					
C		-6					

The cost of the optimal alignment of "" and "A"

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

		A	A	C	C	C	G
	0	-1	-2	-3	-4	-5	-6
A	-1						
A	-2						
G	-3						
G	-4						
C	-5						
C	-6						

The cost of the optimal alignment of "A" and "A"

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

$$V(0,0) + \delta(A,A) = 1$$

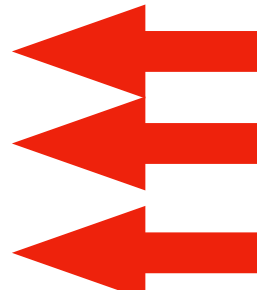
$$+ \begin{matrix} \mathbf{A} \\ \mathbf{A} \end{matrix}$$

$$V(0,1) + \delta(A,-) = -2$$

$$\begin{matrix} - & \mathbf{A} \\ \mathbf{A} & + \\ & - \end{matrix}$$

$$V(1,0) + \delta(-,A) = -2$$

$$\begin{matrix} \mathbf{A} & - \\ - & + \\ & \mathbf{A} \end{matrix}$$

$$V(i,j) = \max \begin{cases} V(i-1,j-1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i-1,j) + \delta(S[i], -) & \text{delete} \\ V(i,j-1) + \delta(-, T[j]) & \text{insert} \end{cases}$$


		A	A	C	C	C	G	
		0	-1	-2	-3	-4	-5	-6
A		-1	1					
A		-2						
G		-3						
G		-4						
C		-5						
C		-6						

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

		A	A	C	C	C	G
	0	-1	-2	-3	-4	-5	-6
A	-1	1 $+\delta(A,A)$	0 $+\delta(A,-)$				
A	-2						
G	-3						
G	-4						
C	-5						
C	-6						

Diagram illustrating the Needleman-Wunch scoring function. The table shows the score for each pair of characters (x, y) where x is the row and y is the column. The scores are: 0 for (x, x) and -1 for (x, y) where x ≠ y. Annotations highlight specific values: a green arrow points to the value 1 at (A, A) labeled $+\delta(A,A)$; a blue arrow points to the value 0 at (A, A) labeled $+\delta(A,-)$; and a red arrow points to the value -1 at (A, A) labeled $+\delta(-,A)$.

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

	A	A	C	C	C	G	
	0	-1	-2	-3	-4	-5	-6
A	-1	1	0 ^{+δ(A,C)}	-1 ^{+δ(-,C)}			
A	-2						
G	-3						
G	-4						
C	-5						
C	-6						

Diagram illustrating the Needleman-Wunch algorithm. The table shows the sequence of characters A, A, C, C, C, G. The values in the table represent the current score. Annotations show the effect of transitions: a green arrow points from the cell (A, C) to (A, C) with the label +δ(A,C), a blue arrow points from the cell (A, C) to (A, A) with the label +δ(A,-), and a red arrow points from the cell (A, C) to (A, C) with the label +δ(-,C).

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

	A	A	C	C	C	G	
	0	-1	-2	-3	-4	-5	-6
A	-1	1	0	-1	-2	-3	-4
A	-2	0	2	1	0	-1	-2
G	-3	-1	1	1	0	-1	0
G	-4	-2	0	0	0	-1	0
c	-5	-3	-1	1	1	1	0
c	-6	-4	-2	0	2	2	1

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

		A	A	C	C	C	G
	0	-1	-2	-3	-4	-5	-6
A	-1	1					
A	-2						
G	-3						
G	-4						
C	-5						
C	-6						

Diagram illustrating the Needleman-Wunch algorithm. The table shows the sequence of characters A, A, C, C, C, G. The values in the table represent the Needleman-Wunch score for each character at each position. The first row shows the initial scores: 0, -1, -2, -3, -4, -5, -6. The second row shows the scores after the first character 'A' is processed: -1, 1, -2, -3, -4, -5, -6. The third row shows the scores after the second character 'A' is processed: -2, 1, -2, -3, -4, -5, -6. The fourth row shows the scores after the third character 'C' is processed: -3, -2, -2, -3, -4, -5, -6. The fifth row shows the scores after the fourth character 'C' is processed: -4, -3, -3, -4, -5, -6. The sixth row shows the scores after the fifth character 'C' is processed: -5, -4, -4, -5, -6. The seventh row shows the scores after the sixth character 'G' is processed: -6, -5, -5, -6. Annotations include: a green arrow pointing from the cell (A, A) to the cell (A, A) with the label $+\delta(A,A)$; a blue arrow pointing from the cell (A, A) to the cell (A, -) with the label $+\delta(A,-)$; and a red arrow pointing from the cell (A, A) to the cell (-, A) with the label $+\delta(-,A)$.

Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

		A	A	C	C	C	G
	0	-1	-2	-3	-4	-5	-6
A	-1	1	0	-1	-2	-3	-4
A	-2	0	2	1	0	-1	-2
G	-3	-1	1	1	0	-1	0
G	-4	-2	0	0	0	-1	0
C	-5	-3	-1	1	1	1	0
C	-6	-4	-2	0	2	2	1

Needleman-Wunch

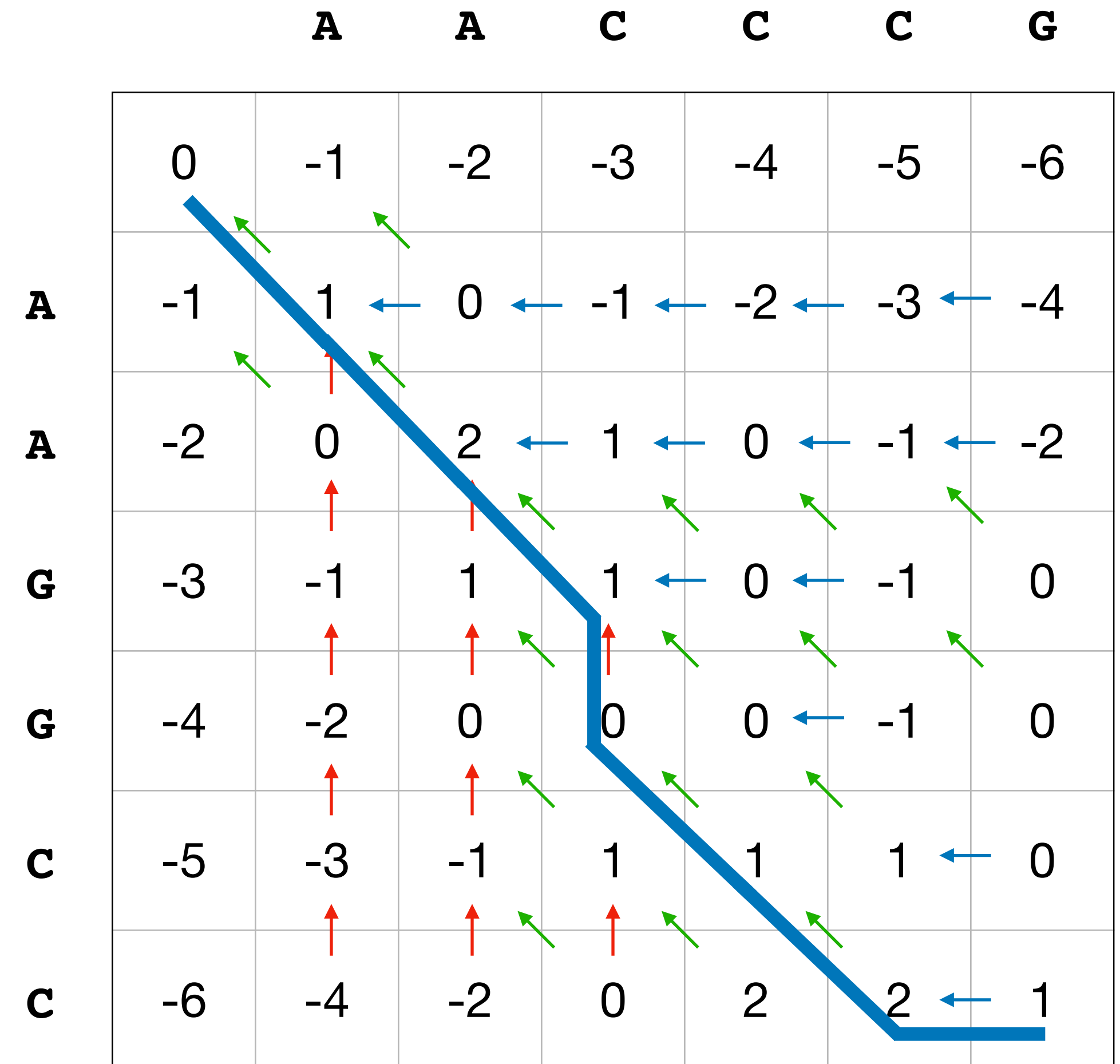
$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

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



Needleman-Wunch

$$\delta(-,x) = -1 \text{ for } x \in \Sigma$$

$$\delta(x,-) = -1 \text{ for } x \in \Sigma$$

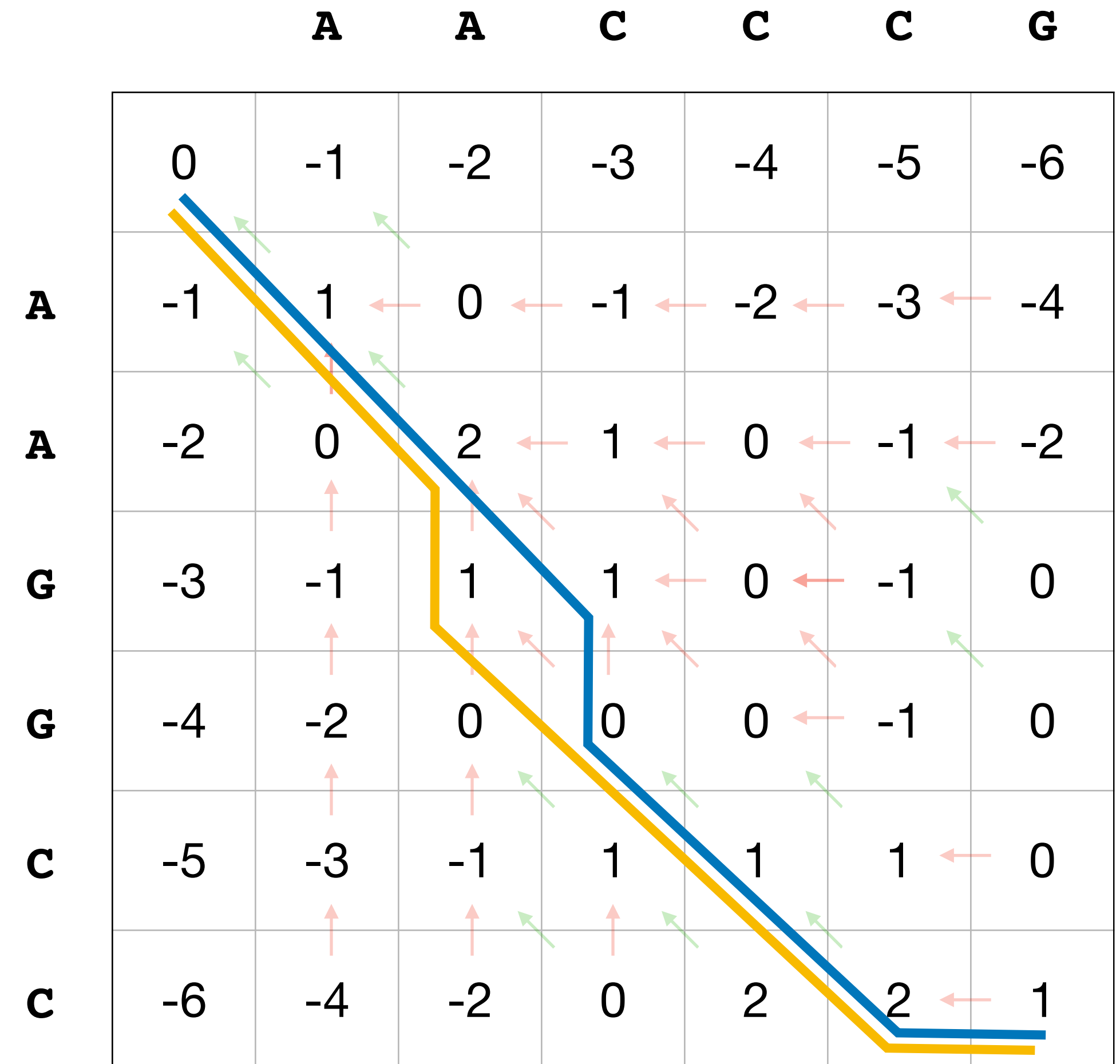
$$\delta(x,y) = 1 \text{ for } y = x$$

$$\delta(x,y) = -1 \text{ for } y \neq x$$

```

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

A A - C C C G
A A G G C C -
    
```



Needleman-Wunch

What about the running time and memory requirements?

- Filling in each cell of the table:

$O(1)$ -time, $O(1)$ -space

- Table is $n \times m$

- Filling in the table:

$O(mn)$ -time, $O(mn)$ -space

- Traceback?

- Each column of the alignment:

$O(1)$ -time

- Maximum Alignment Length:

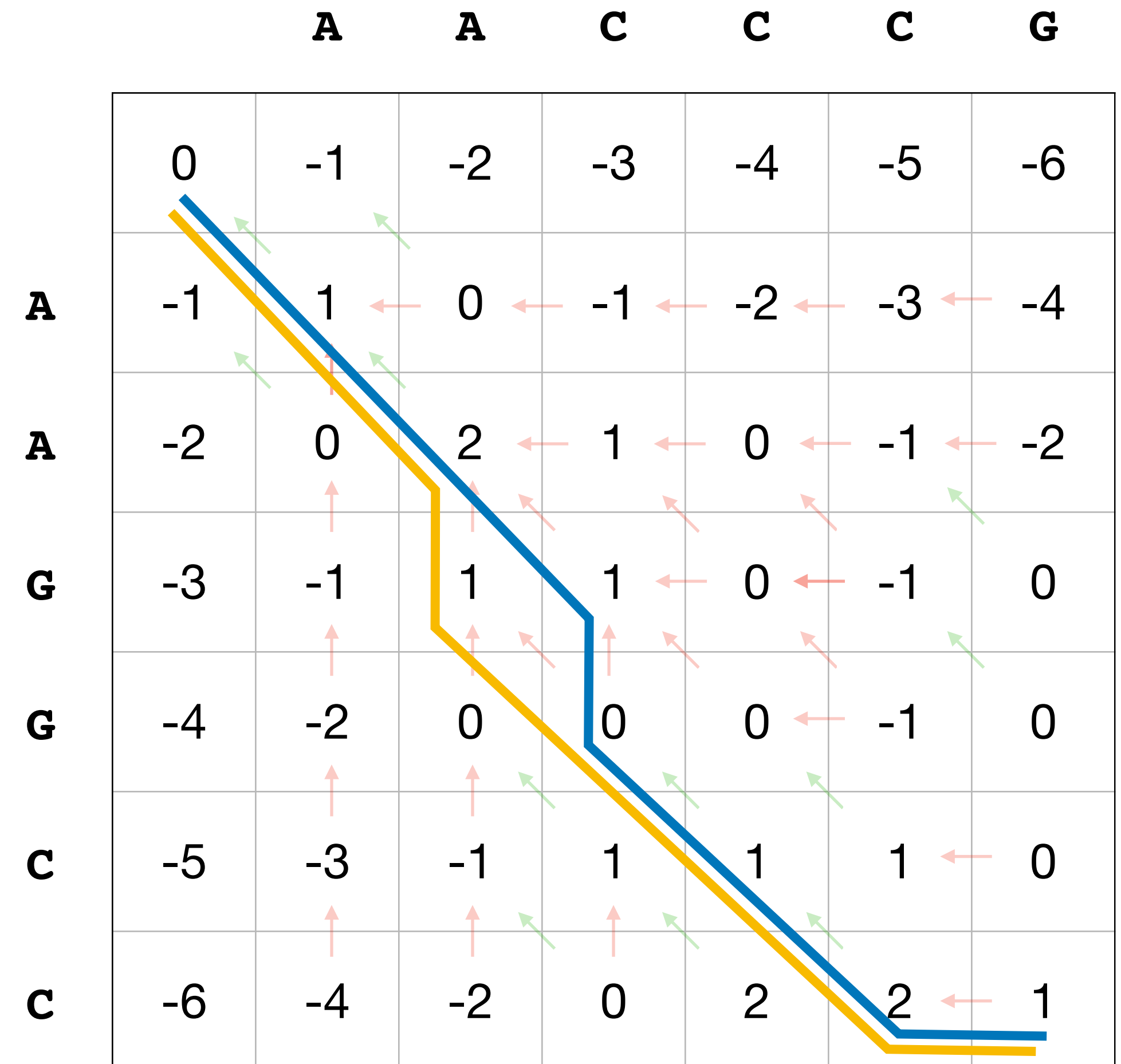
$O(m+n)$

(times the number of optimal alignments)

```

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

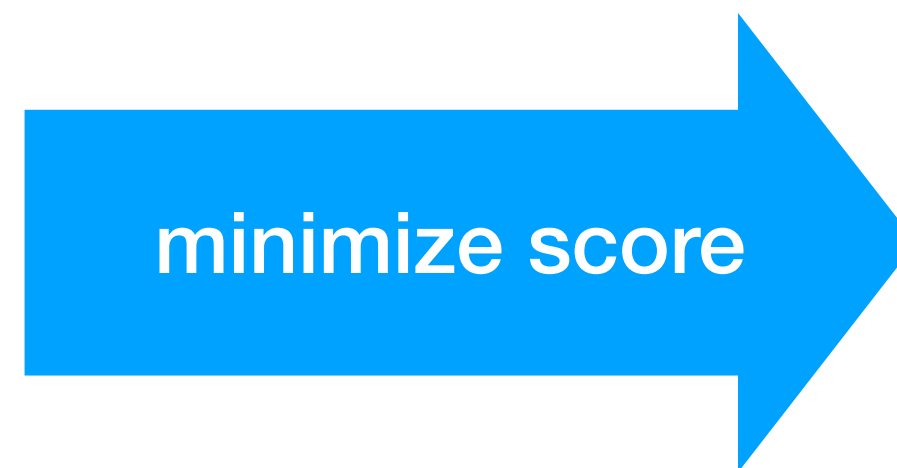
A A - C C C G
A A G G C C -
    
```



Alignment

δ in the previous example

	-	A	C	G
-		-1	-1	-1
A	-1	1	-1	-1
C	-1	-1	1	-1
G	-1	-1	-1	1

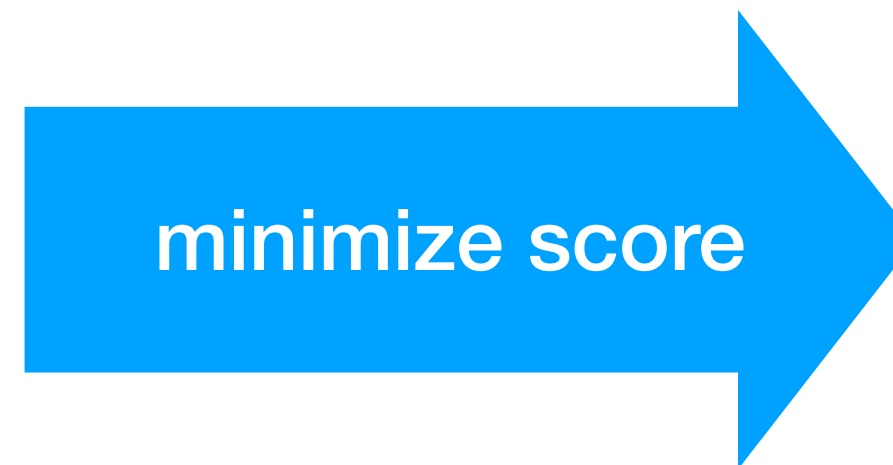


AAC-CCG
AAGGCC-

Alignment

change the δ

	-	A	C	G
-	0	0	0	0
A	0	2	-1	-1
C	0	-1	2	-1
G	0	-1	-1	2



AA--CCCG
AAGGCC--

Alignment

We can find δ s that can produce any of these alignments:

AAC-CCG
AAGGCC-

AA--CCCG
AAGGCC--

AACCCG
AAGGCC

How do we know which one of these is best?

Alignment

So far we have seen that if you have two sequences

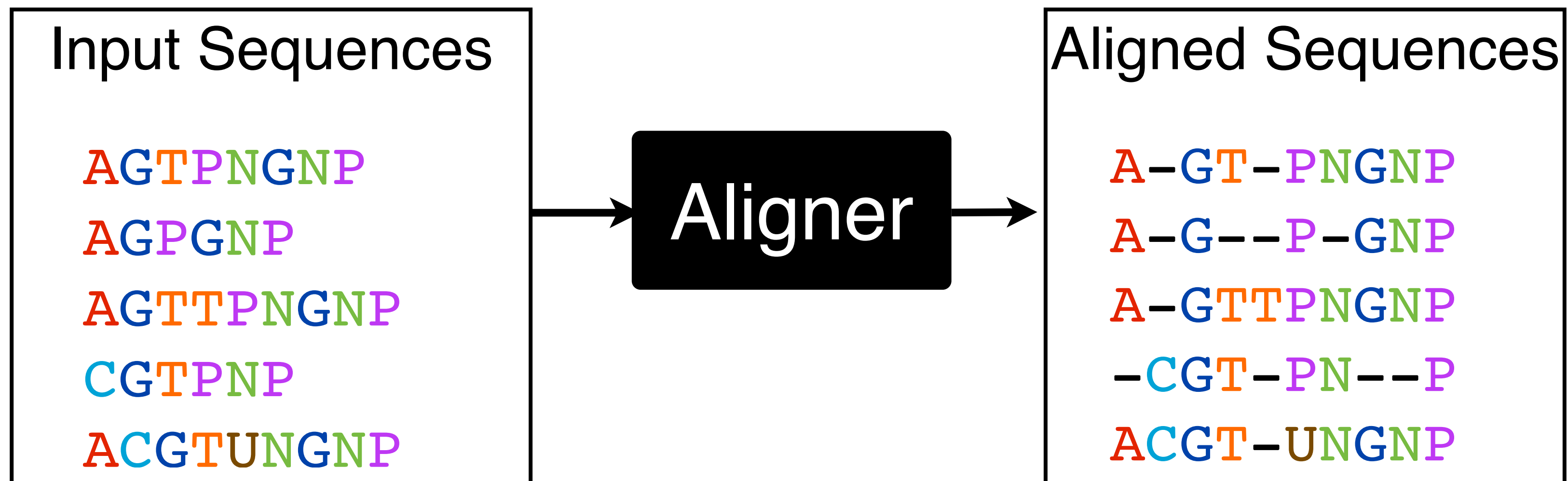
- if you define the **replacement score** (δ)
- you can find an **optimal** solution
- in **linear** time.

What if you have multiple sequences? Can it still be solved exactly*?

Multiple sequence alignment

A **fundamental problem** in bioinformatics.

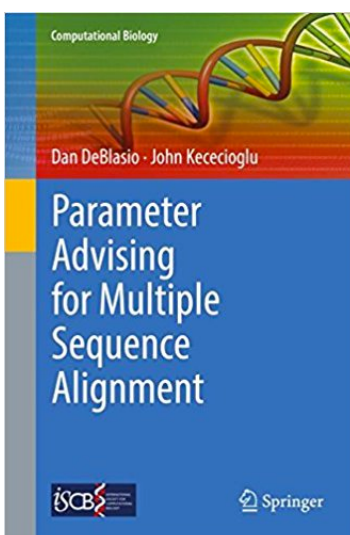
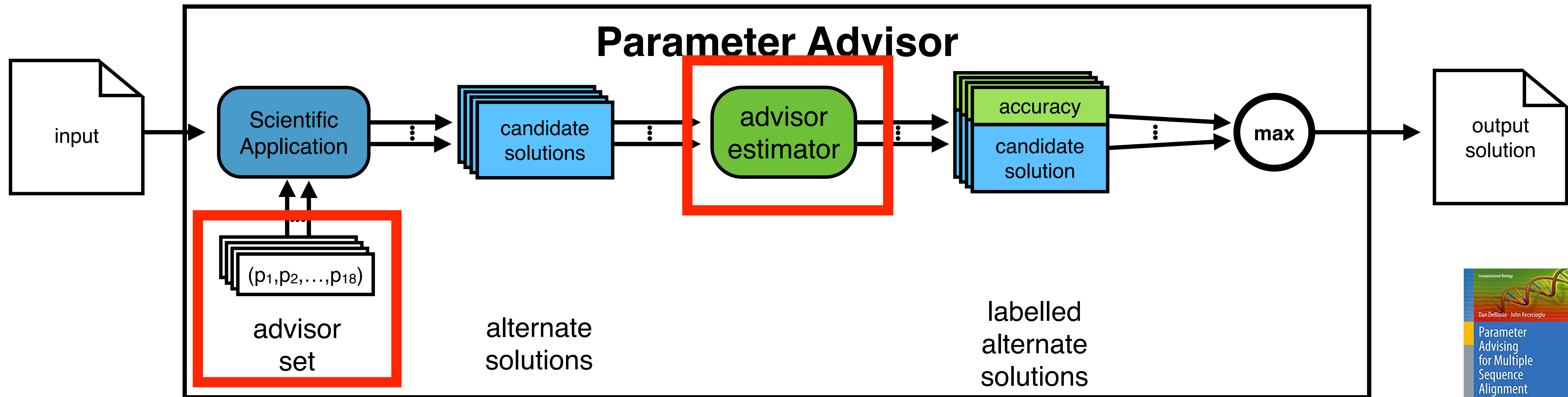
- **NP-Complete**
- many popular aligners
- many parameters whose values affect the output
- no standard metric for measuring accuracy without ground truth



Parameter advising framework

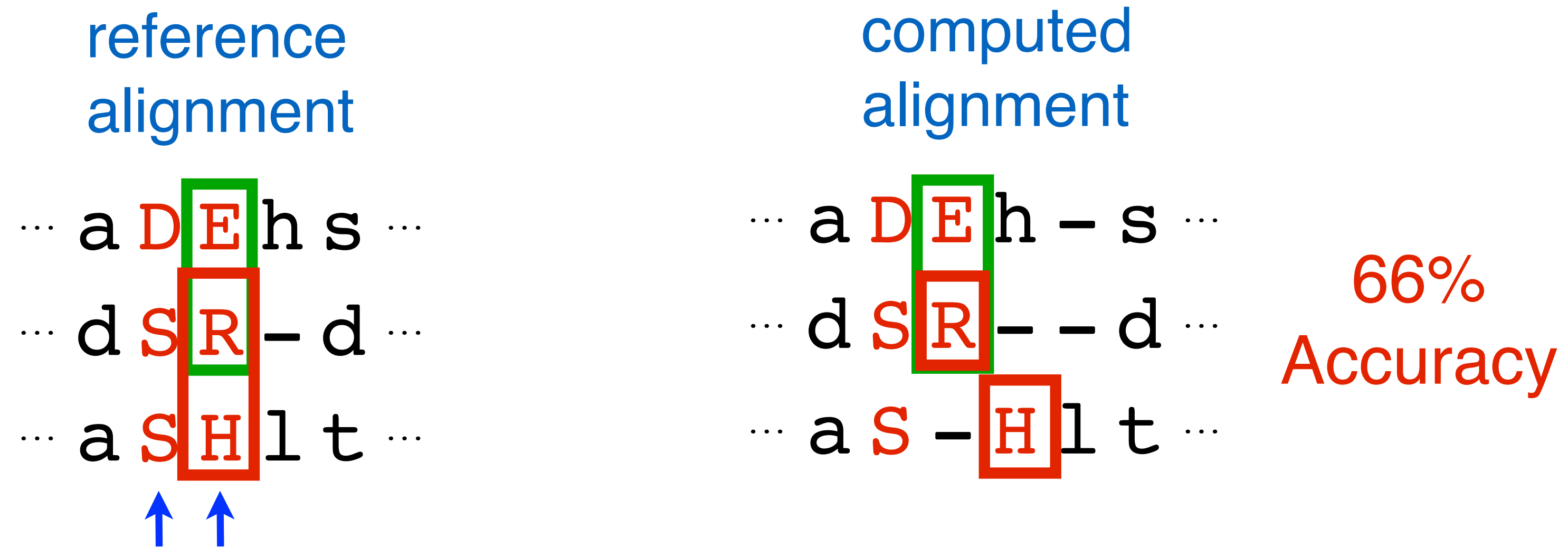
Components of an advisor:

- An **advisor set** of parameter choice vectors.
- An **advisor estimator** to rank solutions.



Accuracy estimation

Alignment accuracy is measured with respect to a reference alignment.



- accuracy is the **fraction of substitutions** from the reference that are in the computed alignment,
- measured on the **core columns** of the reference.

Accuracy estimation

Our estimator **Facet** (“**F**eature-based **AC**curacy **EsT**imator”)

- a polynomial on feature functions
- efficiently learns the coefficients from examples
- uses efficiently computed novel features

**Feature functions are the key:
uninformative features → uninformative estimator**

Accuracy estimation

The estimator $E(A)$ is a **polynomial** in the feature functions $f_i(A)$.

linear estimator

$$E(A) := \sum_i c_i f_i(A)$$

quadratic estimator

$$E(A) := \sum_i c_i f_i(A) + \sum_i \sum_j c_{ij} f_i(A) f_j(A)$$

Always linear in the coefficients.

Learning the estimator

We learn the estimator using **examples** consisting of

- an alignment, and
- its associated true accuracy.

Learning finds optimal **coefficients** that either fit

- accuracy values of the examples, or
- **accuracy differences** on pairs of examples,
- by solving a **linear** or quadratic program.

Learning the estimator

$$e_{a,b} \geq E(b) - E(a) = \sum_i c_i (f_i(b) - f_i(a))$$

$$e_{a,b} \geq 0$$

$\forall a, b \in \text{Examples} :$
 $\text{Accuracy}(a) > \text{Accuracy}(b)$

Feature functions

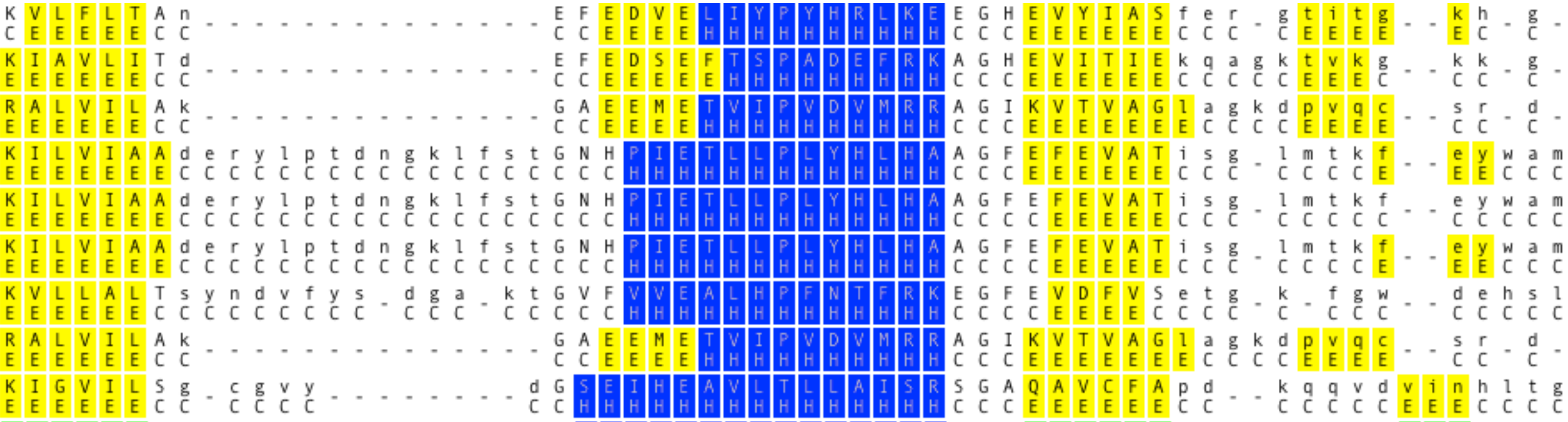
We use protein alignment feature functions that

- are **fast** to evaluate,
- measure **novel** properties,
- use **non-local** information,
- involve **secondary structure**.

Feature functions

There are three types of secondary structure

- α -helix,
- β -strand,
- coil.



Feature functions

Features based only on the **sequence** information

- Amino Acid Identity
- Average Substitution Score
- Information Content
- ...

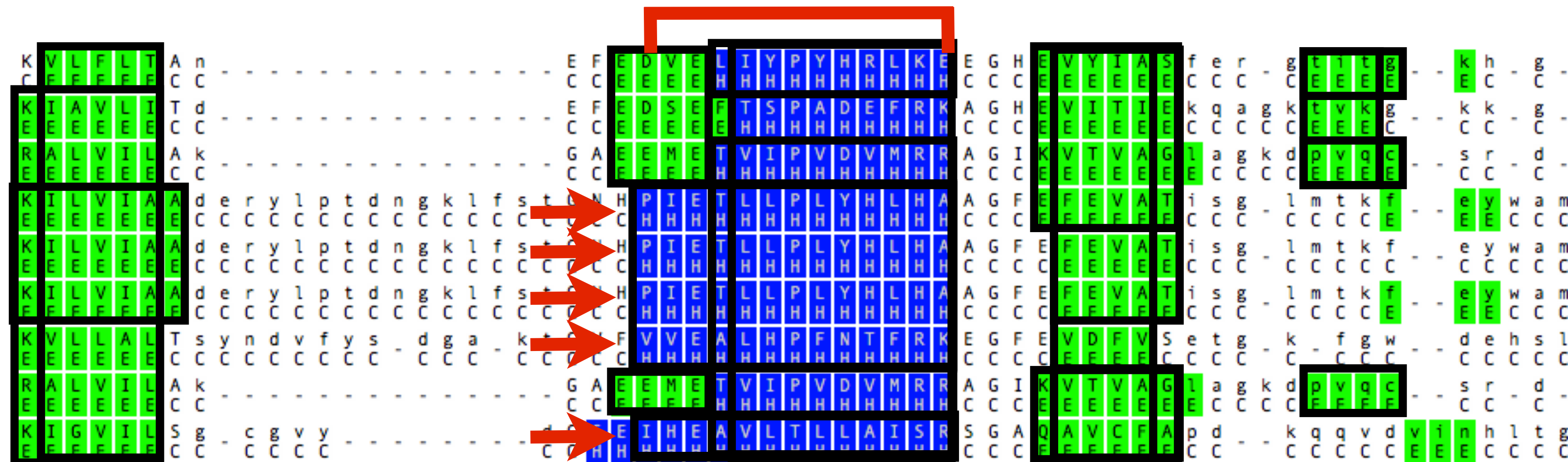
Features using predicted **secondary structure**

- Secondary Structure Percent Identity
- Secondary Structure Agreement
- Secondary Structure Blockiness
- ...

Secondary structure blockiness

A **block** B in alignment A is

- an interval of at least l columns,
- a subset of at least k rows,
- with the same secondary structure for all residues in B .



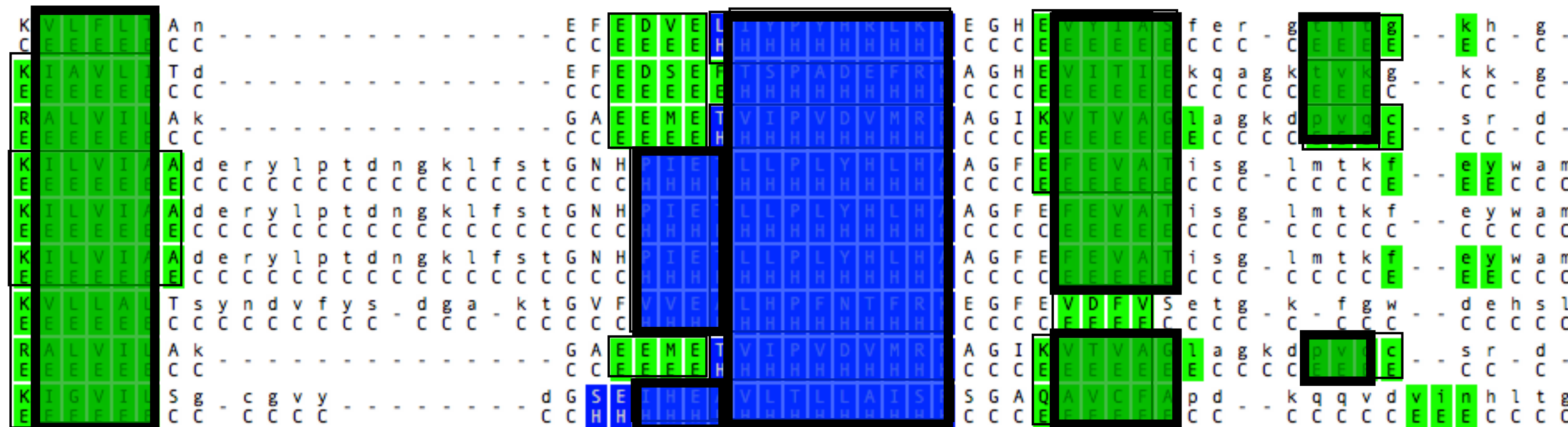
Secondary structure blockiness

A **packing** P for alignment A is

- a set of blocks from A ,
- whose columns are disjoint.

The value of P is the number of substitutions it contains.

The **Blockiness** feature is the maximum value of any packing.



Secondary structure blockiness

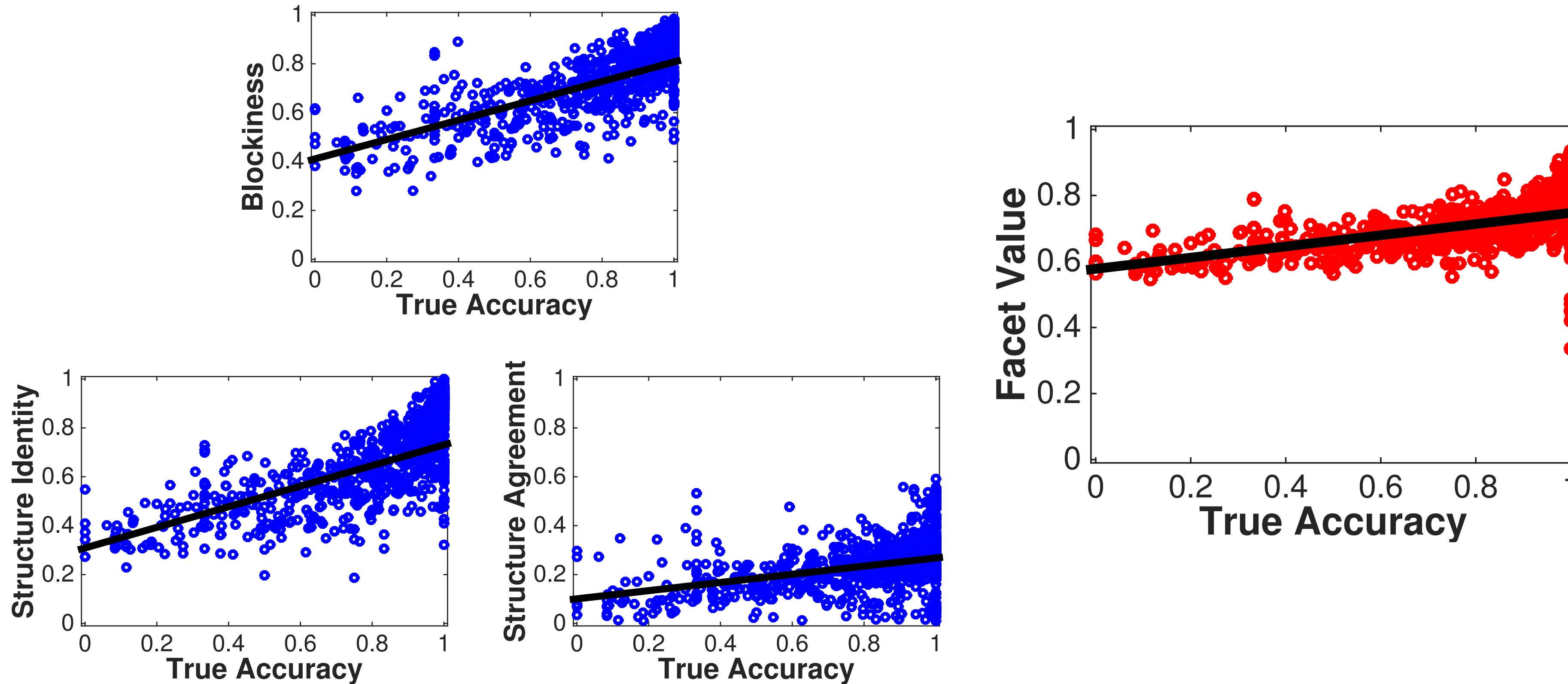
Theorem (Evaluating Blockiness)

Blockiness can be computed in $O(mn)$ time,
for an alignment with m rows and n columns.

Algorithm translates the problem into finding the longest path in a directed acyclic graph.

Accuracy estimation

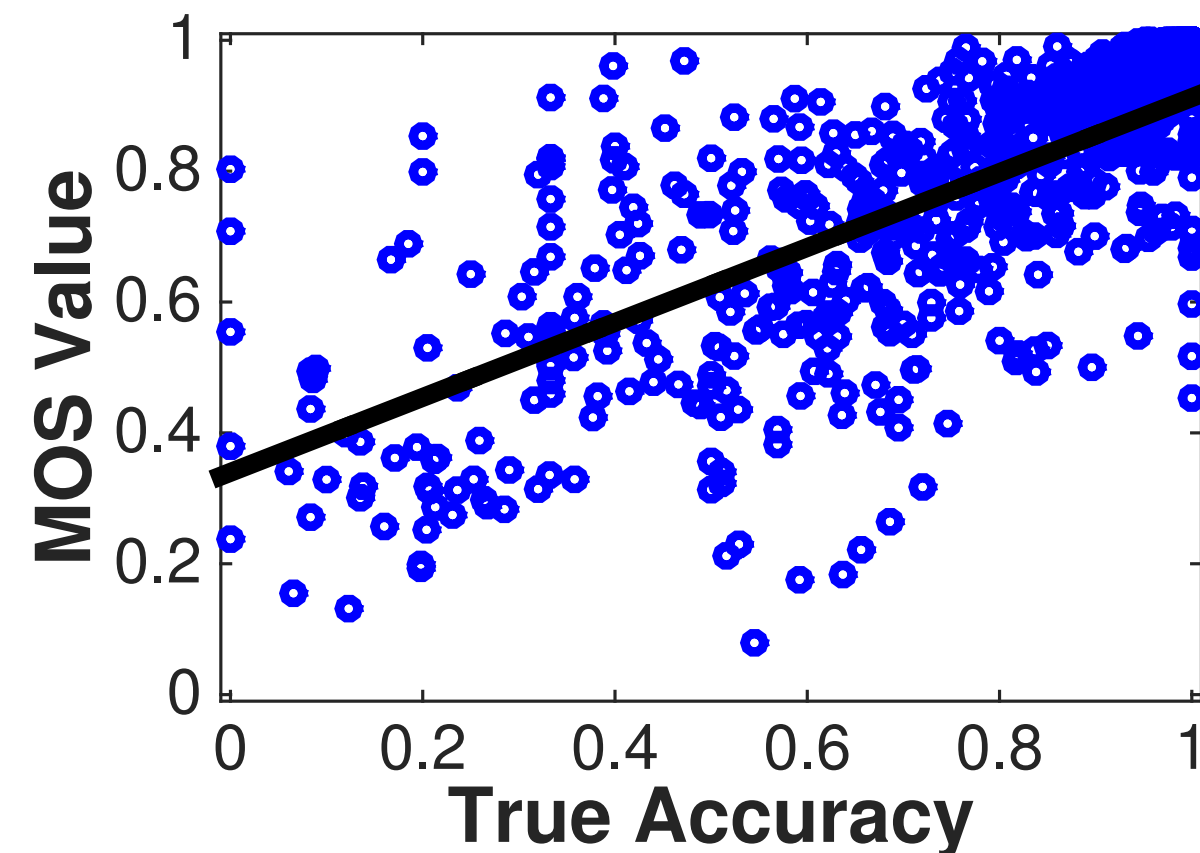
Best features trend well with accuracy.



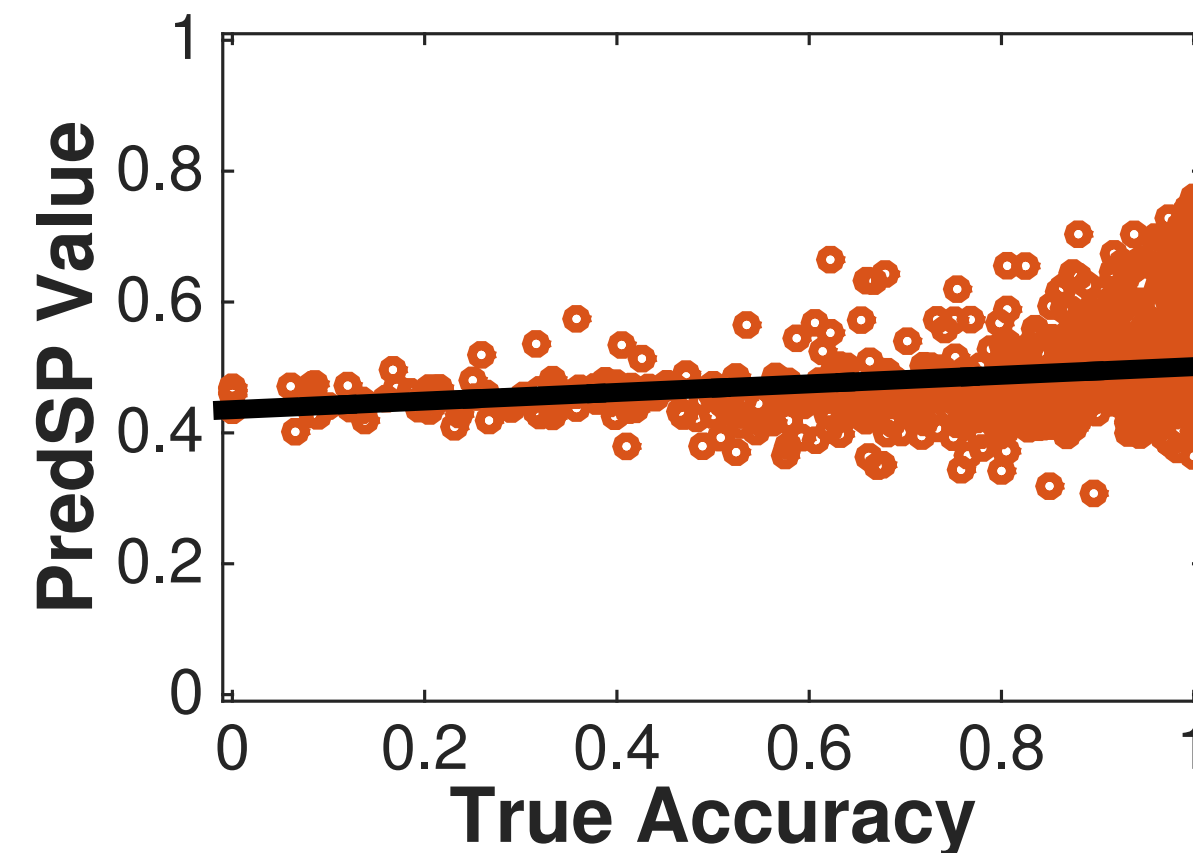
Facet estimator has **less spread** than its features.

Accuracy estimation

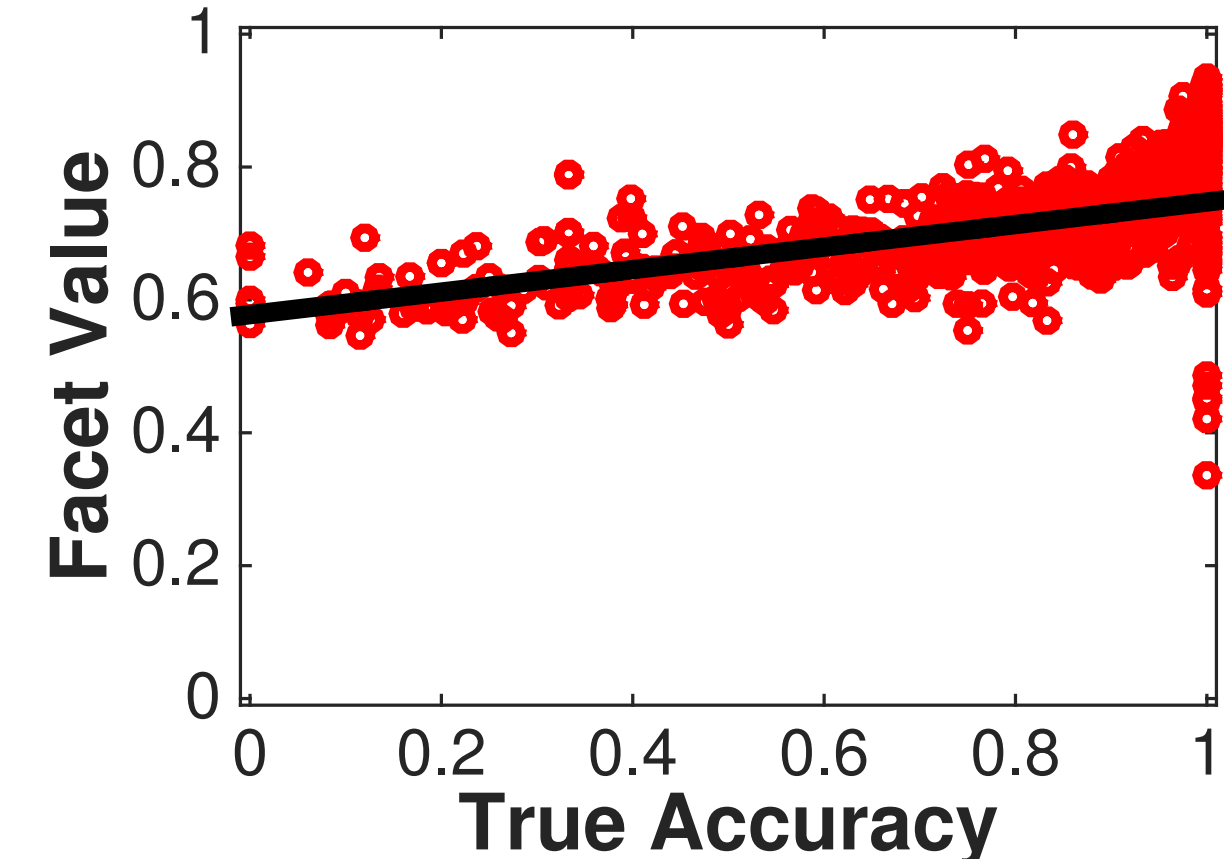
For parameter advising, an estimator should have high **slope** and low **spread**.



high slope,
high spread



low slope,
low spread



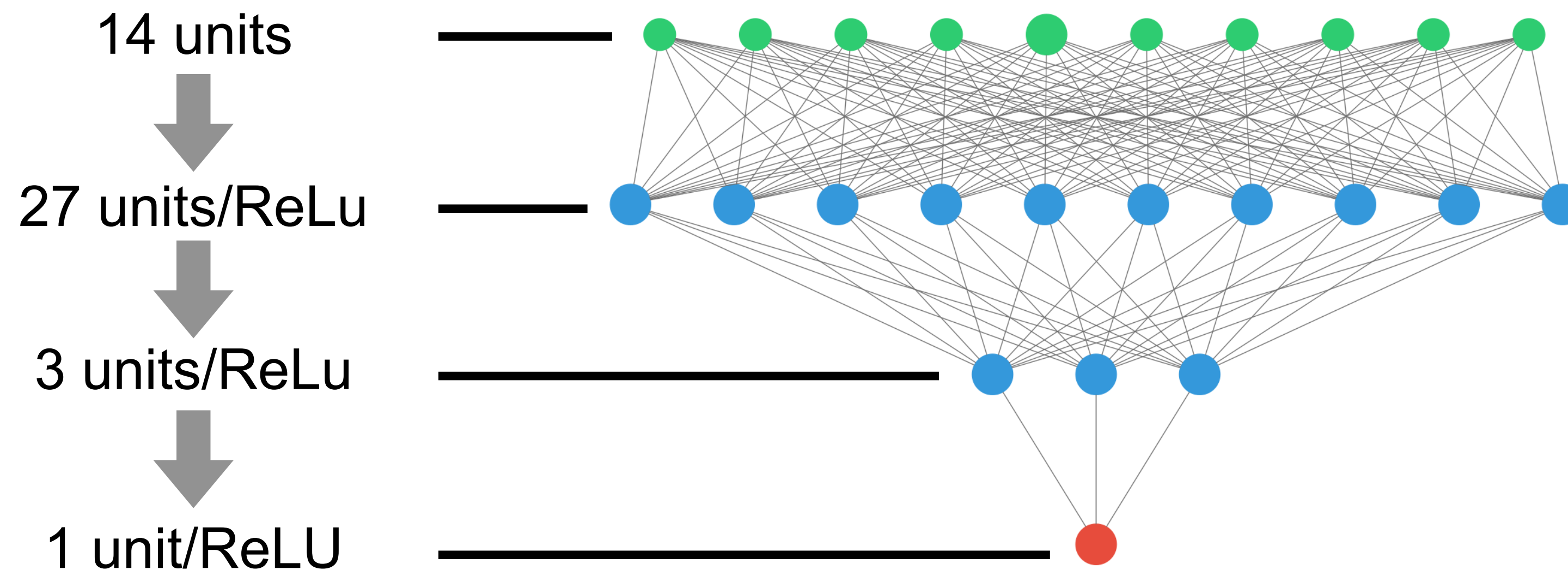
medium slope,
low spread

Facet's slope and spread is **best for advising**

Exploiting non-linearity

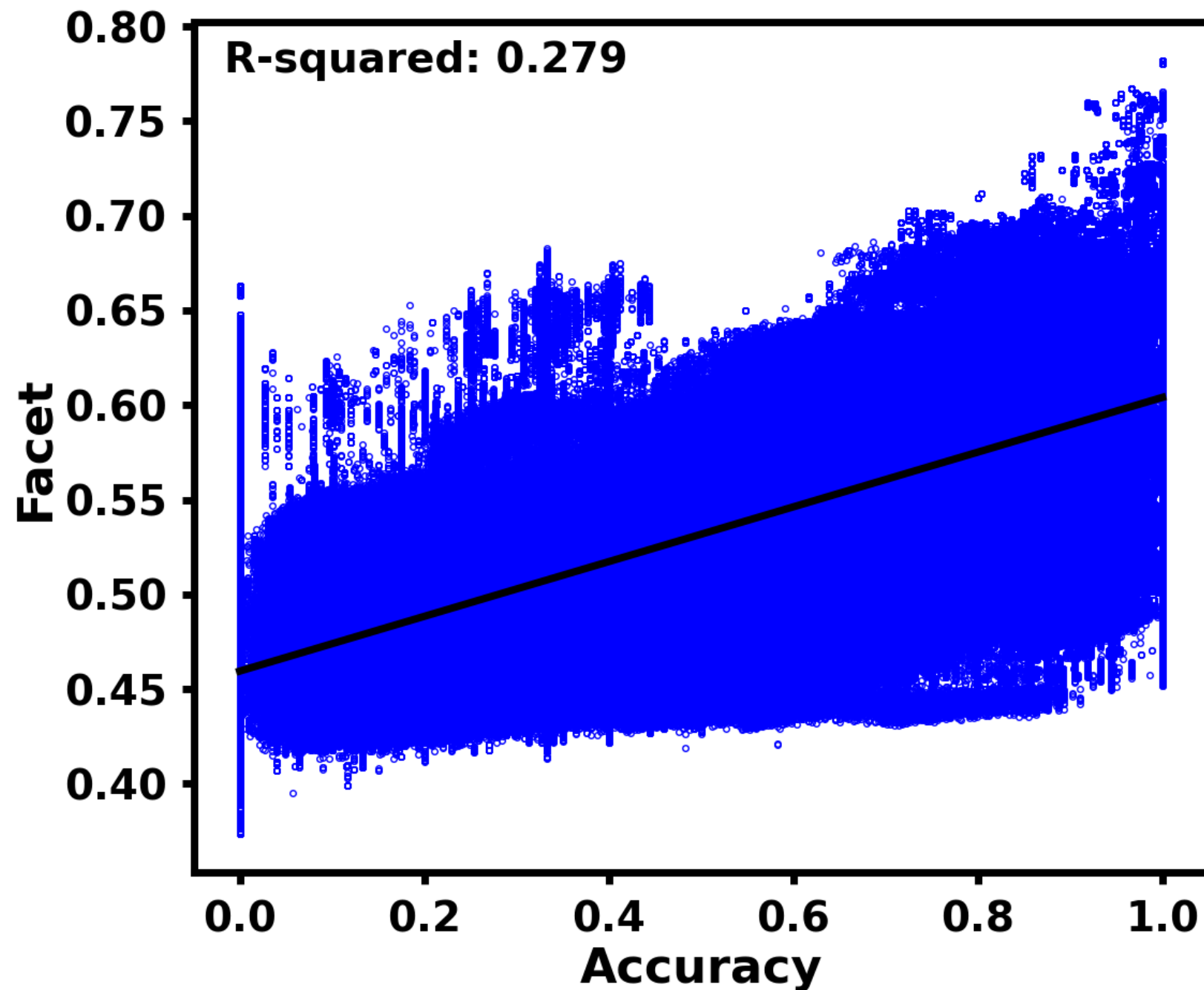
While we designed the features to scale **linearly** with accuracy, some show some **non-linear** behavior when plotted.

- Advanced machine learning allowed for the use of a **neural network predictor**.
- We also produced a much **larger training set** (now >14M alignments).

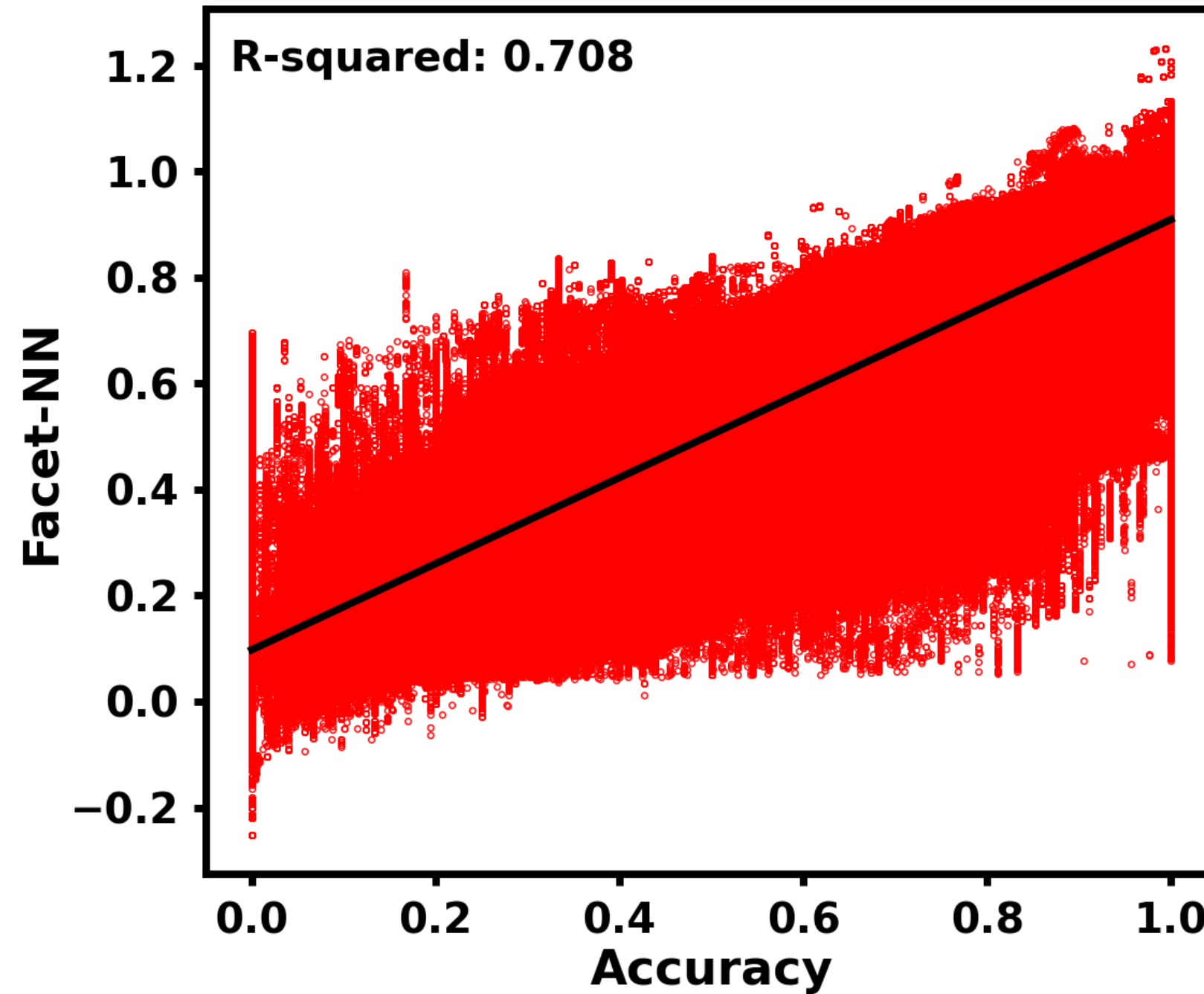


Exploiting non-linearity

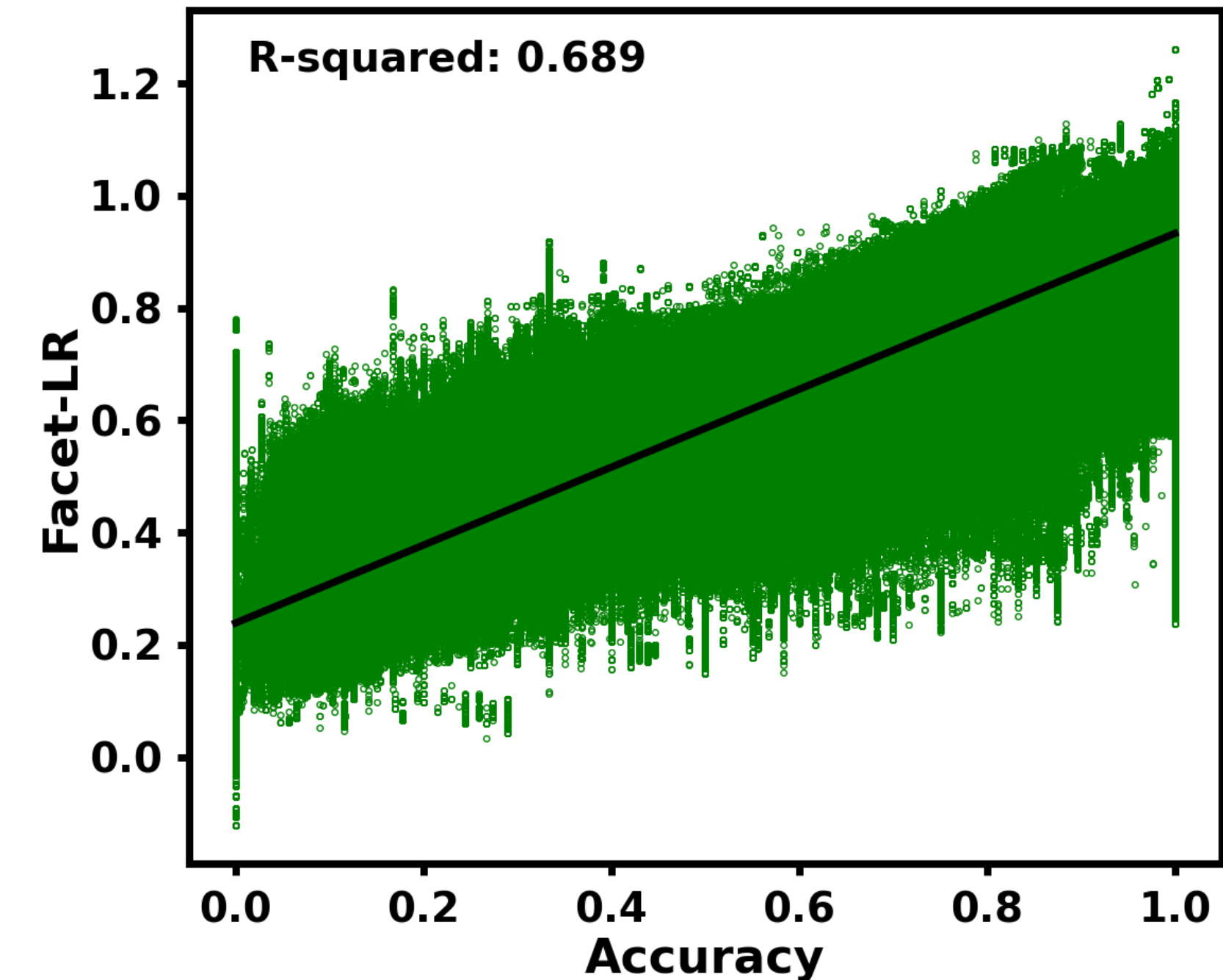
Previous Result



Neural Network

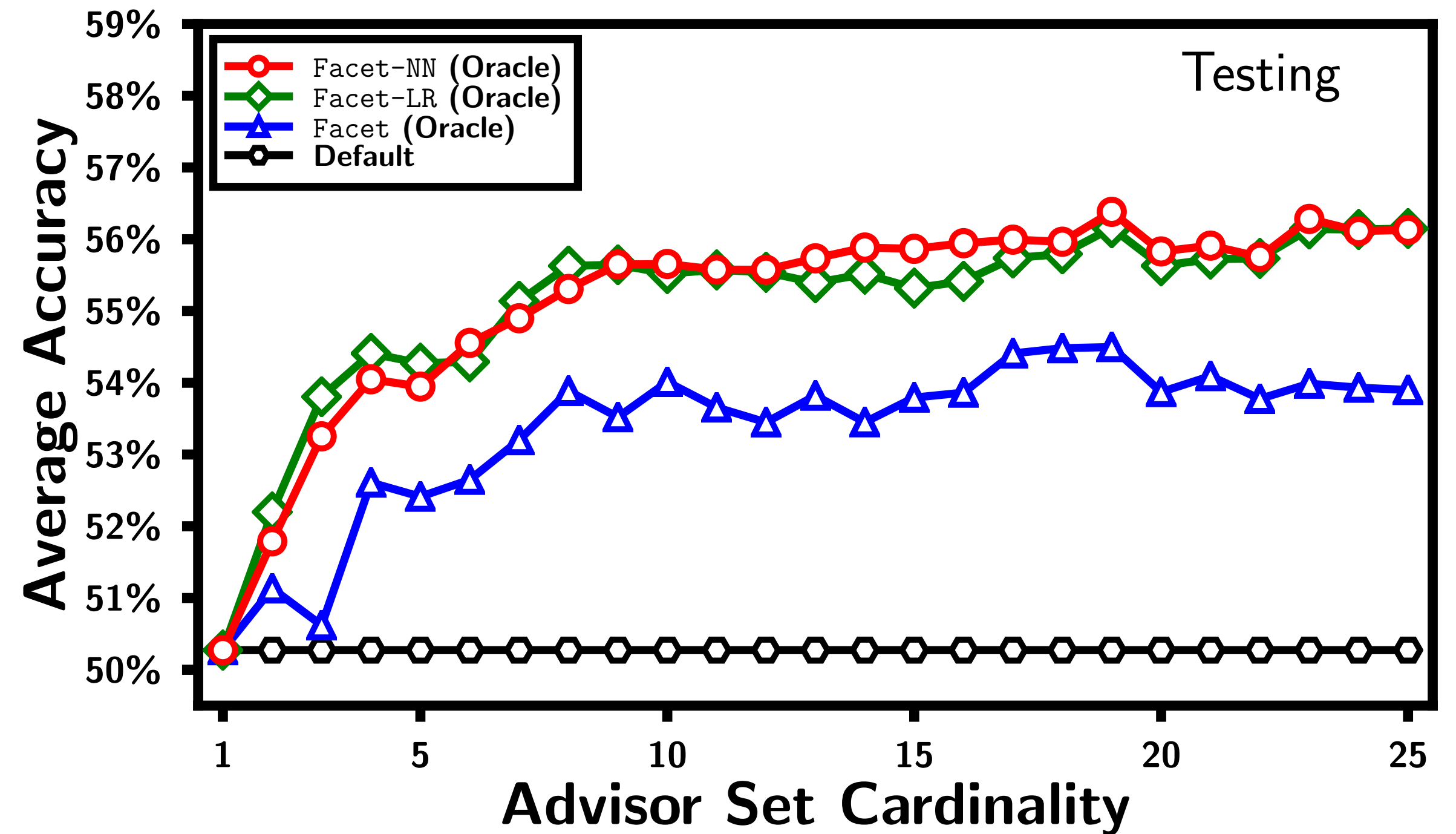
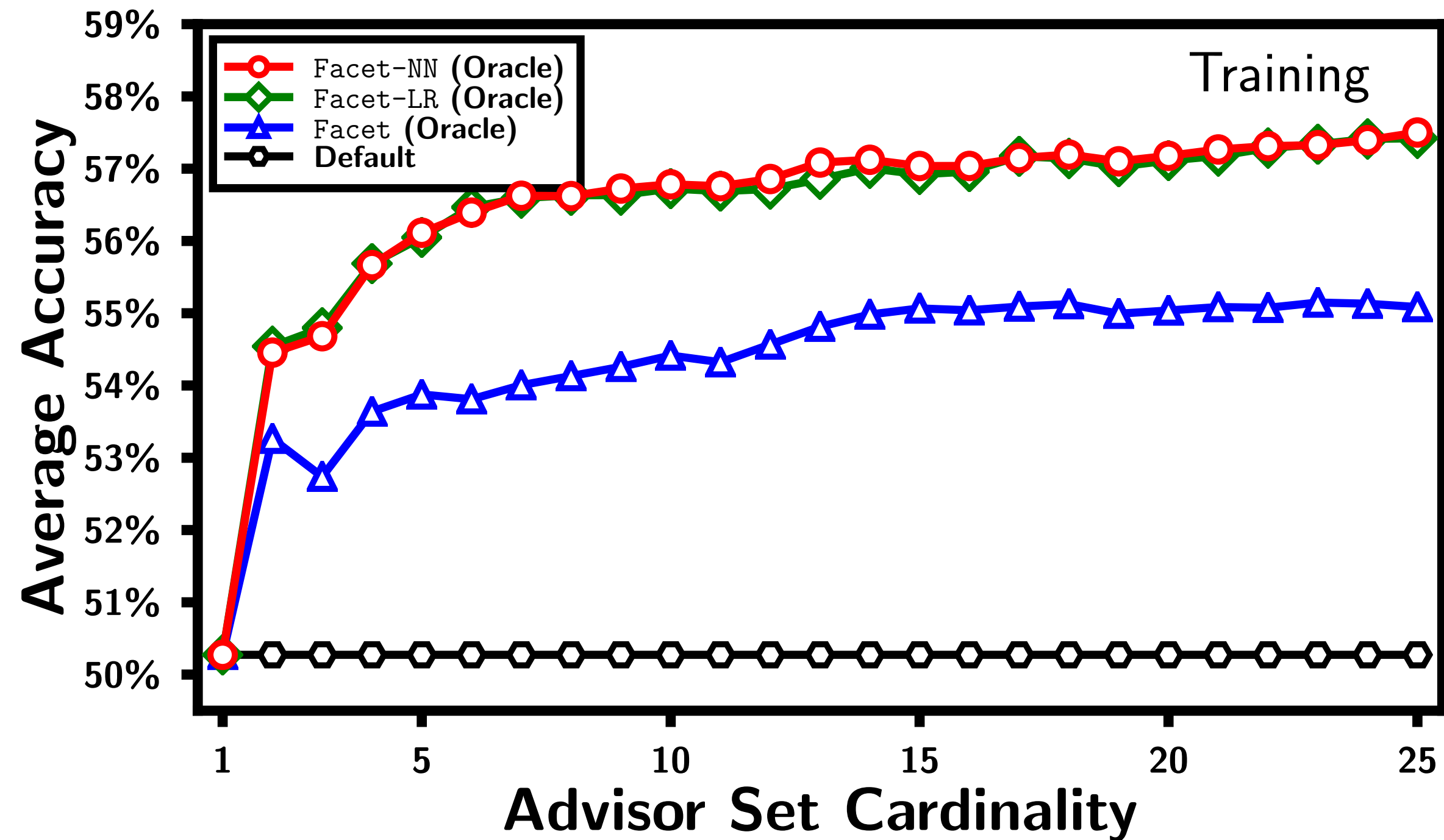


Linear Regression



Modern techniques and larger training also lead to a more accurate linear model.

Advising for Multiple Sequence Alignment



Facet-NN and Facet-LR outperform original Facet on the advising task.

Applying Advising to Transcript Assembly

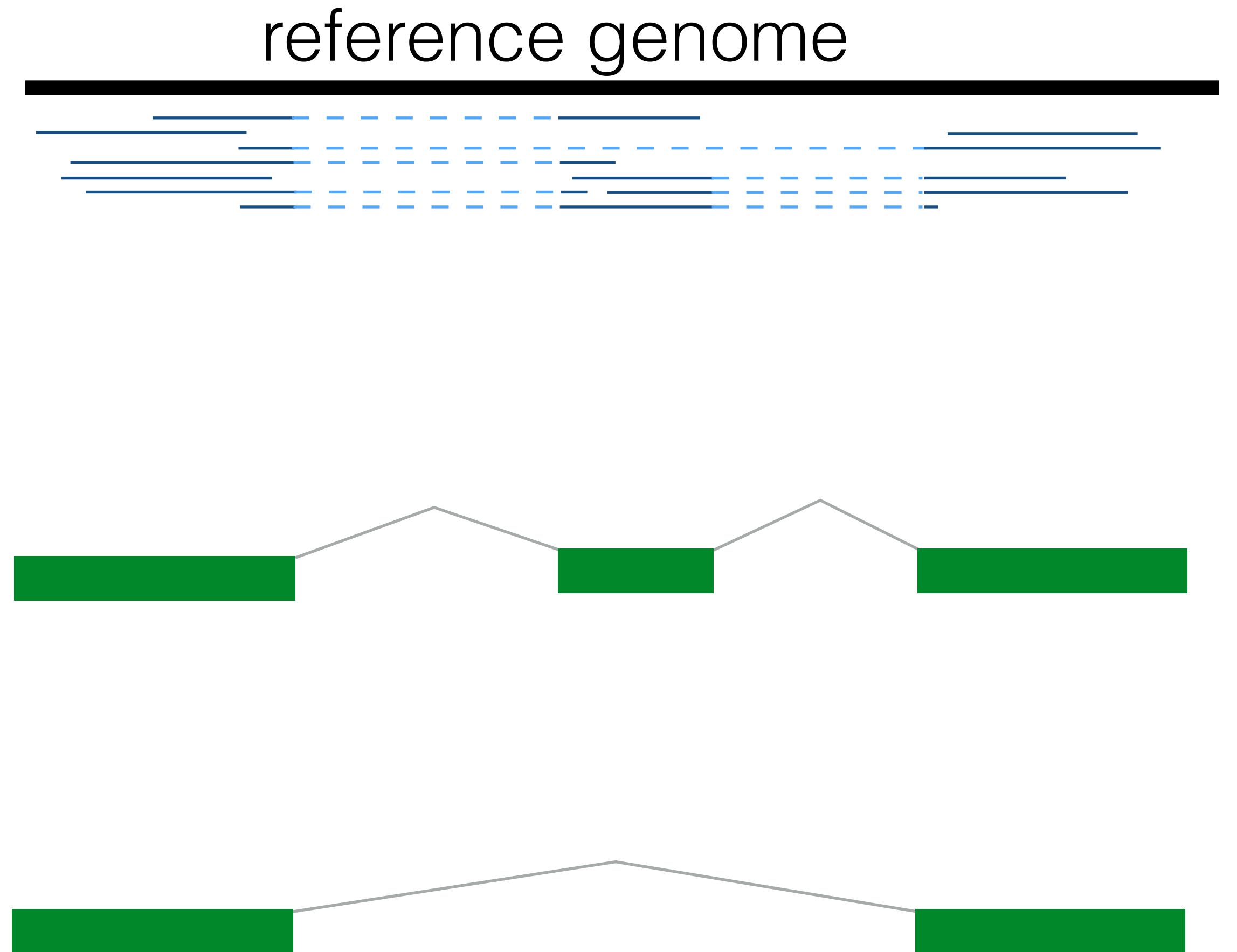
Transcript assembly (TA)

Given

- a set of **RNA-seq reads** aligned to a reference genome, and
- a **set of thresholds** for transcript construction

find:

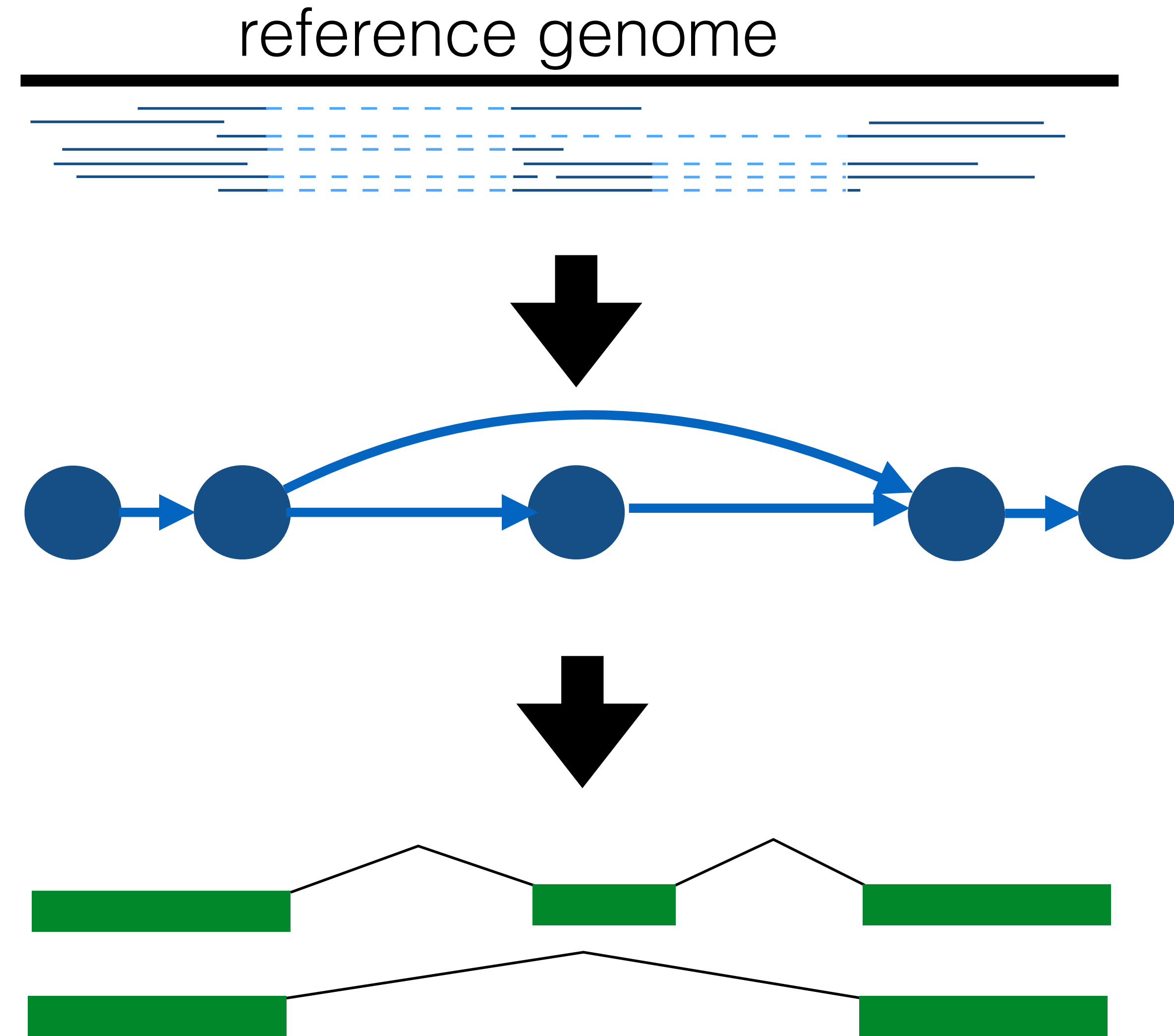
- a set of **constructed transcripts** that explains the reads.



Transcript assembly

A is **fundamental** in transcriptomics.

- It's computationally difficult.
- It's easily impacted by choices of parameter values.
- There is no readily available way to confirm an assembly's accuracy.



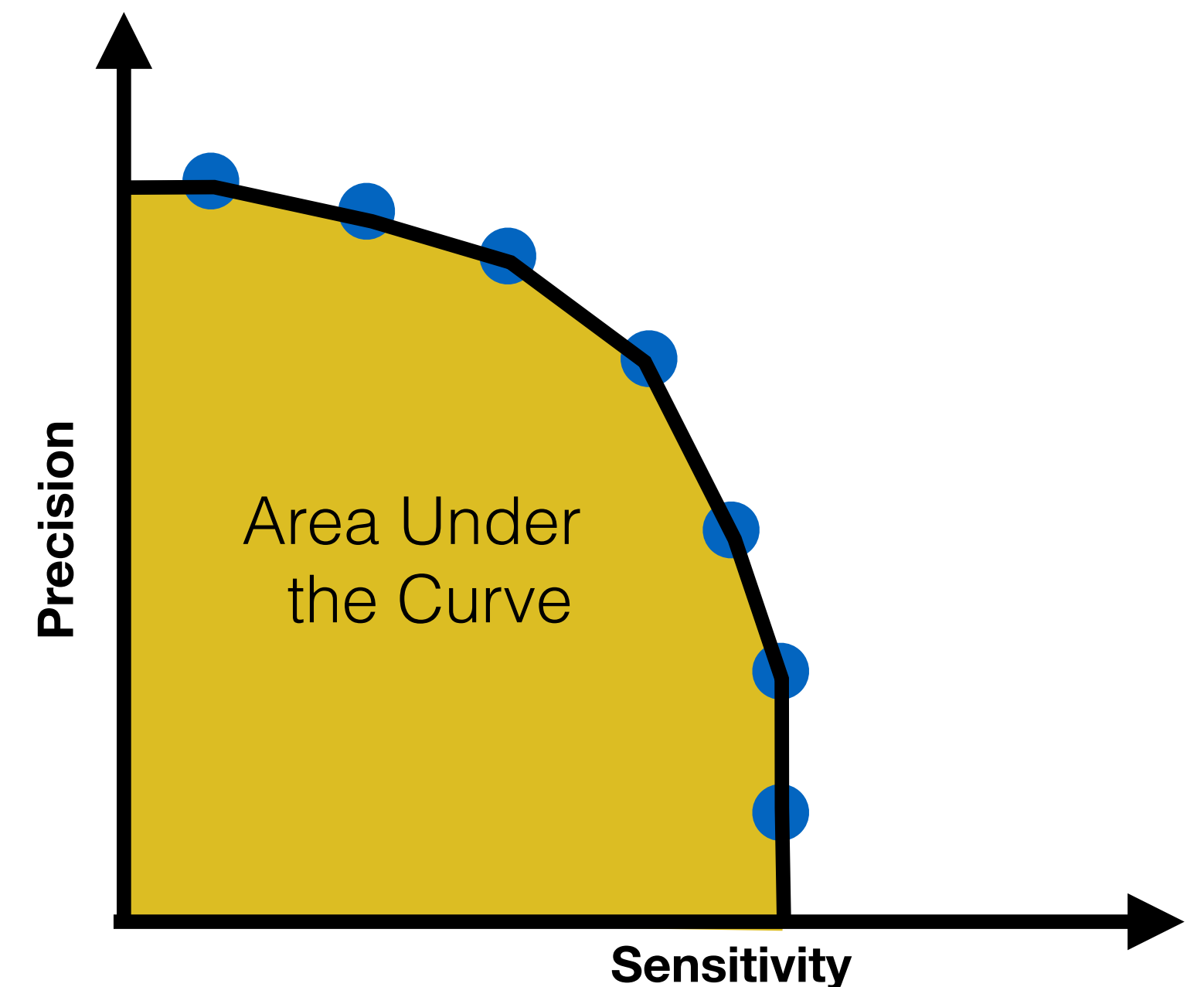
Transcript assembly

For the human genome there is a [reference transcriptome](#).

- Contains a large set of biologically verified transcripts.
- More than will be seen in a single experiment.
- Missing novel transcripts for any given experiment.

[Area Under the Curve \(AUC\)](#) can be calculated using the reference transcriptome.

- Map assembled transcripts to the reference.
- Threshold the quality score from the assembler to get precision/sensitivity.
- Commonly used to compare assembler quality.



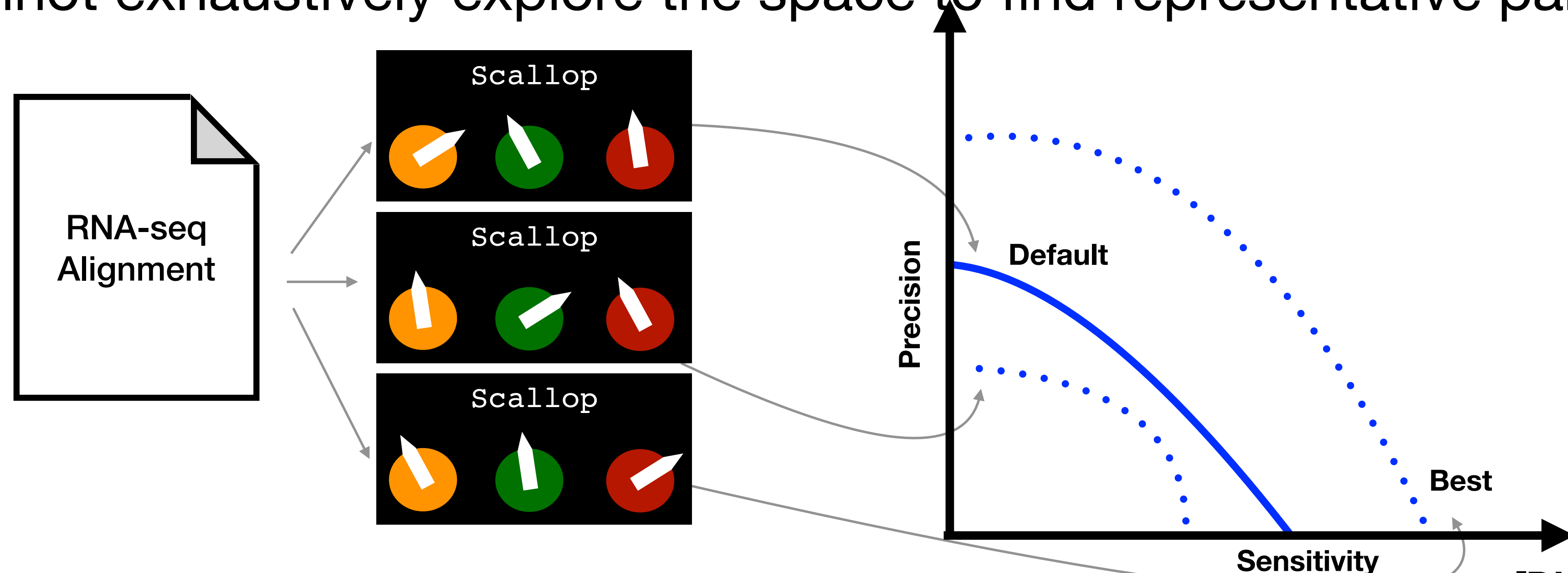
Transcript assembly advising

Advisor estimator:

- area under the curve

Advisor set:

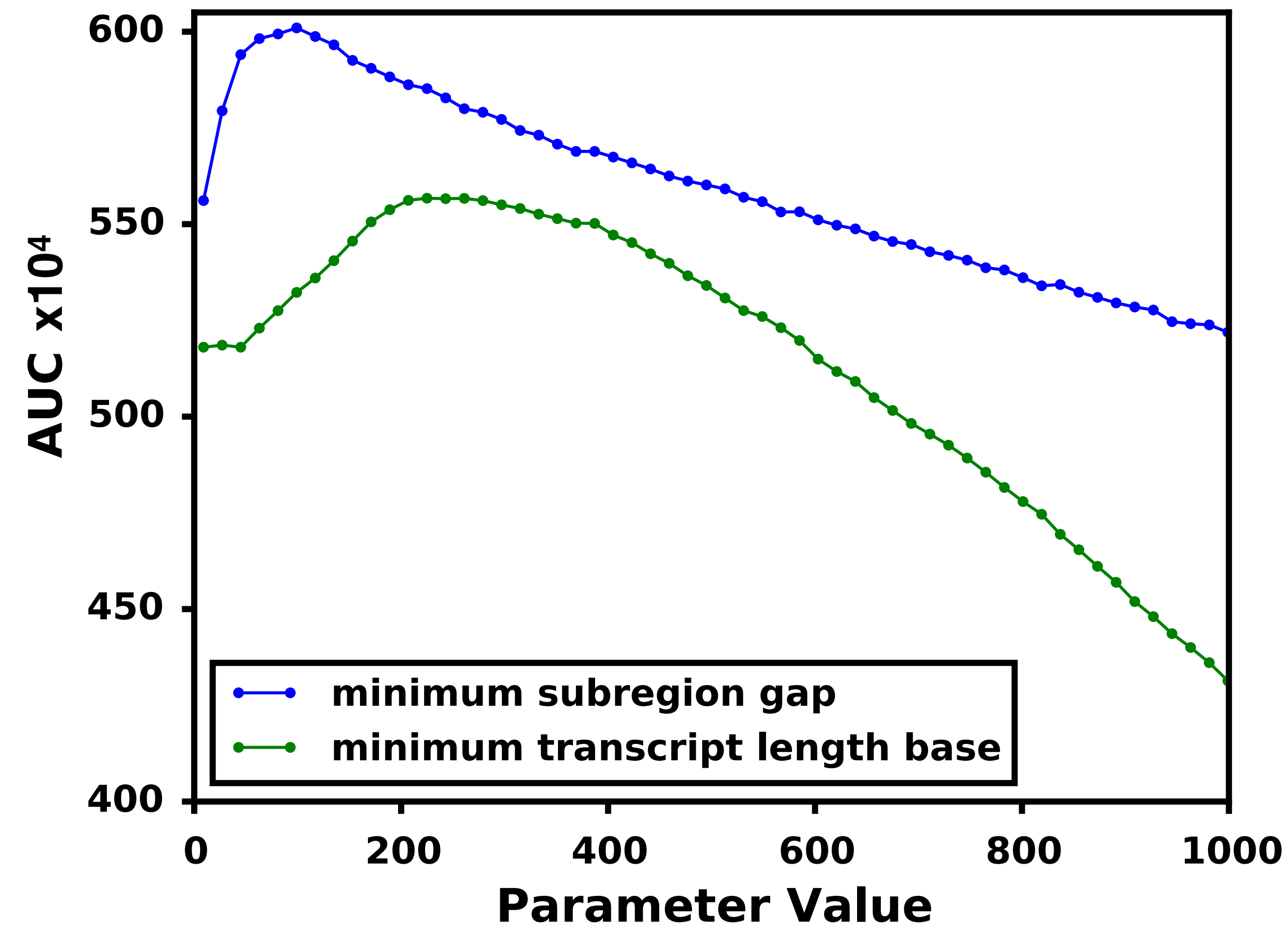
- the number of tunable parameters is very large
- cannot exhaustively explore the space to find representative parameter vectors



Finding an advisor set

Use information about parameter behavior to guide advisor set construction.

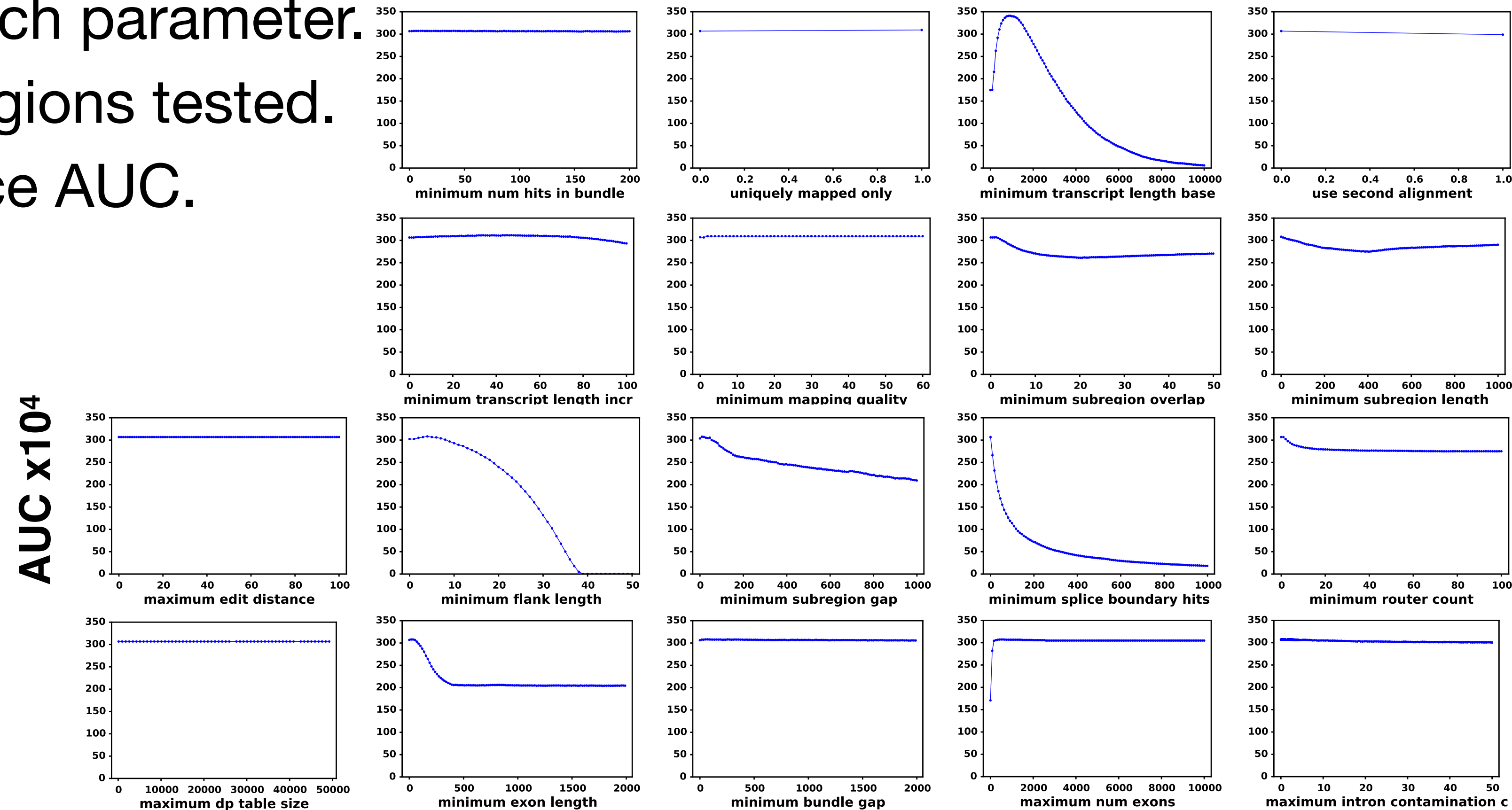
- Tested the influence of each parameter.
- Single maximum in the regions tested.



Finding an advisor set

Use information about parameter behavior to guide advisor set construction.

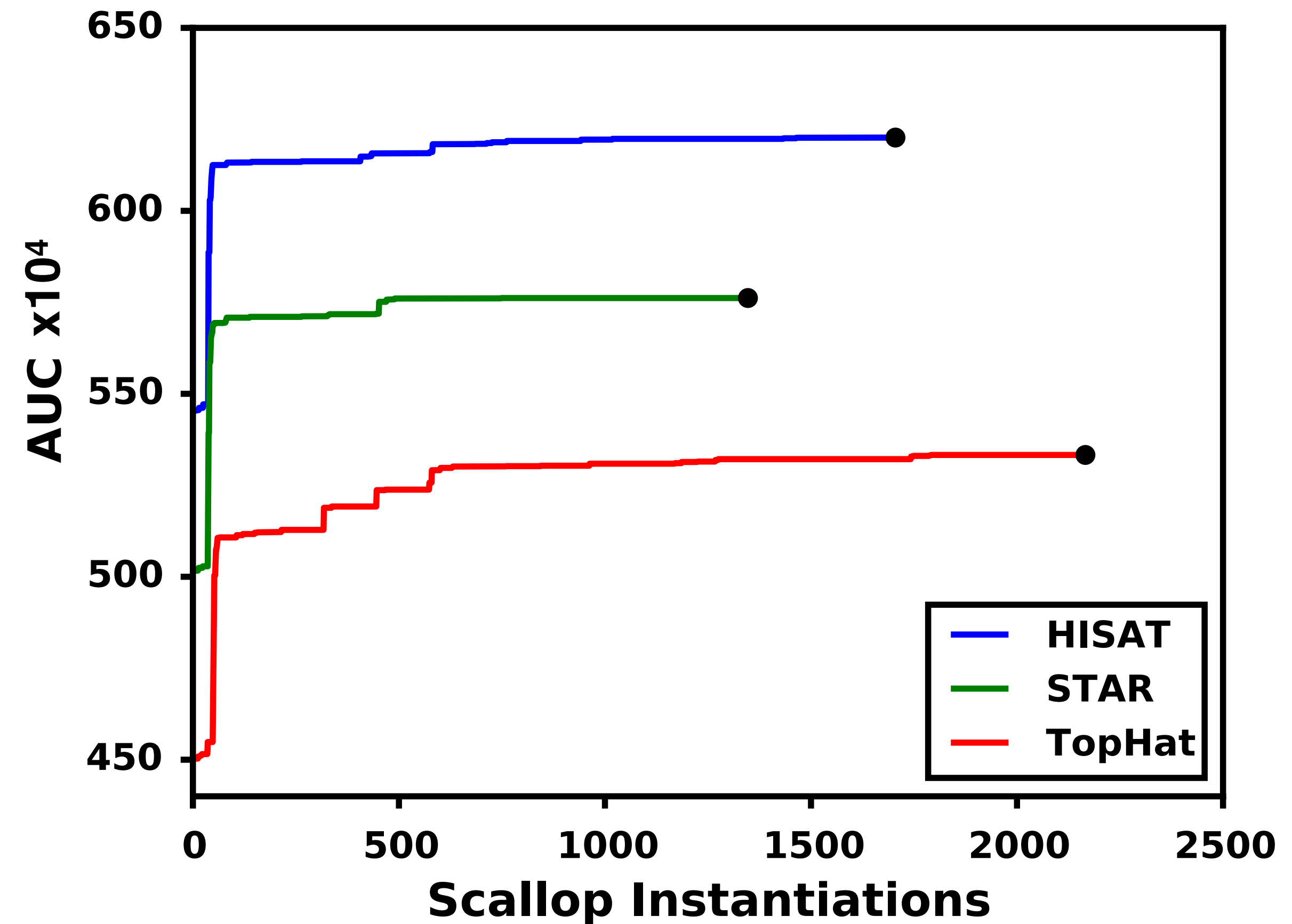
- Tested the influence of each parameter.
- Single maximum in the regions tested.
- Many parameters influence AUC.



Finding an advisor set

Parameter curve smoothness and single maxima help parameter selection.

- Iterative optimization will work well.
- Process is slow.

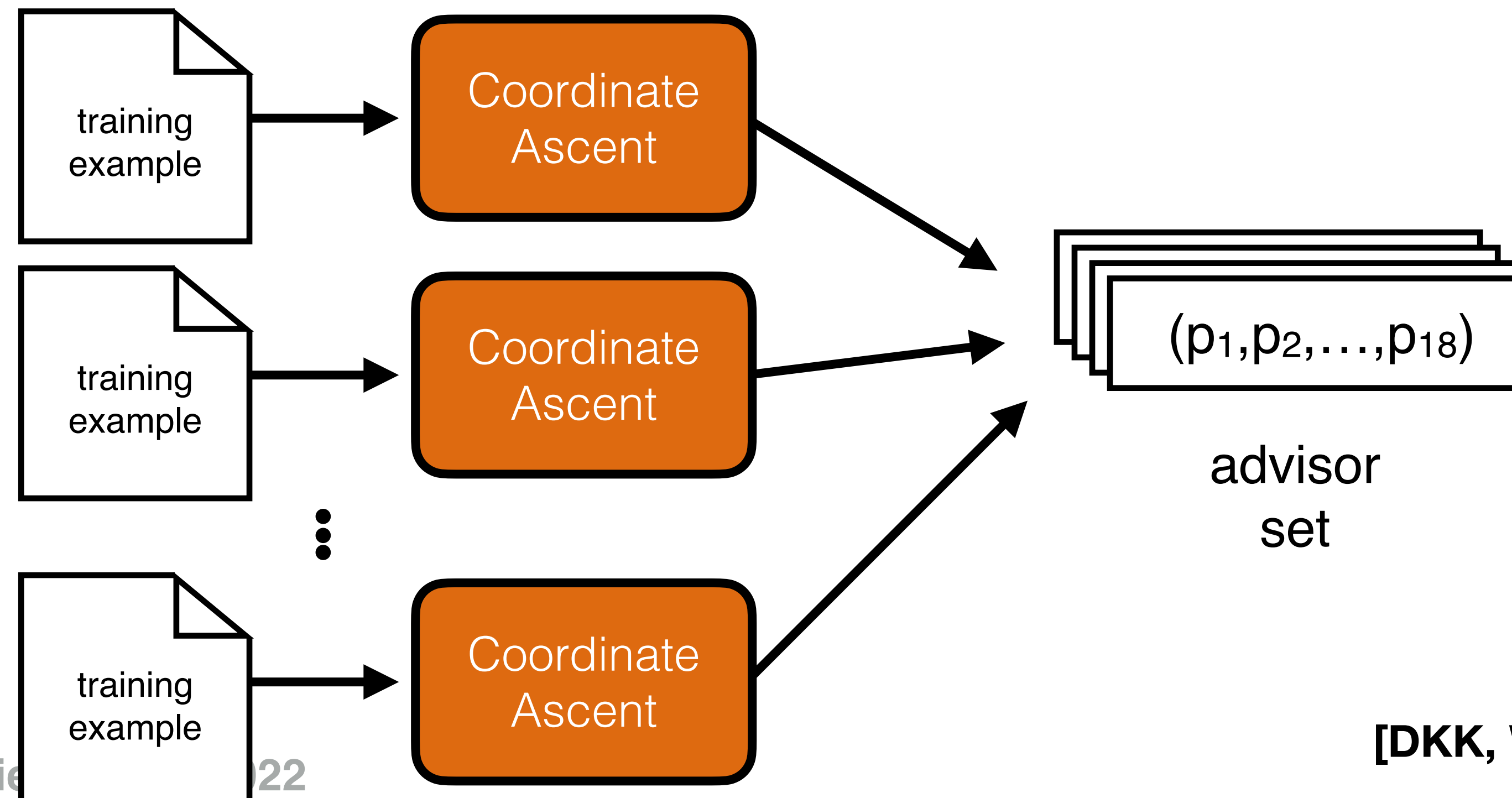


[DKK, WCB@ICML 2019]

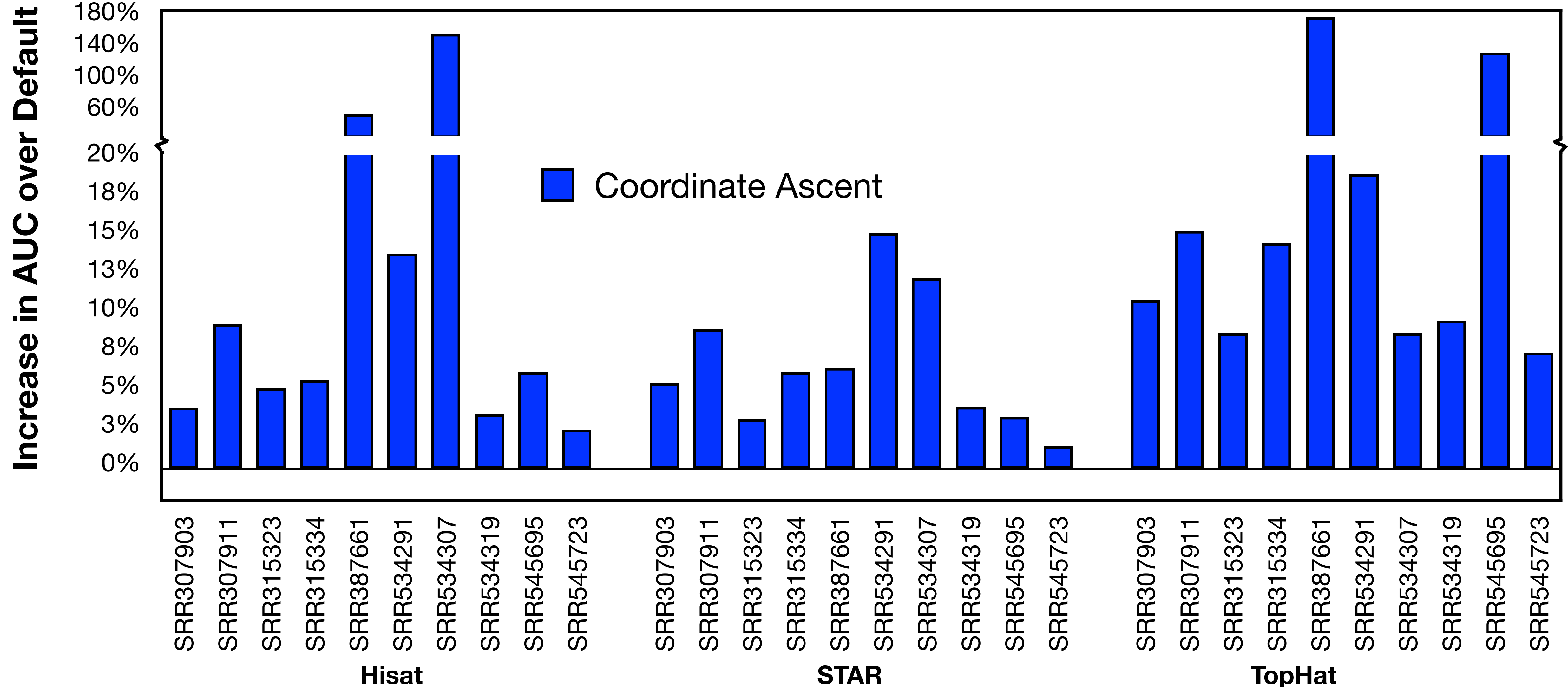
Finding an advisor set

We can use **coordinate ascent** to find optimal parameter vectors.

- Training samples should cover the range of expected input.
- Settings are found for all 18 tunable parameters.
- Collection of produced vectors is advisor set.
- The set is precomputed and doesn't impact the advising time.



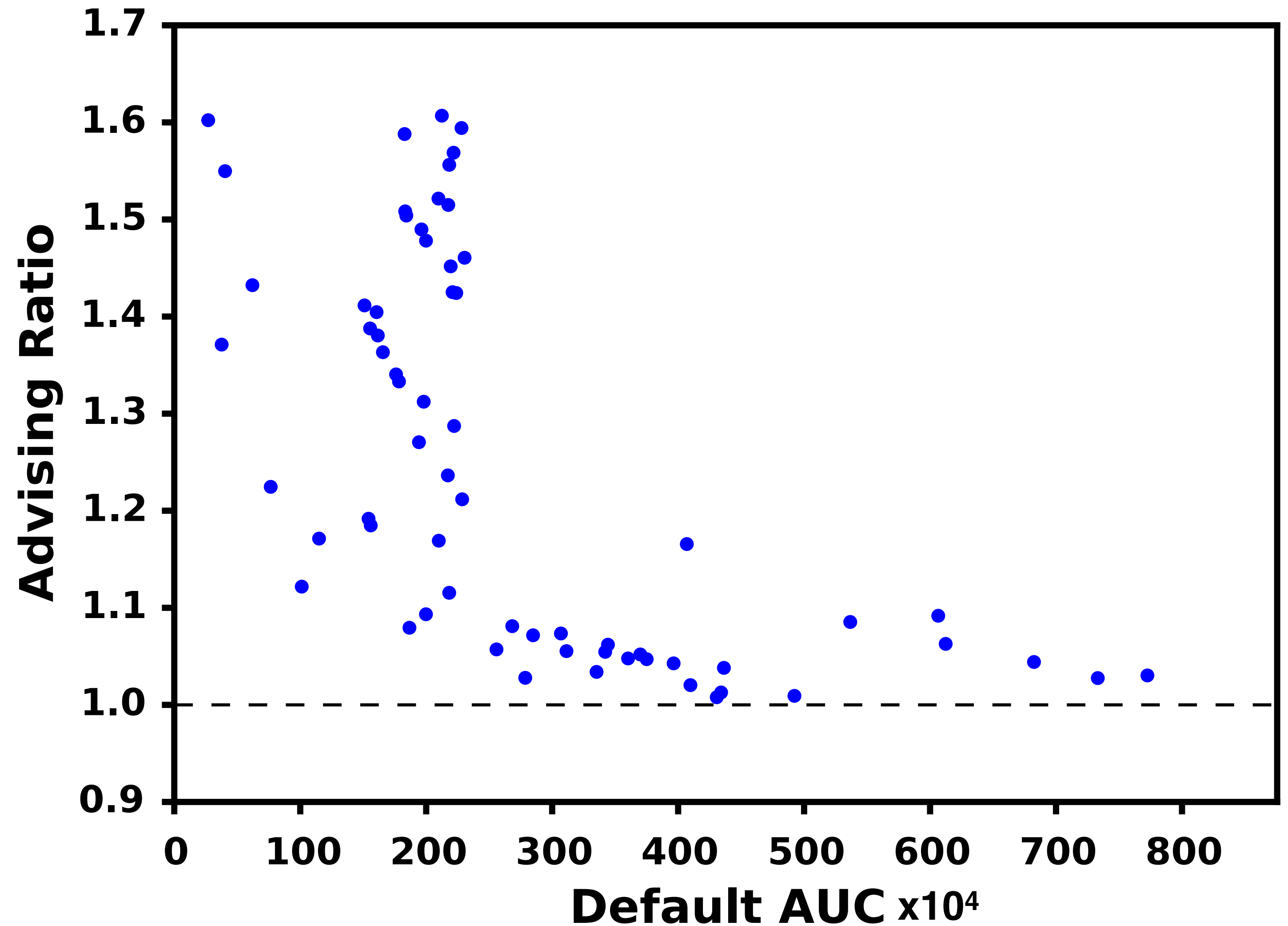
Scallop advising



Average of 18.1% increase in AUC using Coordinate Ascent

Scallop advising

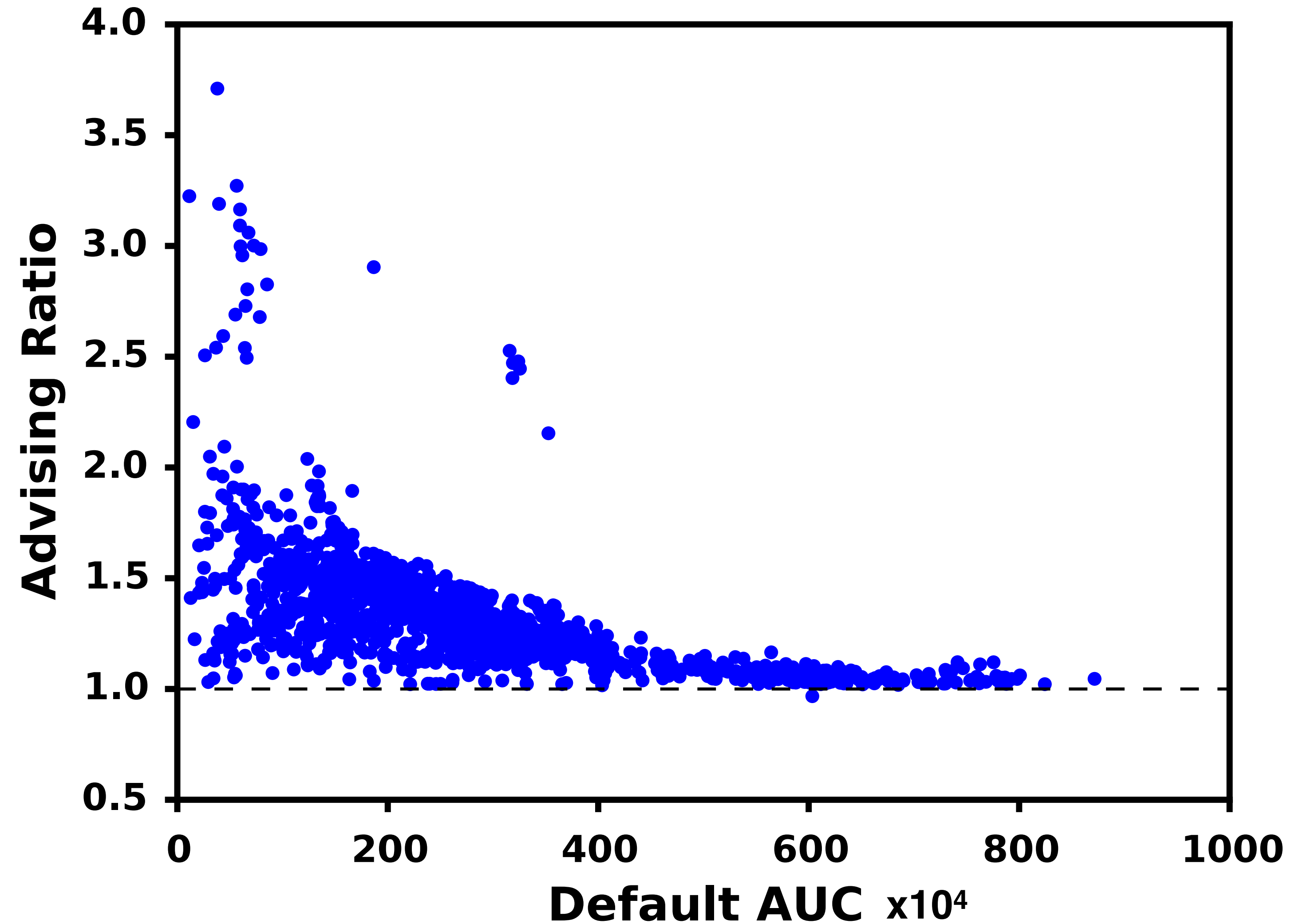
- all aligned RNA-seq from ENCODE
- variety of aligners
- example of performance in general



average advising ratio: 1.257

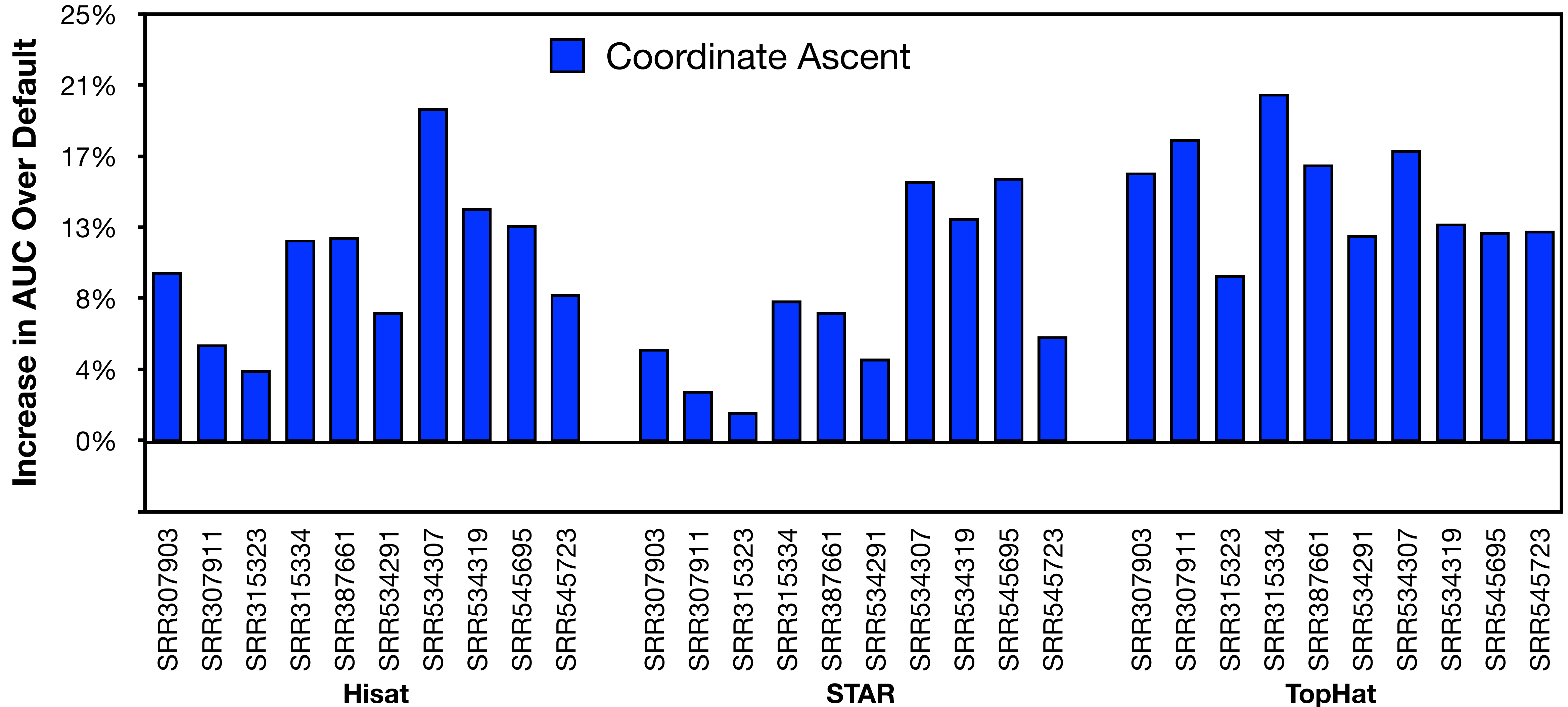
Scallop advising

- 1595 RNA-Seq from SRA
- aligned using STAR
- example of high-throughput performance



average advising ratio: 1.382

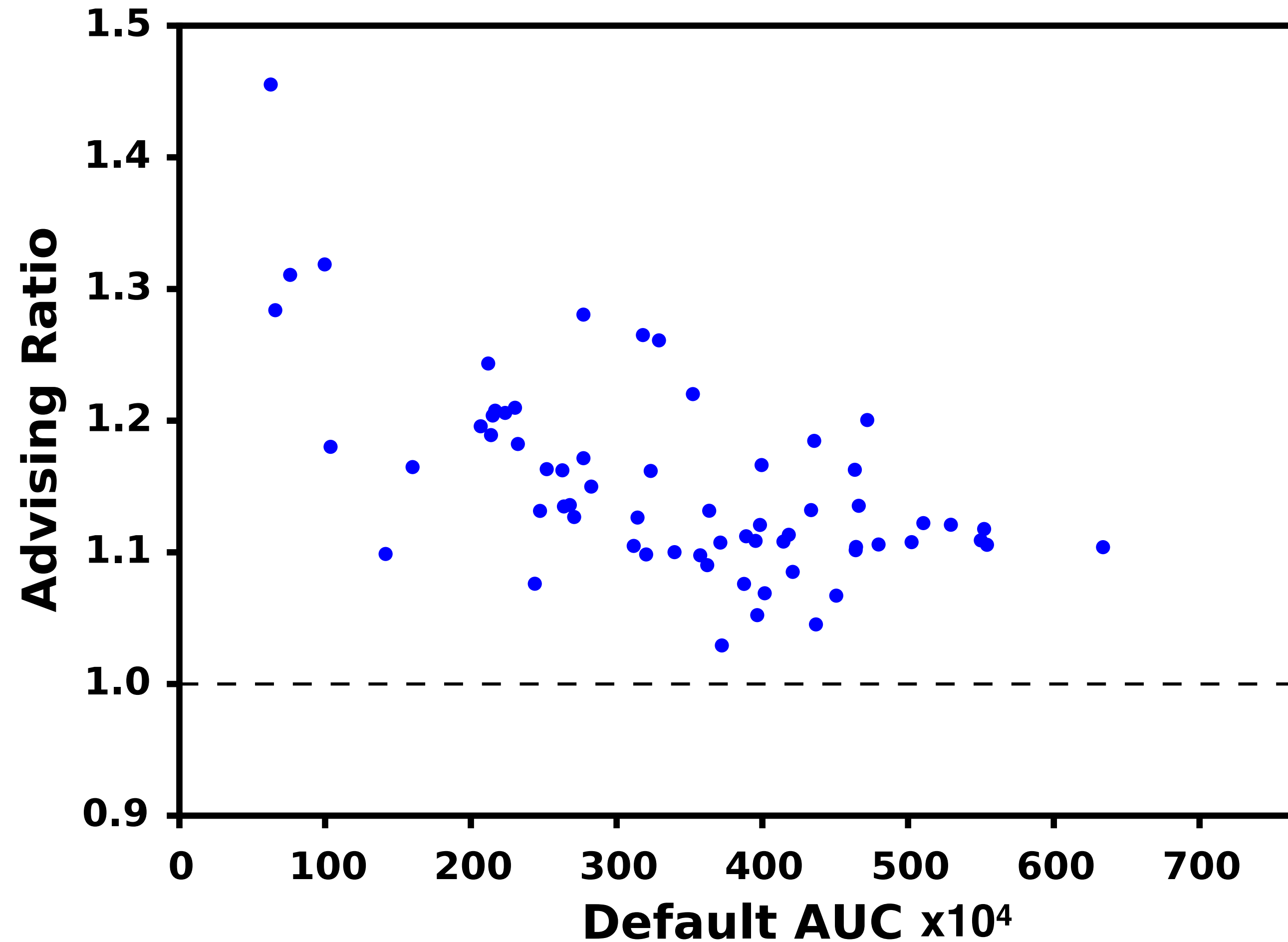
StringTie advising



11.1% average increase in accuracy for Coordinate Ascent

StringTie advising

ENCODE 65 Dataset



- all aligned RNA-seq from ENCODE
- variety of aligners
- example of performance in general

average advising ratio: 1.151

Transcript assembly advising

Parameter advising **increases AUC** for transcript assembly.

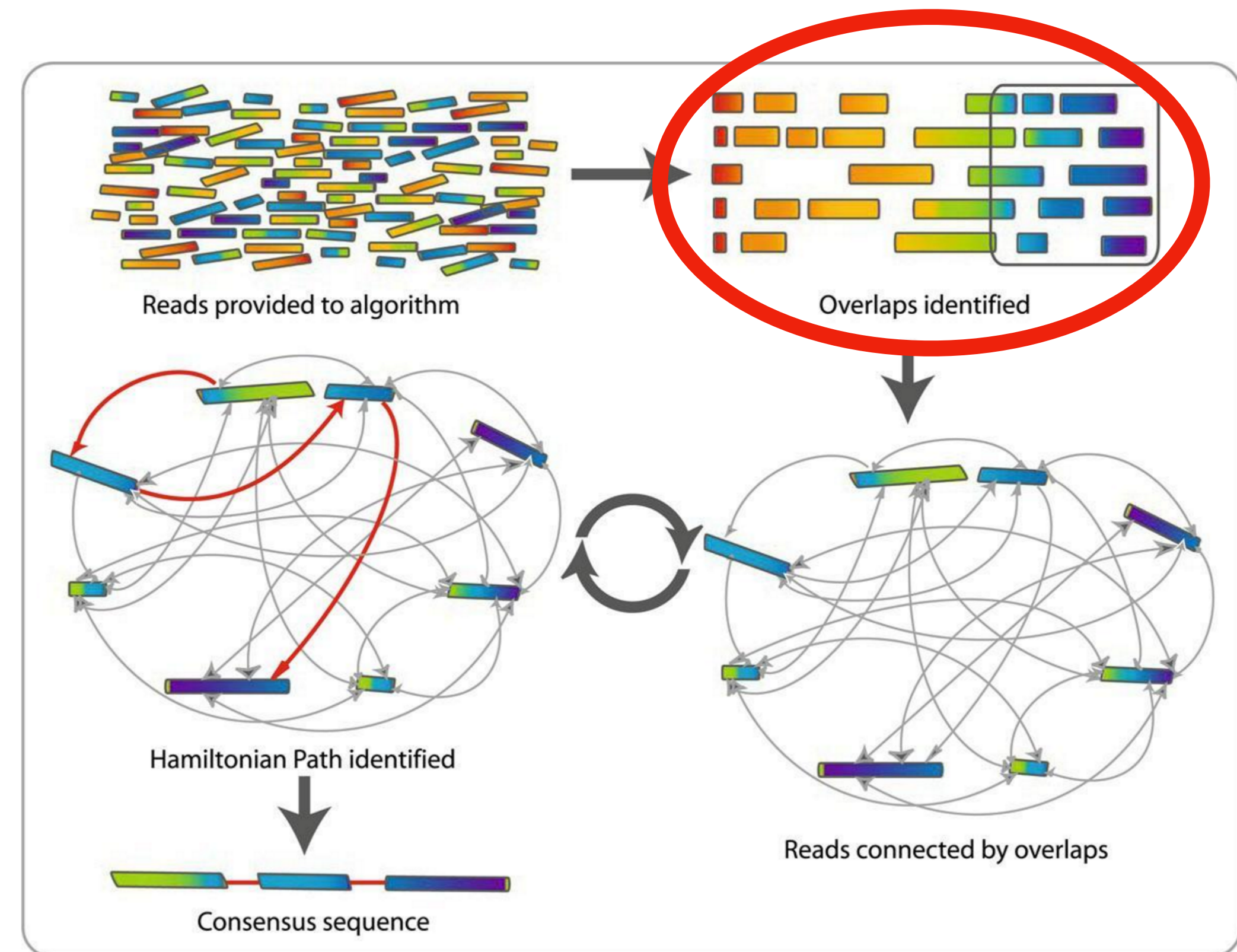
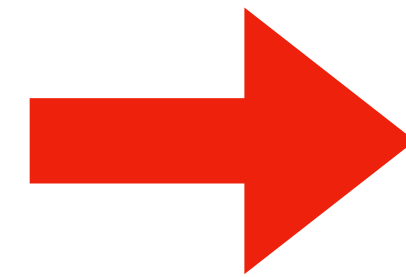
- **Coordinate ascent** is useful to advisor sets.
- Improvements are seen for both **Scallop** and **StringTie**.

Minimizer Schemes for Genome Analysis

Sequence Similarity

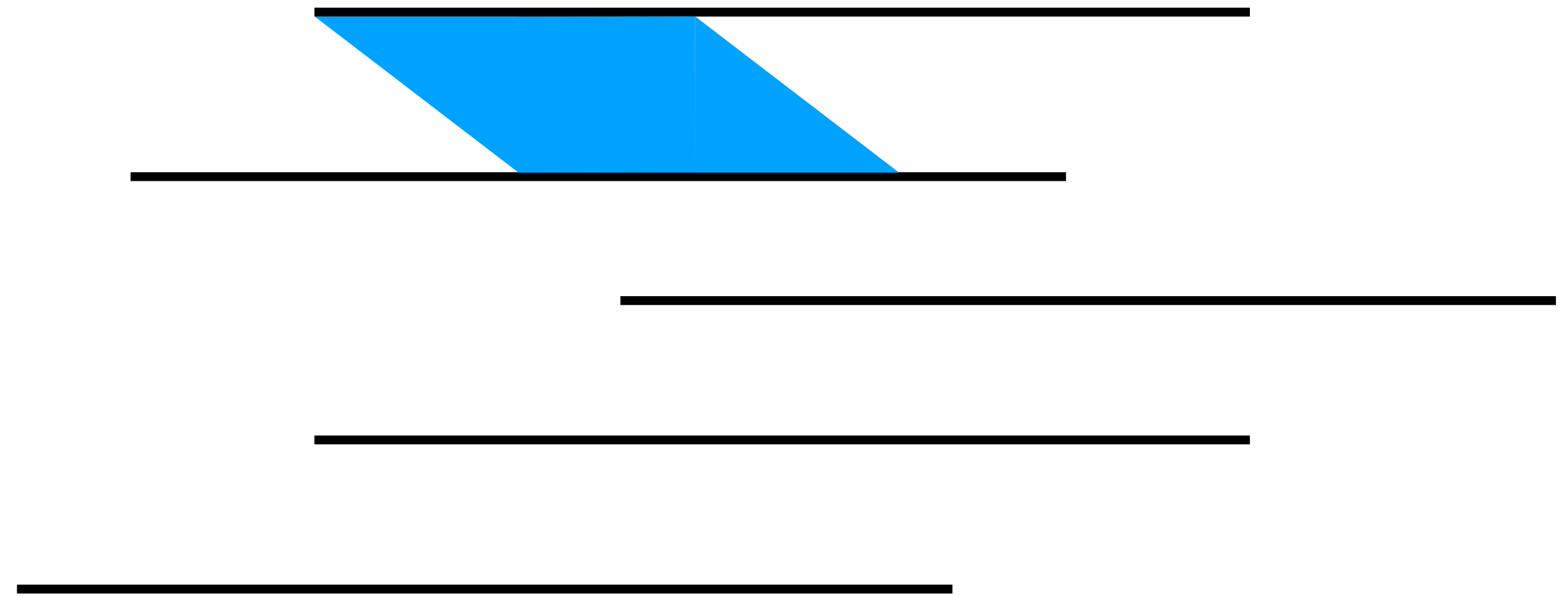
Sequence similarity is used in many contexts:

- comparing web pages
- suggestion systems
- finding plagiarism
- matching sequencing reads
- binning genetic material



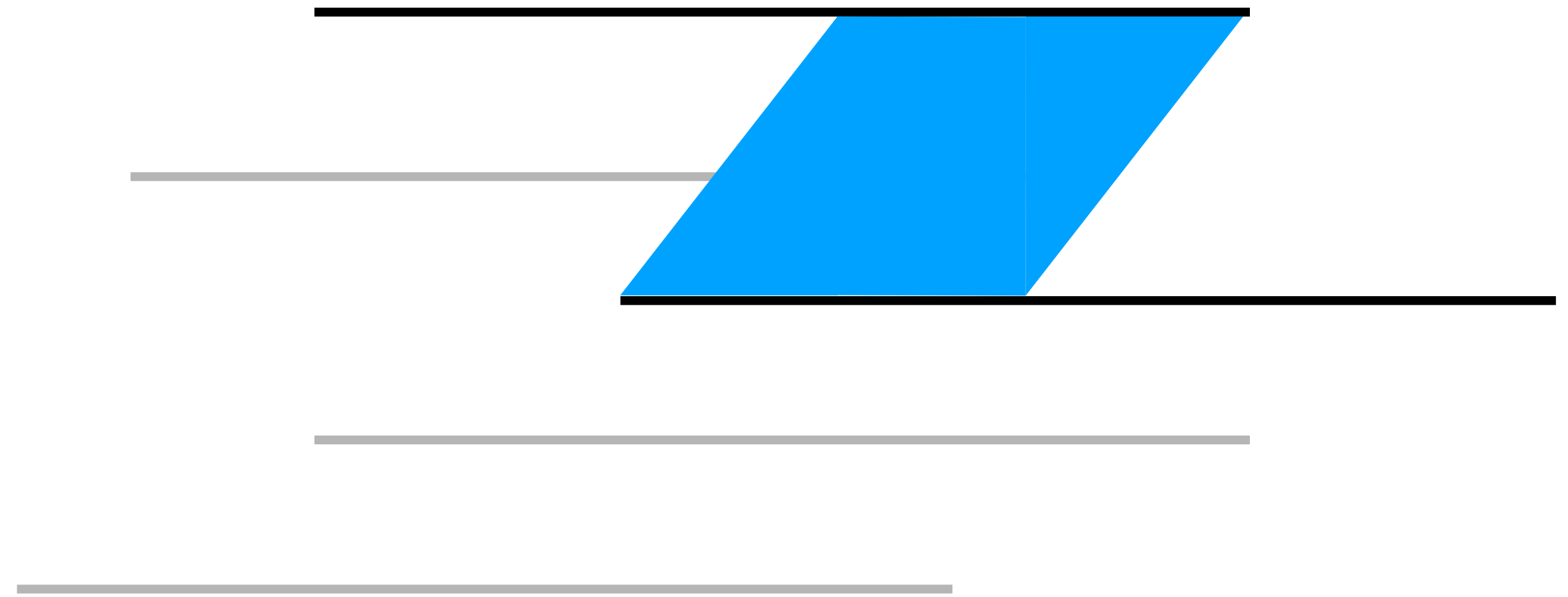
Minimizer Schemes

Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation



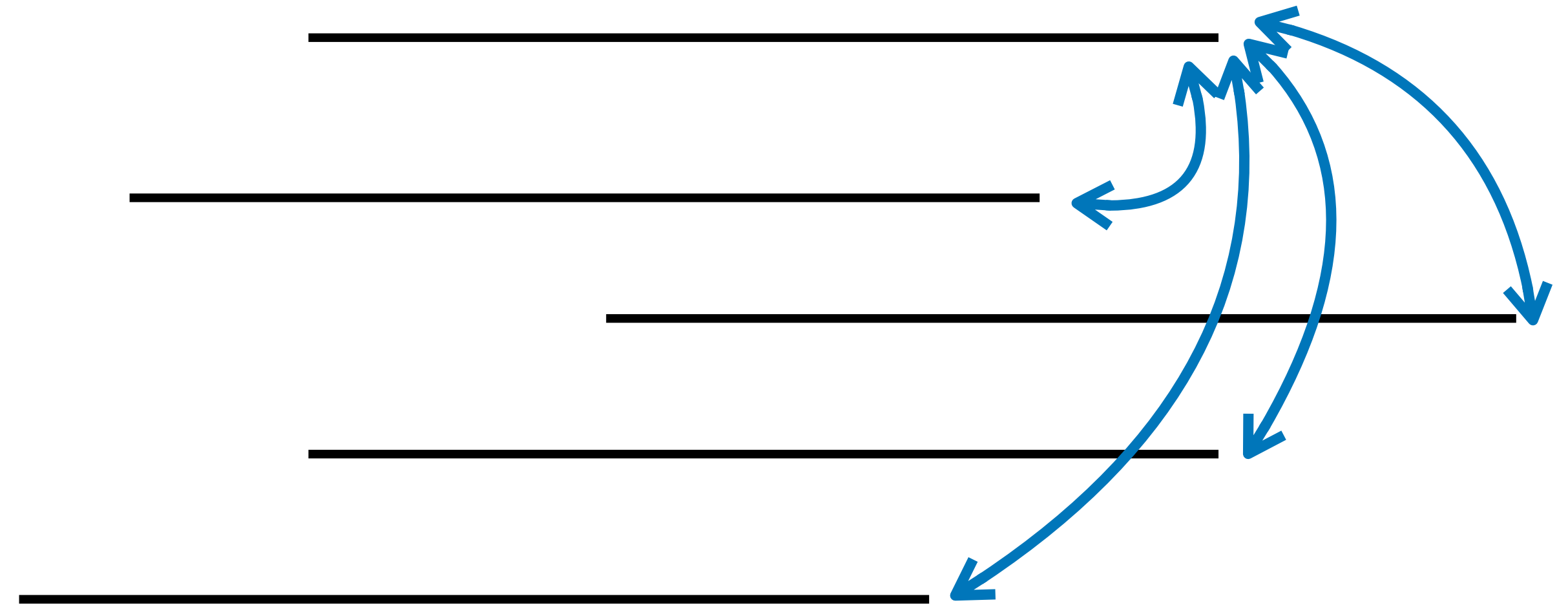
Minimizer Schemes

Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation



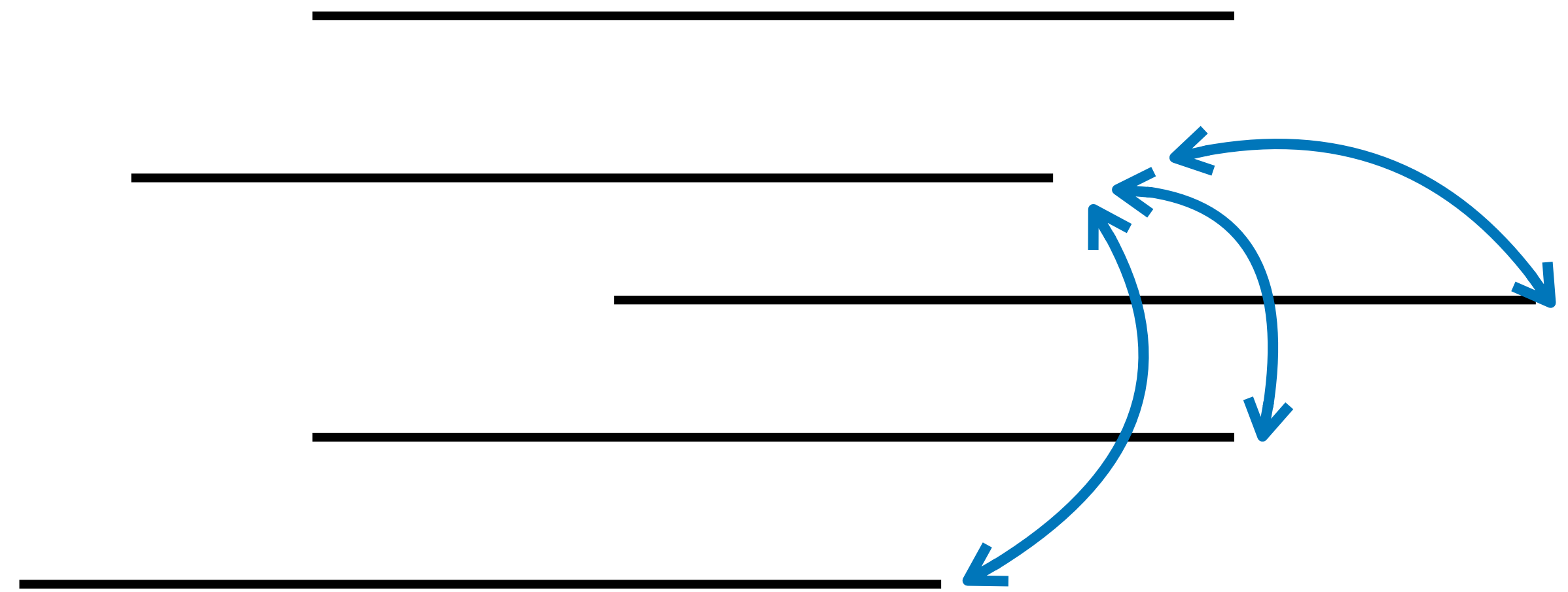
Minimizer Schemes

Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation



Minimizer Schemes

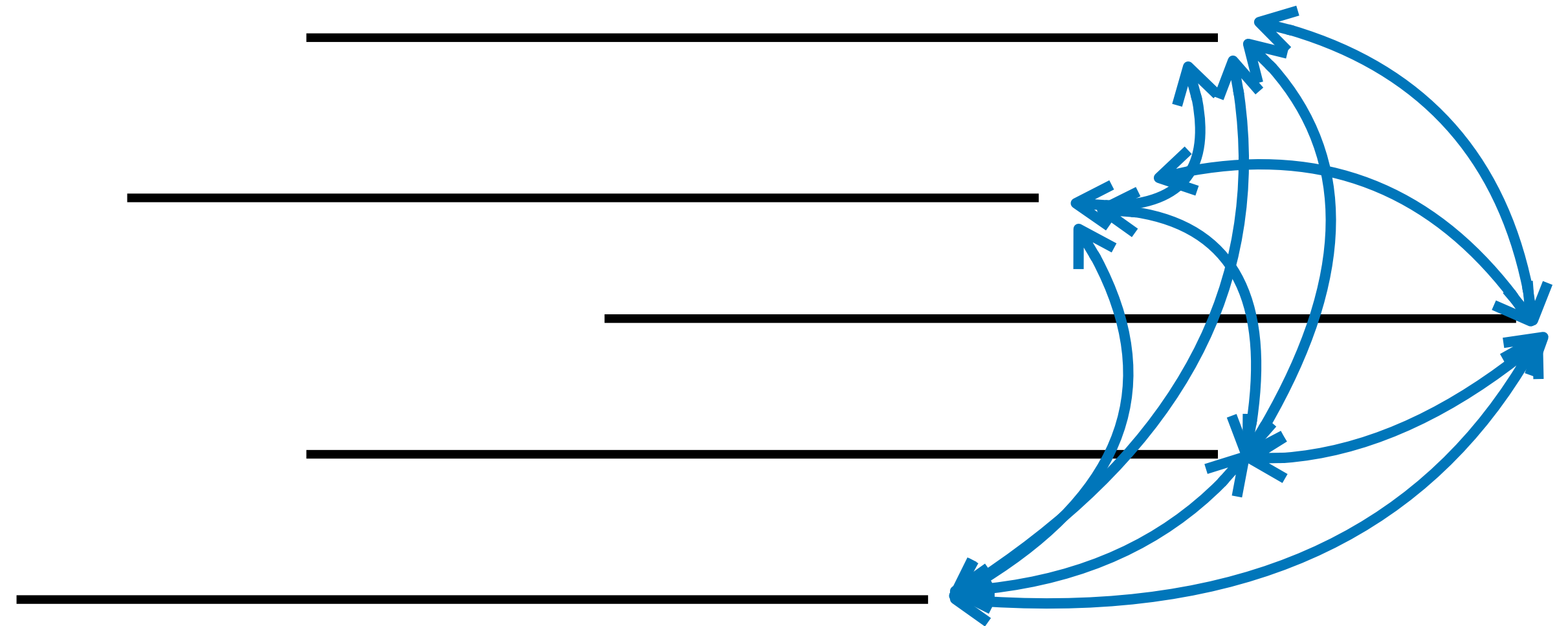
Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation



Minimizer Schemes

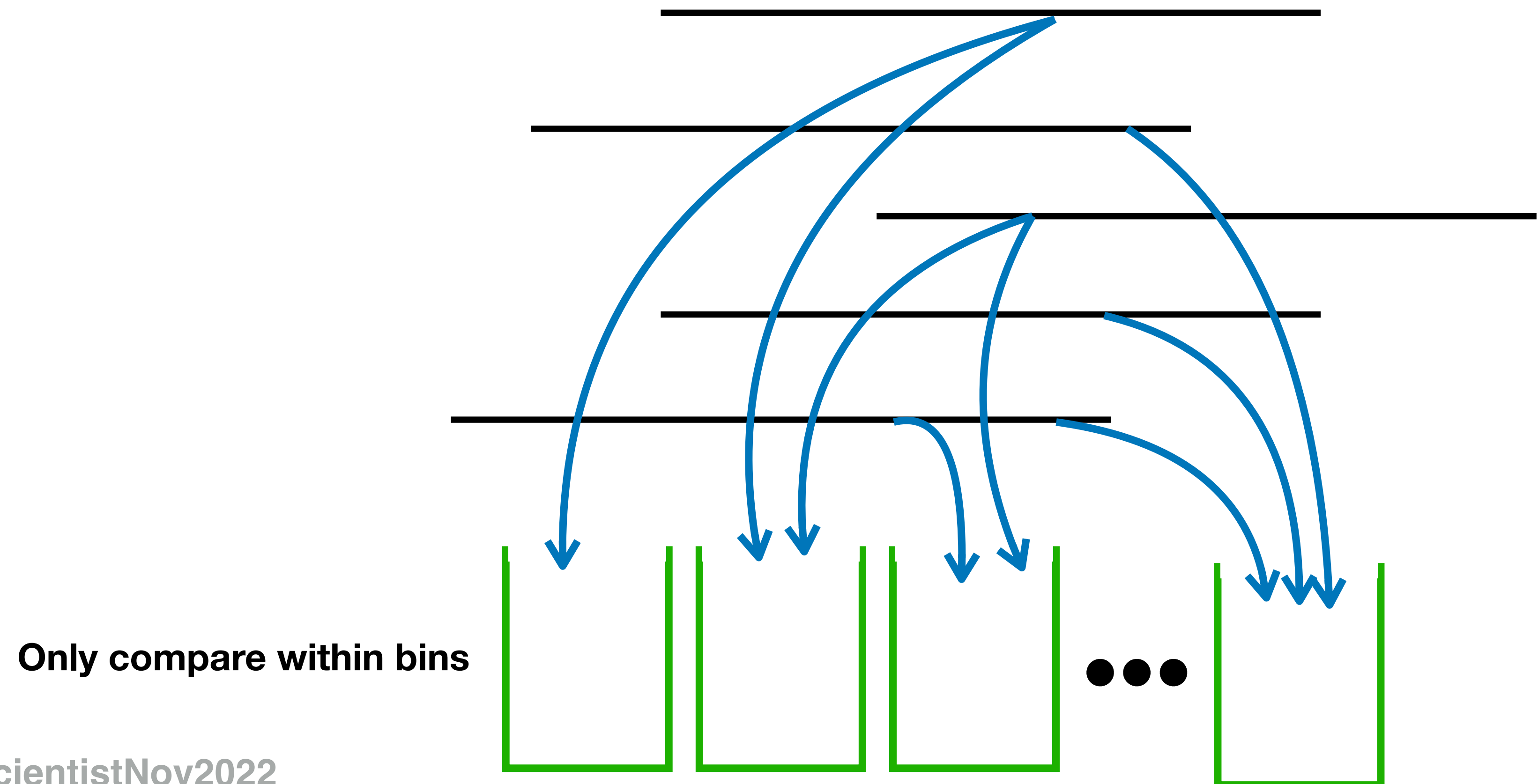
Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation

$O(n^2)$ alignments!



Minimizer Schemes

Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation



Minimizer Schemes

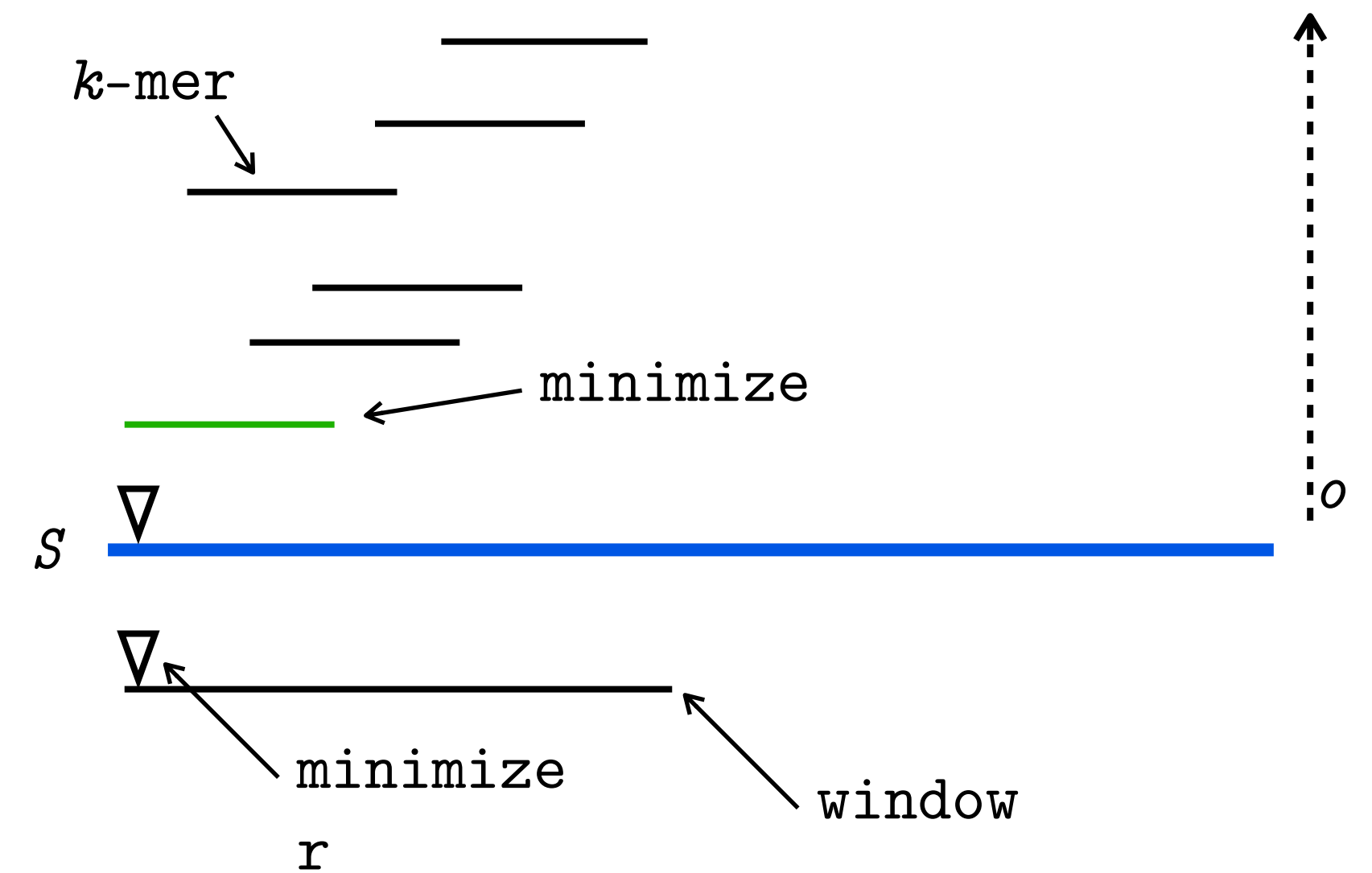
Minimizer schemes have two special properties:

- two sequences with a long exact match must select the same k -mers
- there are no large gap between selected k -mers

Used in k -mer counting, *de Bruijn* graph construction, data structure sparsification, etc.

Minimizer Schemes

For a windows of w consecutive k -mers from a sequence S , a minimizer scheme selects the minimum according to an ordering o as a representative



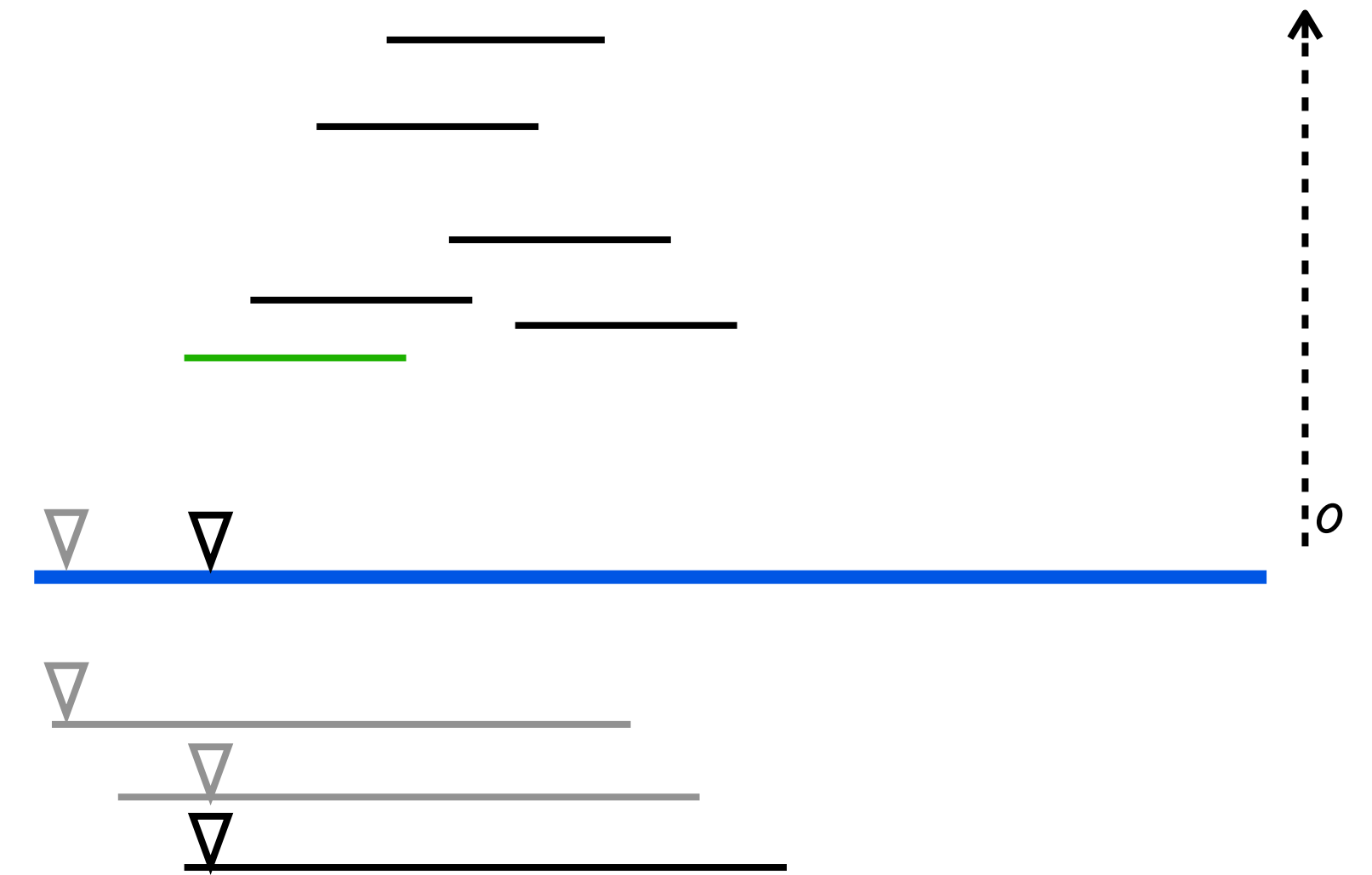
Minimizer Schemes

For a windows of w consecutive k -mers from a sequence S , a minimizer scheme selects the minimum according to an ordering o as a representative



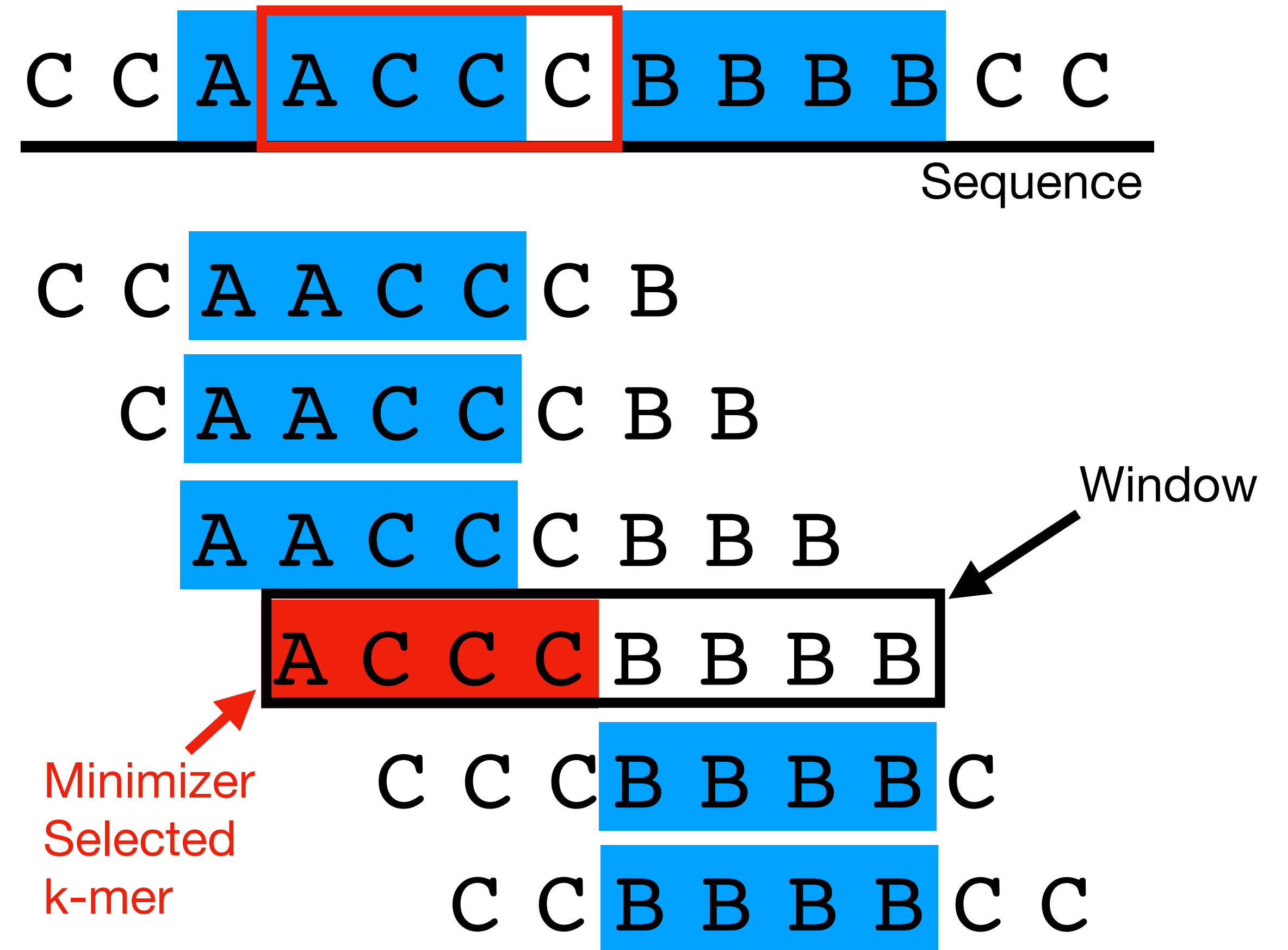
Minimizer Schemes

For a windows of w consecutive k -mers from a sequence S , a minimizer scheme selects the minimum according to an ordering o as a representative



Minimizer Schemes

An extra example

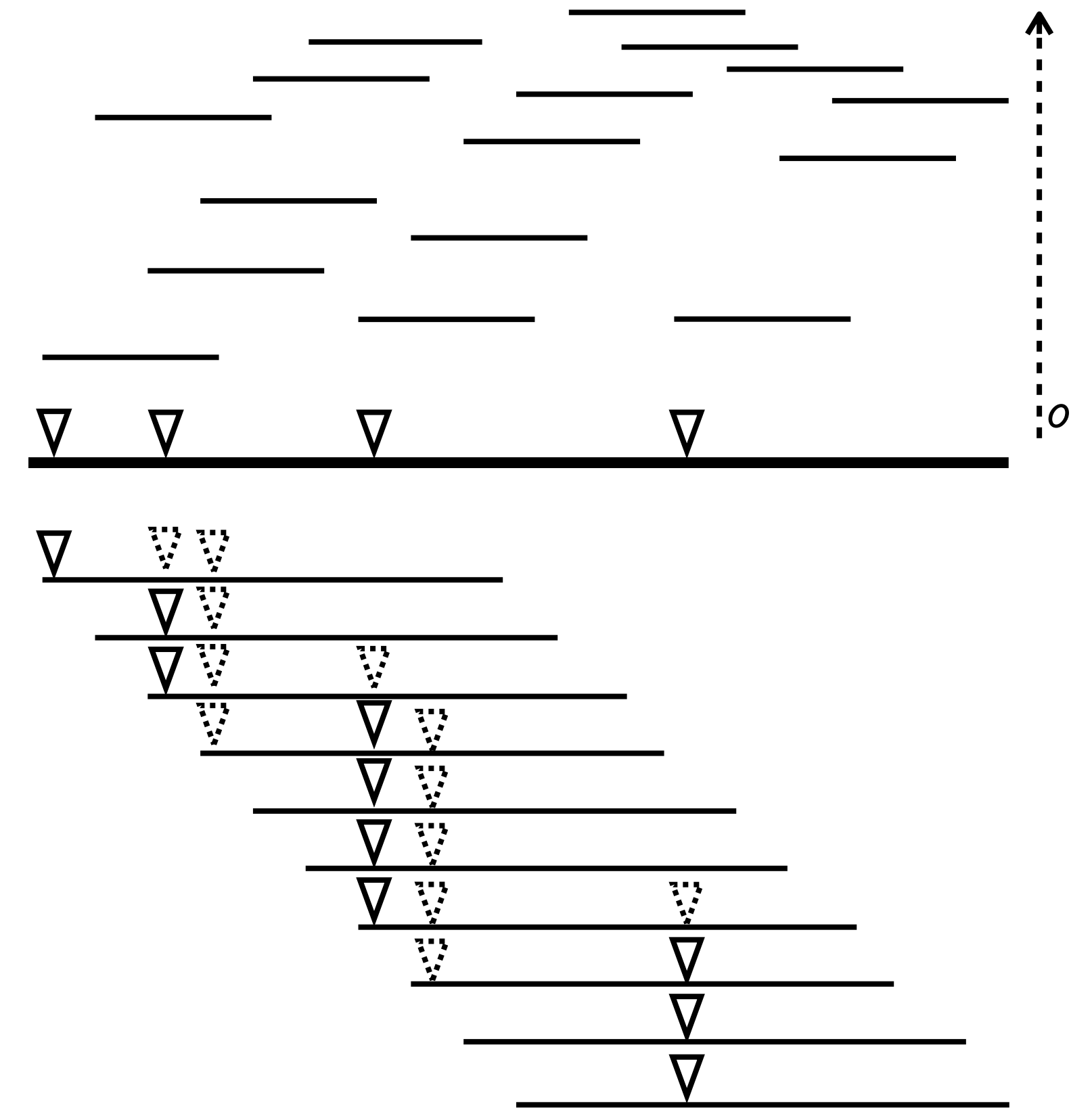


Minimizer Schemes

The ordering can impact how well the minimizer scheme performs.

We measure performance using **density**:

- Normalized count of minimizer locations in S

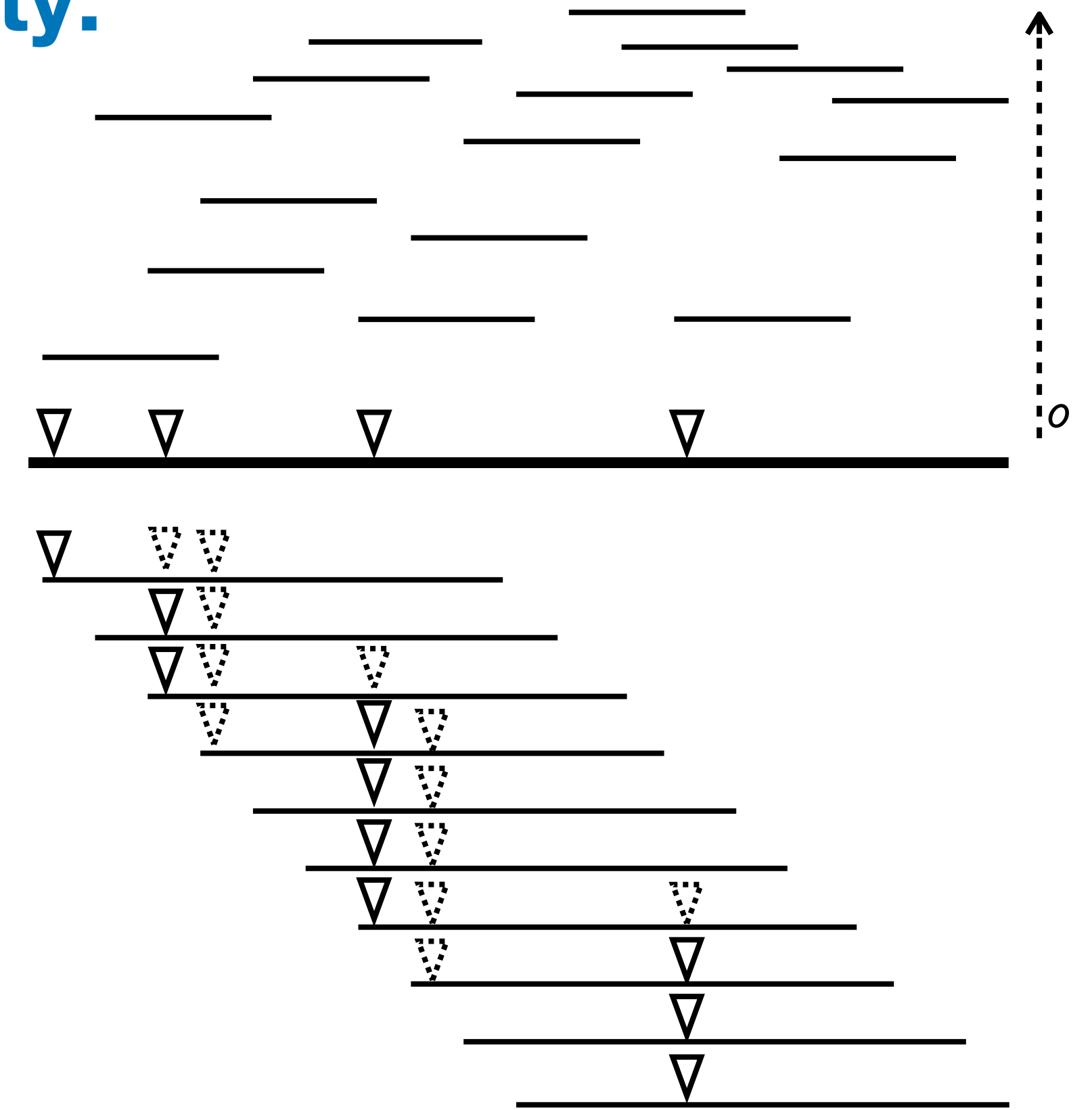


Minimizer Schemes

The ordering can impact how well the minimizer scheme performs.

We measure performance using **expected density**:

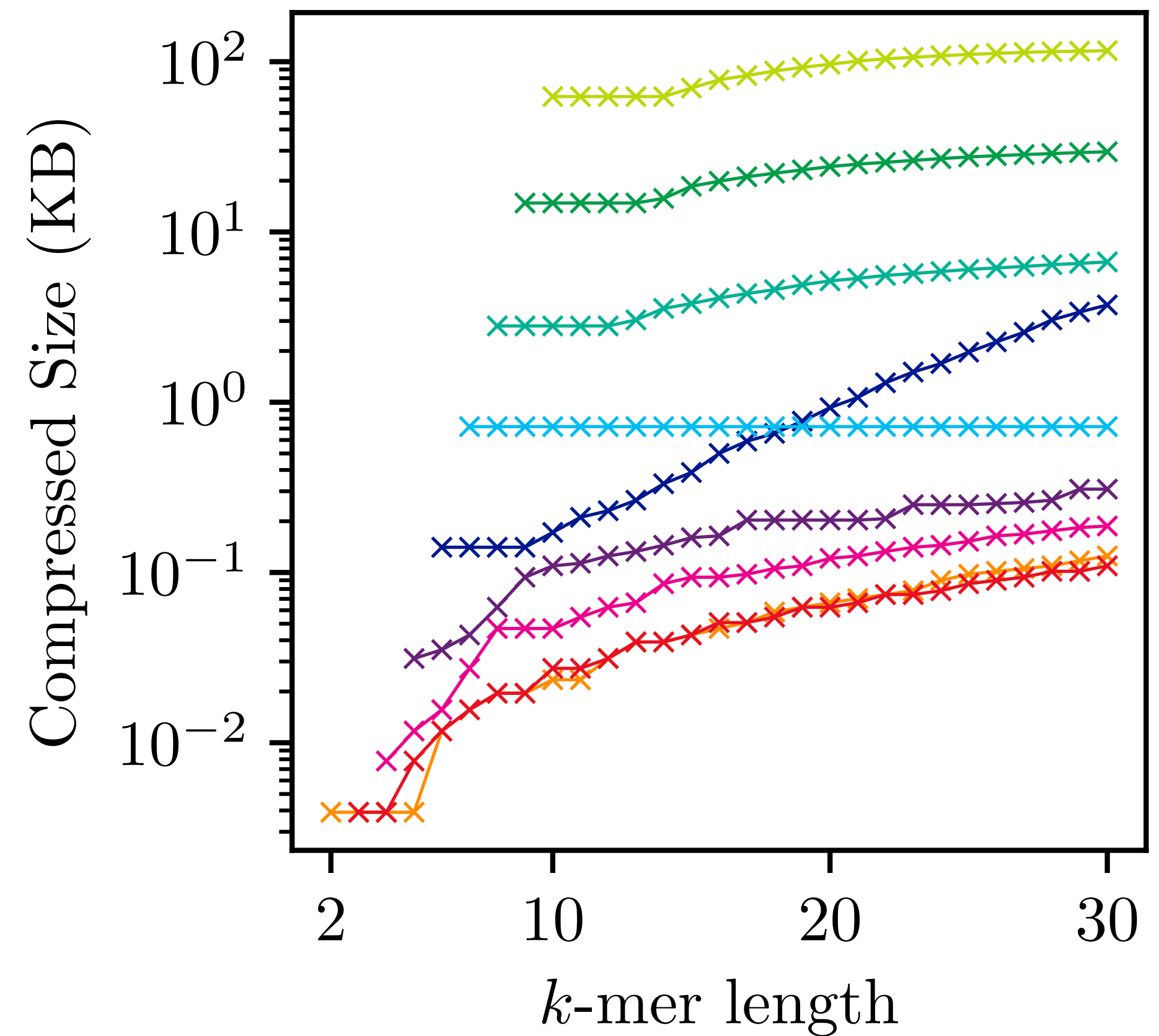
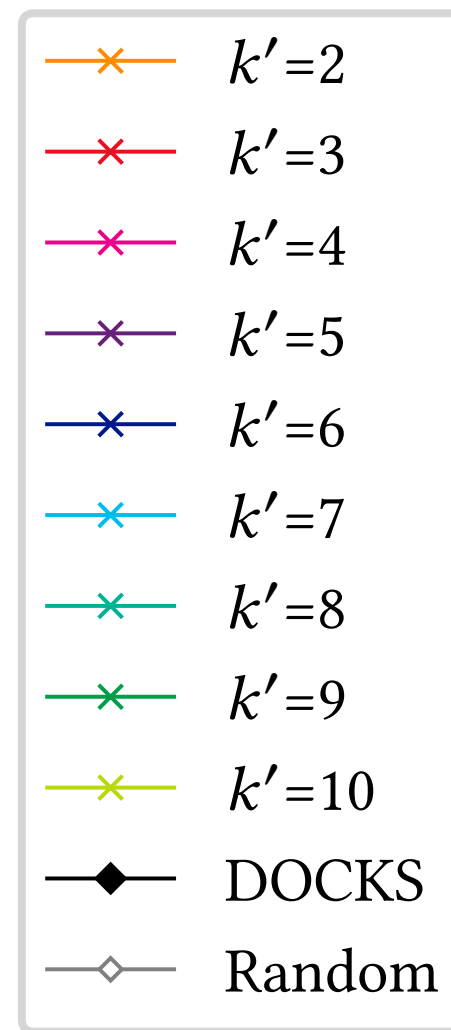
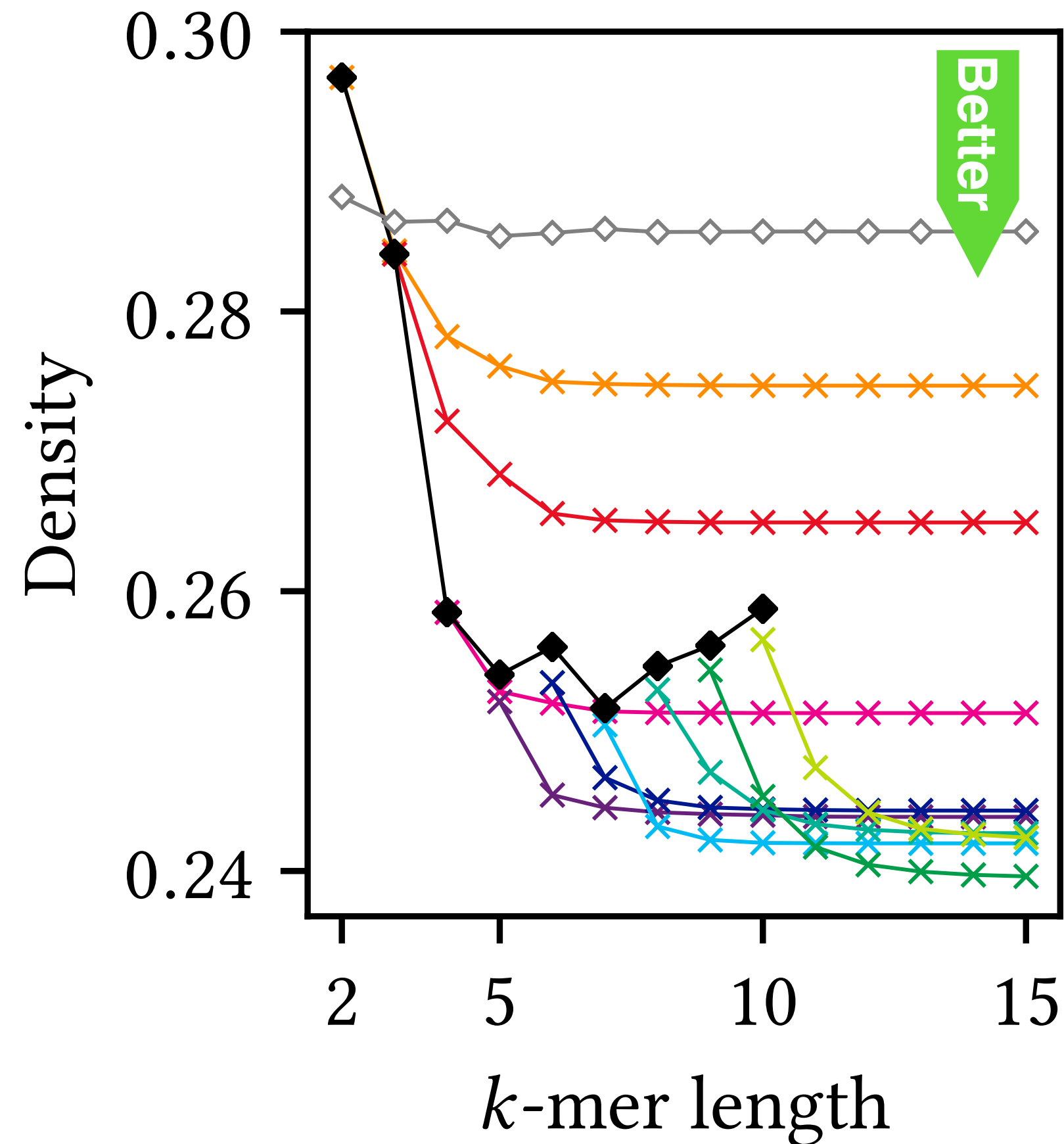
- Normalized count of minimizer locations in B_L



B_L is the **de Bruijn** sequence of order L , it contains each window exactly once

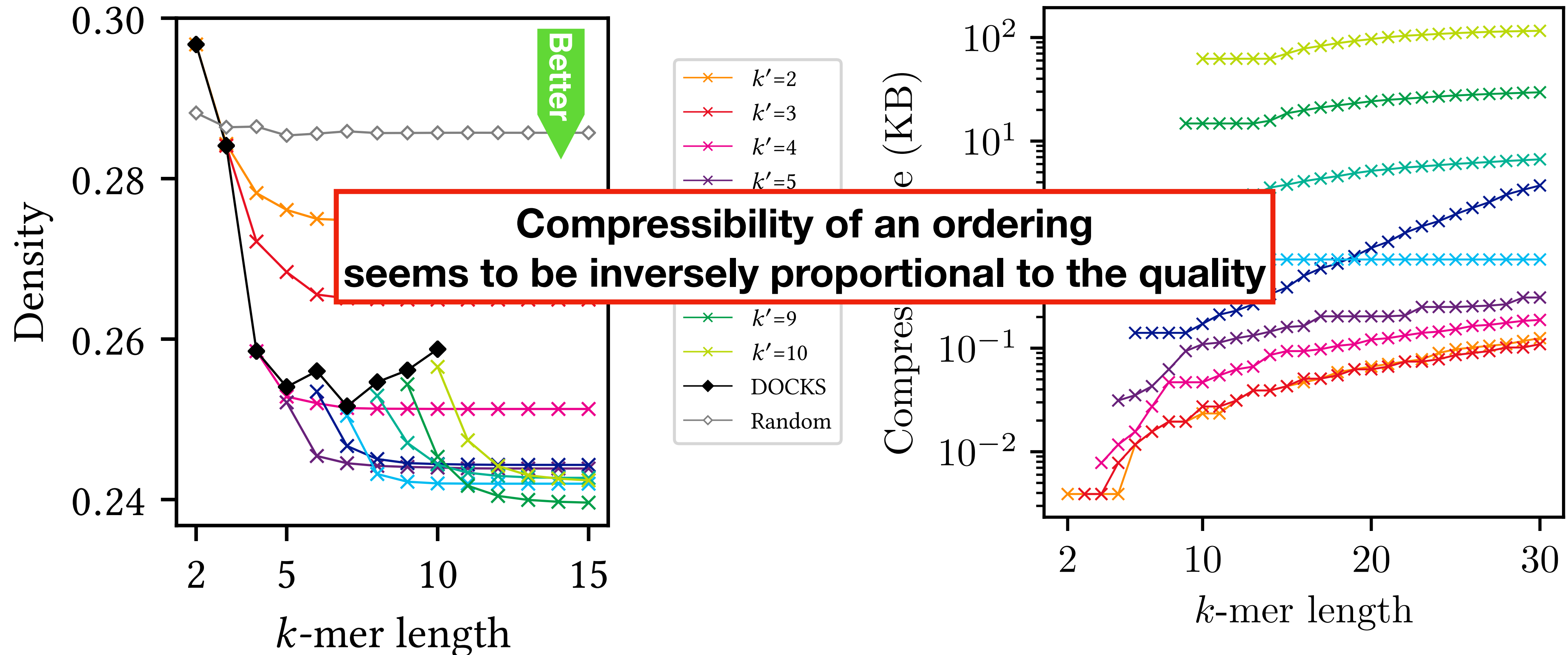
Storing a universal set is inefficient

Stored using a sequence trie, high complexity leads to large files



Storing a universal set is inefficient

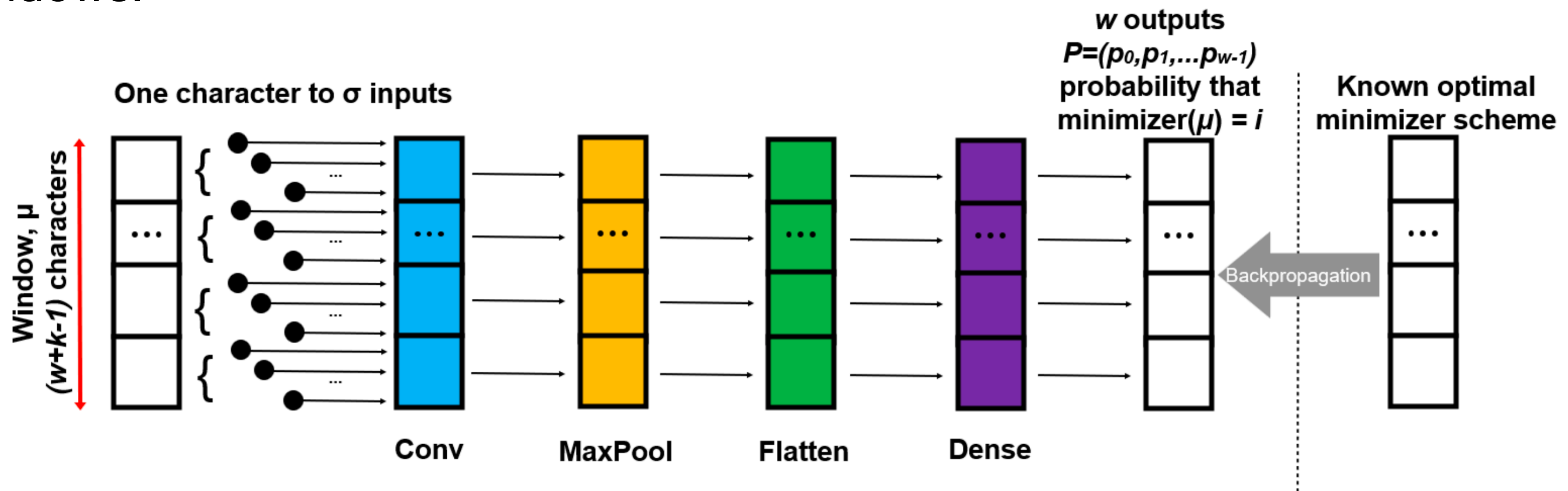
Stored using a sequence trie, high complexity leads to large files



Our learning method

Task -- learn the minimizer schemes using back propagation

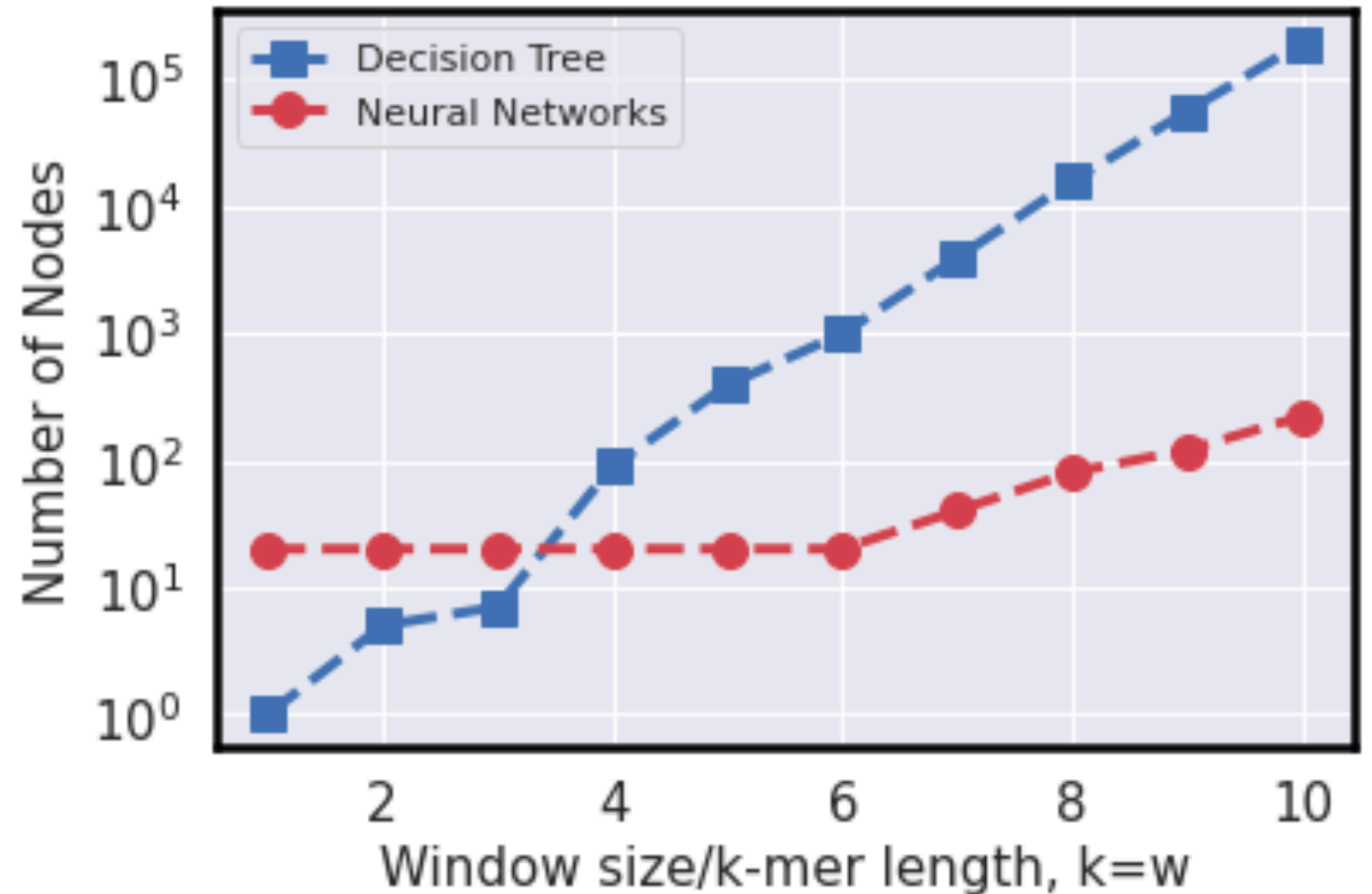
- Our task is to create a network topology is complex enough to encode existing schemes, but not so complicated that it provides extreme training times.
- One issue that arises is that for small values of w and k there may not be enough information to train the network completely since there are only so many unique windows.



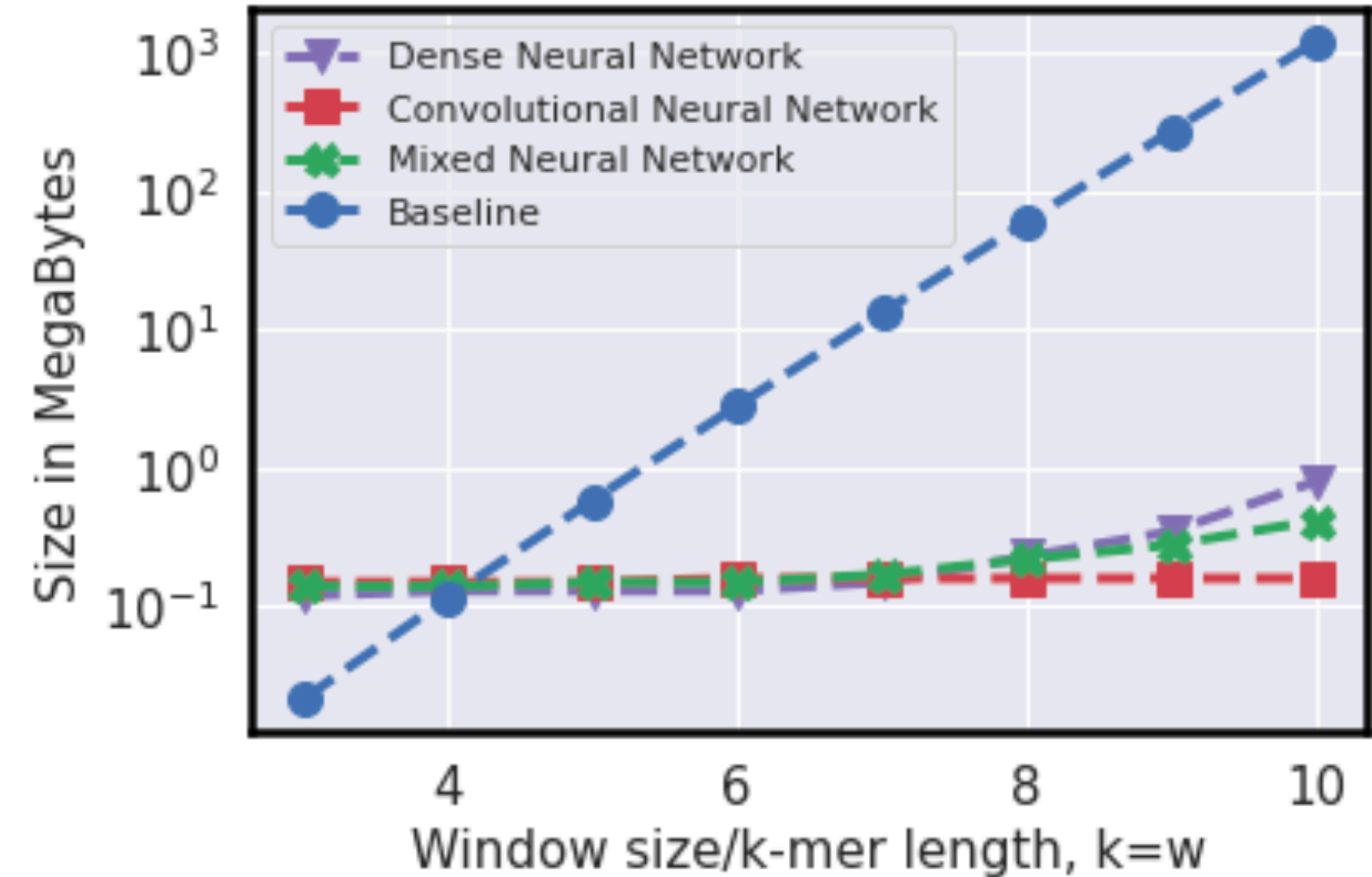
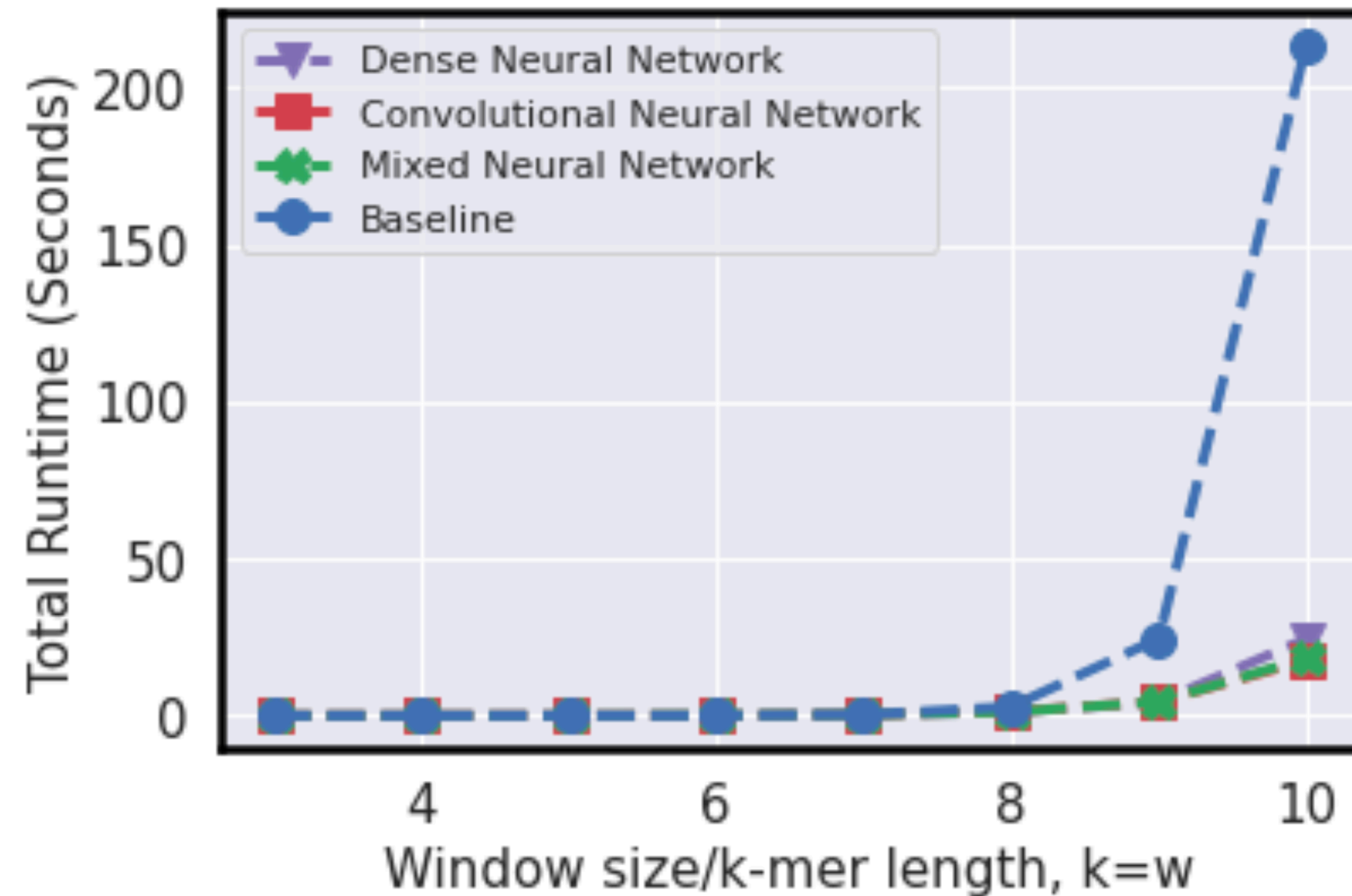
A note about Neural Networks

Used Decision Trees and Dense Neural Networks.

The number of nodes to encode minimizers is significantly larger with decision trees than with neural network implementations.



Performance of the networks



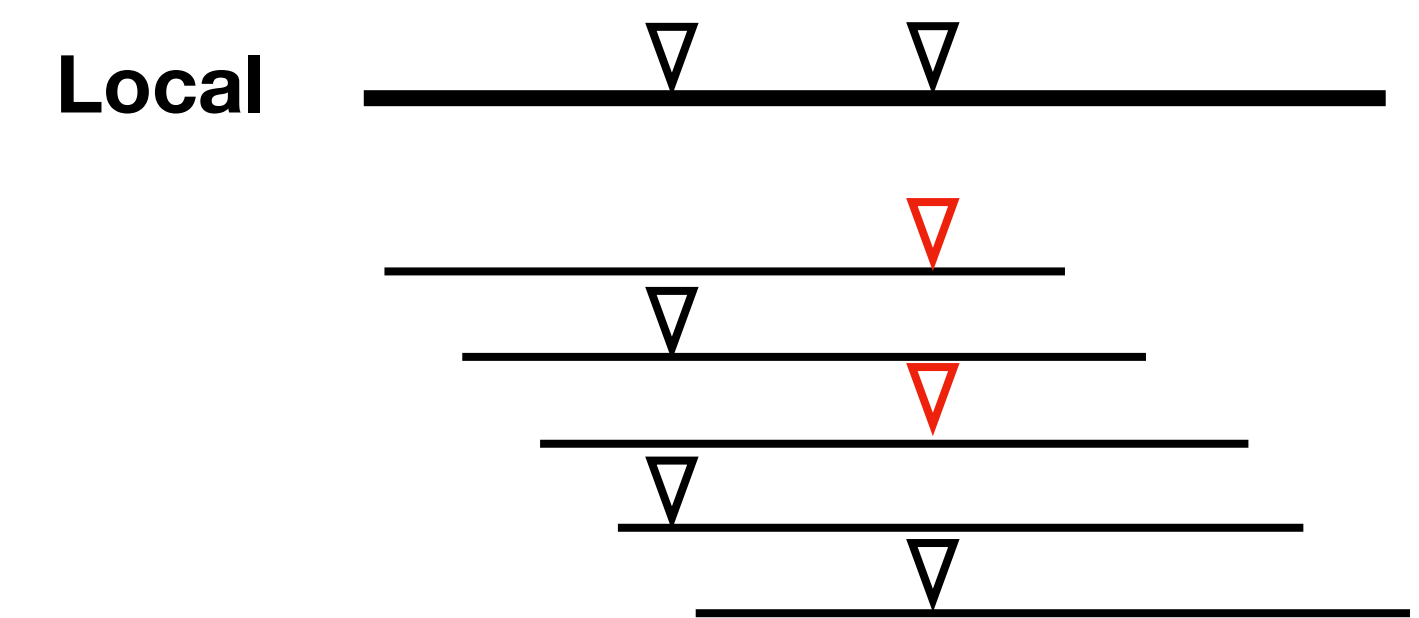
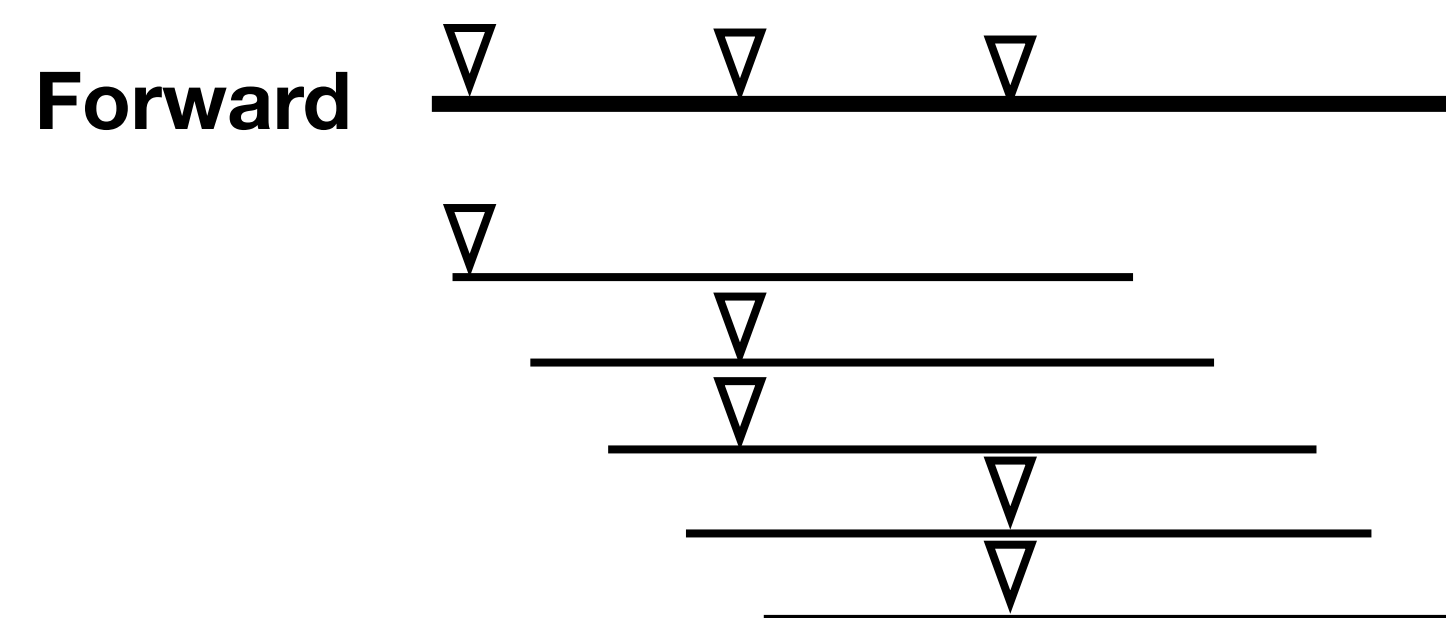
A trained model has a shorter k-mer lookup time and smaller memory footprint than a naïve implementation of minimizers.

Local vs. Forward vs. Minimizer Schemes

Assume we're going to rewrite it $F(M) \rightarrow m$ where M is the ordered set of k -mers from the window, and m is the returned k -mer.

What if we relax the rules a bit:

- **Minimizer Schemes** -- choose the $m = \arg \min_{m' \in M} (O(m'))$
- **Forward Schemes** -- choose any m such that for all M' that can proceed M the choice is at the same position or later
- **Local Schemes** -- choose any m



Minimizer \subset Forward \subset Local

Some of our other work

Multiple sequence alignment benchmark set bias identification

- Fransisco Parra (Senior/REU), Luis Cedillo (Sophomore)

Automatic parameter configuration for additive manufacturing

- Fernando Sepulveda (Freshman)
- Collaboration with faculty in Electrical and Computer Engineering

Identifying unresolved space objects using ground-based hyperspectral imaging

- Taposh Sarker (Graduate Student)
- Collaboration with faculty in Electrical and Computer Engineering

Acknowledgments


The DeBlasio Lab

Luis Cedillo (UG, Facet-NN/LR)
Demetrius Hernandez (UG, Minimizers)
Taposh Sarkar (PhD, USROs)
Fernando Sepulveda (UG, Additive Manufacturing)
Md. Easin Hasan (former MS, BWA)
Hector Richart-Ruiz (former UG)

The Current Collaborators

Miguel Velez-Reyes
Arizbe Najera

Contact

deblasiolab.org
 danfdeblasio

Previous Collaborators



John Kececioglu
Travis Wheeler (Montana)
Jen Wisecaver (Purdue)



Carl Kingsford
Fiyinfoluwa Gbosibo (visiting UG)
Kwanho Kim (MS)
Guillaume Marçais

Funding

