Precision Medicine 2: Genomic Processing

10-742

Carl Kingsford, November 7, 2024

Roadmap

- Questions from last time
- Importance of accurate and efficient molecular feature extraction
- Predicting gene expression values from RNA sequencing data
- Learning how to run sequencing processing tools

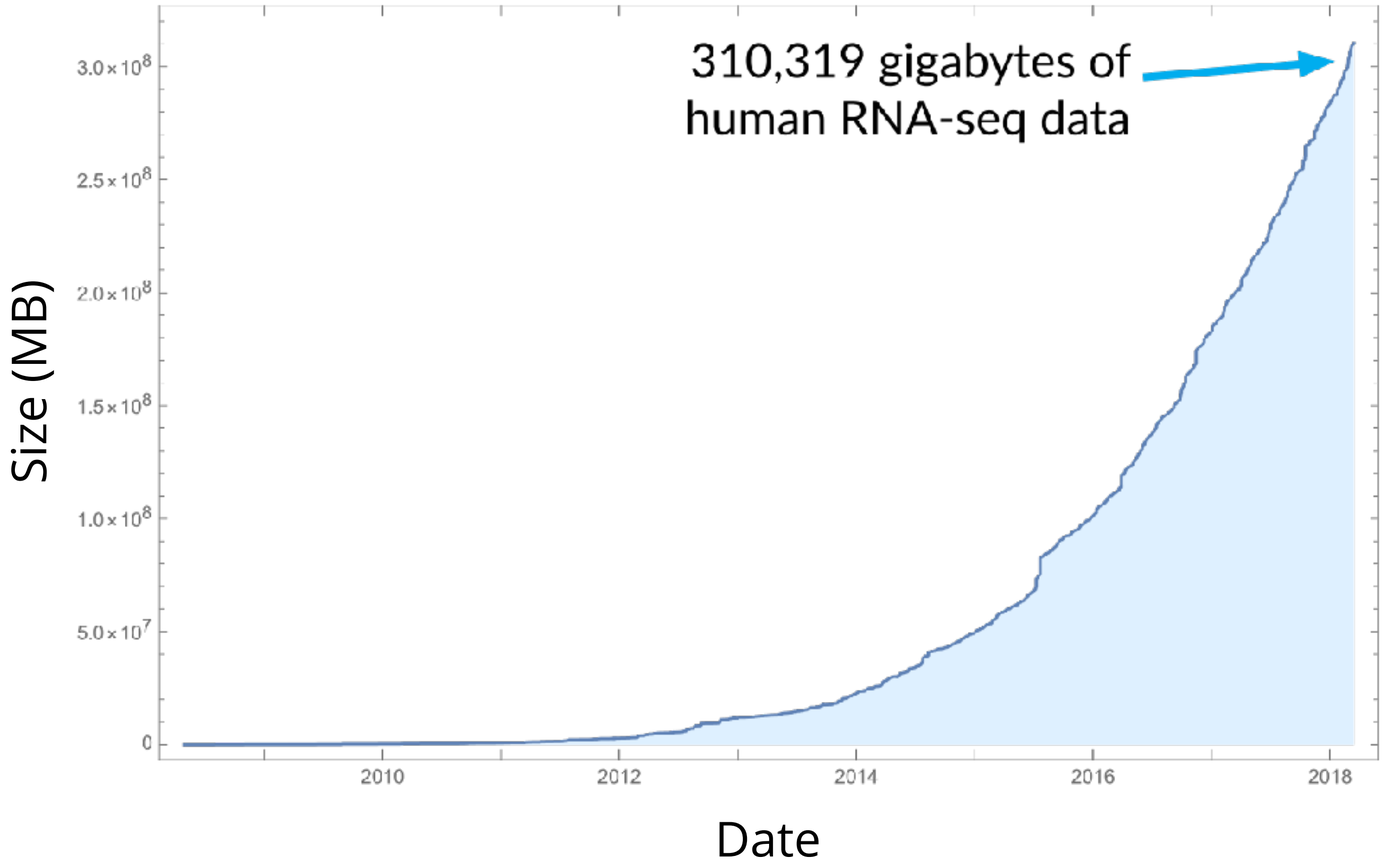
Suggested Questions About Last Time

- What are the biggest challenges in commercializing work in precision medicine?
- Who are the customers and what do they want (and how do they think/evaluate an advance)?
- What are the biggest failure states / risks for PM?
- What are the essential components of success for a PM startup?
- What are things to avoid when doing PM in a commercial setting?
- What are the future opportunities for PM in a commercial setting?
- Why Knowledge Graphs? Aren't there other abstractions approaches?
- How are open source / open methods viewed in translational settings?
- I have a new algorithm / technique, how should I get it used?
- I'm considering applying my ML/AI expertise to biomedical domains vs. other domains; What should I think about while making that decision?

Transcriptomics is the Next Frontier

- DNA tells you only part of the story.
- Mostly static information about what *might* happen.
- RNA (gene expression / transcriptomics) = what *is* happening (in response to a drug or disease)
- Dynamic measurement of response to drug over time.
- Low-cost, growing in use.

The Available Sequencing Data is Growing Quickly



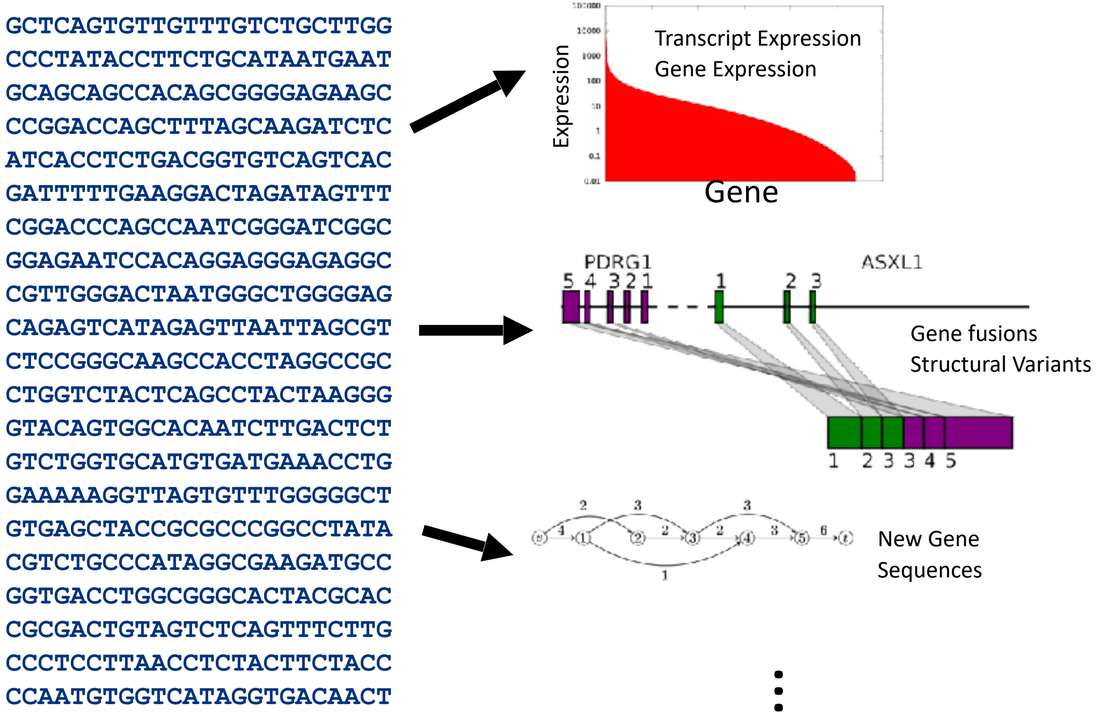
Illumina sequencing machines generate > Netflix catalog worth of data per year.

Scalability is essential.

Must make this data AI-ready.

Sequencing is not useful without extracting actionable features

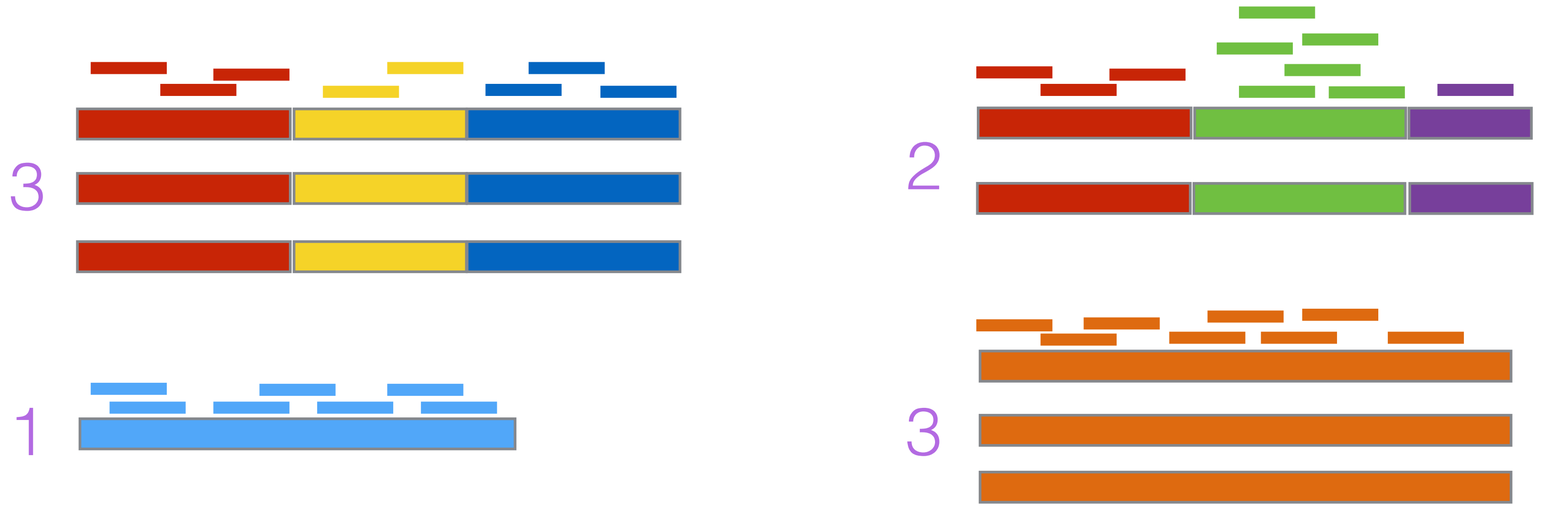
- Sequencing provides tens or hundreds of millions of fragmented observations
- Can't directly use for ML / AI
- Efficient algorithms are needed to accurately predict:
  - Mutations
  - Expression levels
  - Novel transcripts
  - Biological pathway regulation
  - …
- These are the features needed for AI/ML to use this data



Part 1: Predicting Gene Expression Levels

Problem: Fast gene expression estimation from RNA-seq

Goal: estimate the abundance of each kind of transcript given short reads sampled from the expressed transcripts.

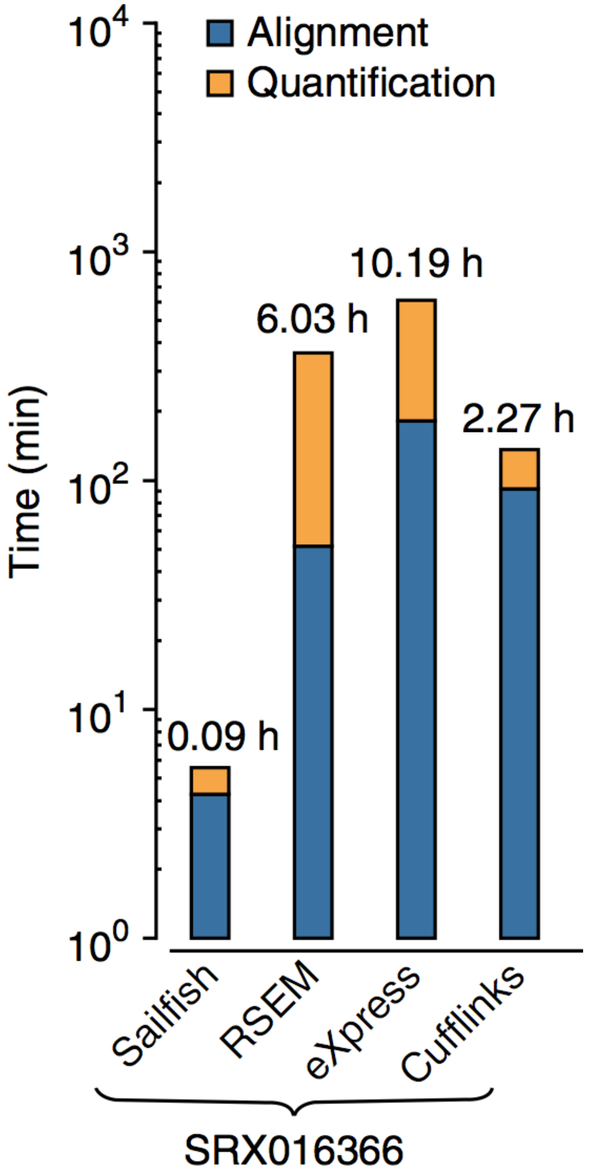


Challenges:

- hundreds of millions of short reads per experiment
- finding locations of reads (mapping) is traditionally slow
- alternative splicing creates ambiguity about where reads came from
- sampling of reads is not uniform

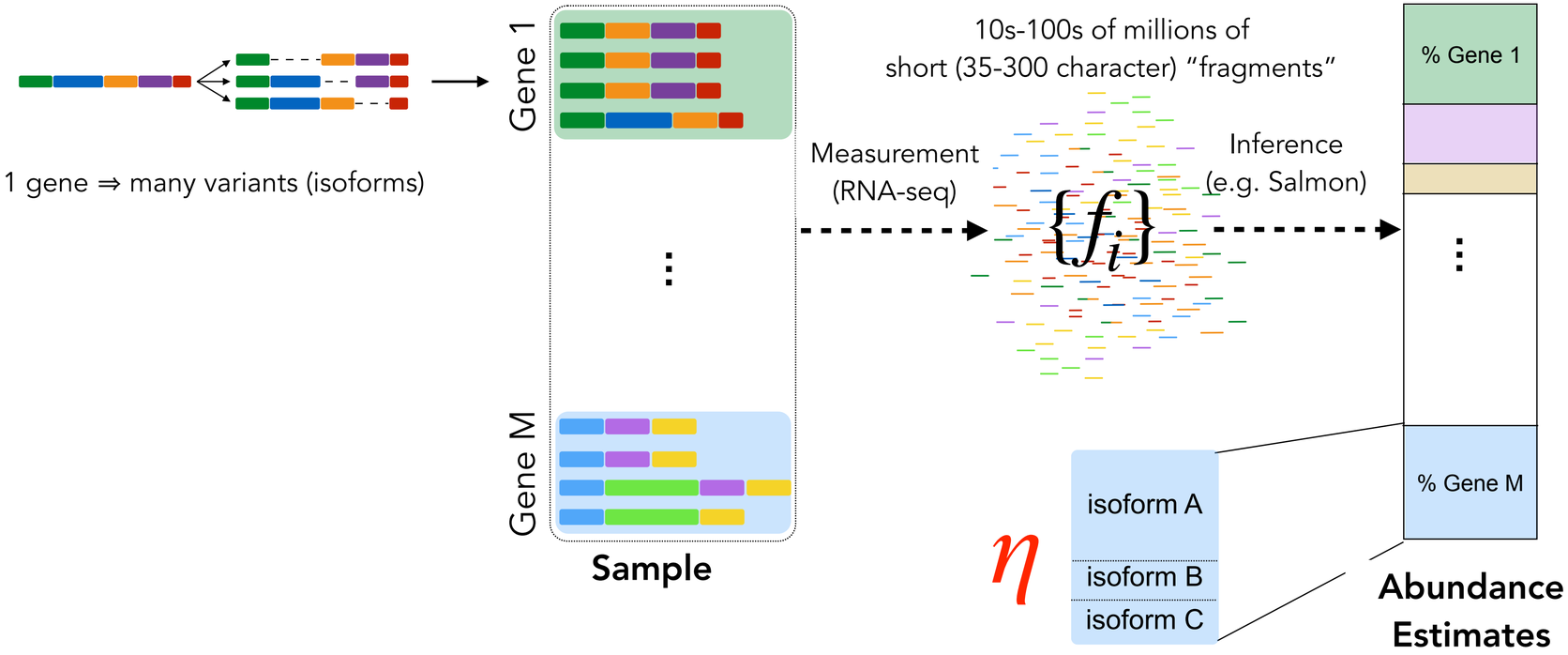
Sailfish: Ultrafast Gene Expression Quantification

- Fast expectation maximization algorithm
- Extremely parallelized
- Uses small data atoms rather than long sequences
- More tolerant of genetic variation between individuals



Patro, Mount, Kingsford, *Nature Biotech*, 2014

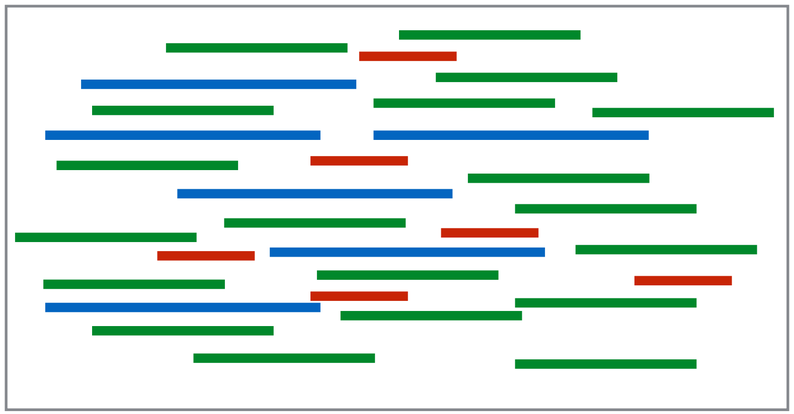
Expression Quantification Inference



Rob Patro, Geet Duggal, Michael I Love, Rafael A Irizarry, Carl Kingsford (2017).
Salmon provides fast and bias-aware quantification of transcript expression.
Nature Methods **14**:417-419.

Inference Problem

Experimental mixture:



length( ) = 100 x 6 copies = 600 nt ~ 30% blue

length( ) = 66 x 19 copies = 1254 nt ~ 60% green

length( ) = 33 x 6 copies = 198 nt ~ 10% red

These values η = [0.3, 0.6, 0.1] are the *nucleotide fractions*;
they are the quantities we want to infer

Maximum Likelihood Model

nucleotide fractions

true read origins

assumes independence of fragments

$$\Pr\{\mathcal{F} \mid \boldsymbol{\eta}, \boldsymbol{Z}, \mathcal{T}\} = \prod_{j=1}^{N} \Pr\{f_j \mid \boldsymbol{\eta}, \boldsymbol{Z}, \mathcal{T}\}$$

observed fragments (reads)

$$= \prod_{j=1}^{N} \sum_{i=1}^{M} \Pr\{t_i \mid \boldsymbol{\eta}\} \cdot \boxed{\Pr\{f_j \mid t_i, \boldsymbol{z}_{ji} = 1\}}$$

Prob. of selecting t_i *given* **η**

Depends on abundance estimate

Prob. of generating fragment f_j *given* t_i

Independent of abundance estimate

“Bias” Model

a fragment
starting at given position

$$\Pr\{f_j \mid t_i\} = \Pr\{\ell \mid t_i\} \cdot \Pr\{p \mid t_i, \ell\} \cdot \Pr\{o \mid t_i\} \cdot \Pr\{a \mid f_j, t_i, p, o, \ell\}$$

a fragment
of the given length

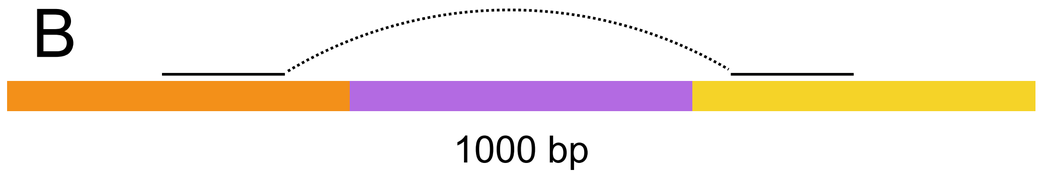
a fragment
of given orientation

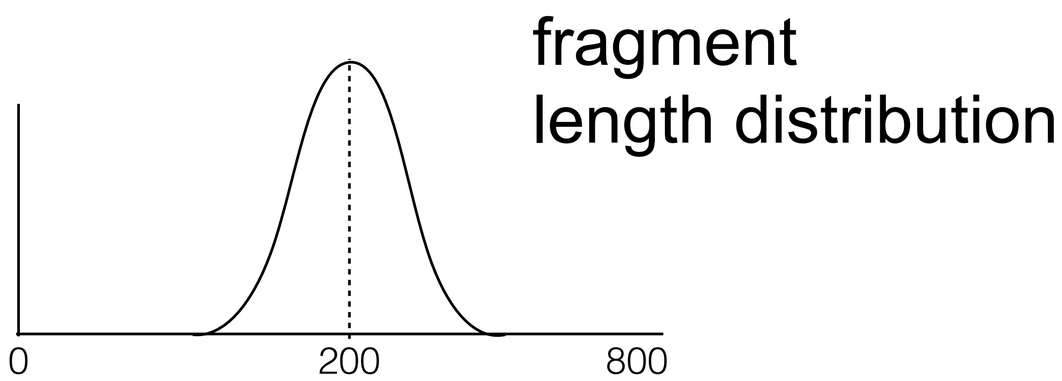
generating the given
alignment

- Salmon estimates an auxiliary model *from the data* for each term (e.g. fragment length, fragment start position, etc.)
- Accounts for sample-specific parameters and biases.

Why does this matter?

“Bias” model can provide strong information about origin of a fragment.
For example:

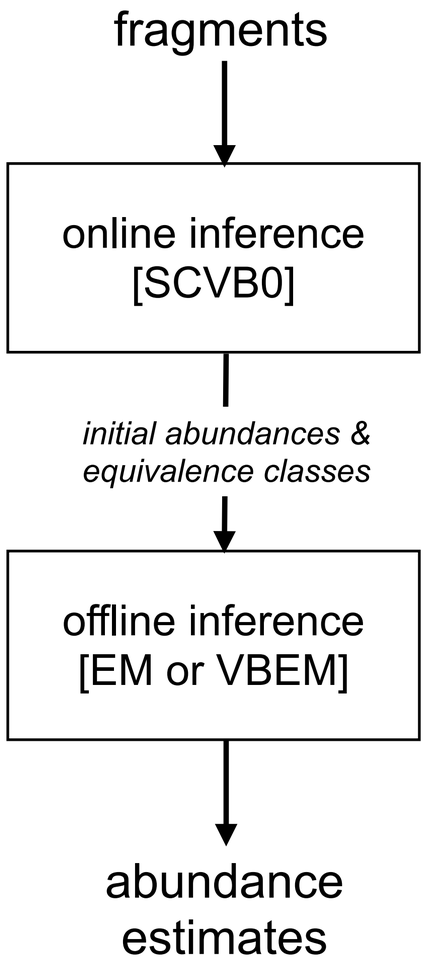




Salmon's two phase inference procedure

Optimizes the full model using a streaming algorithm & trains the "bias" model parameters

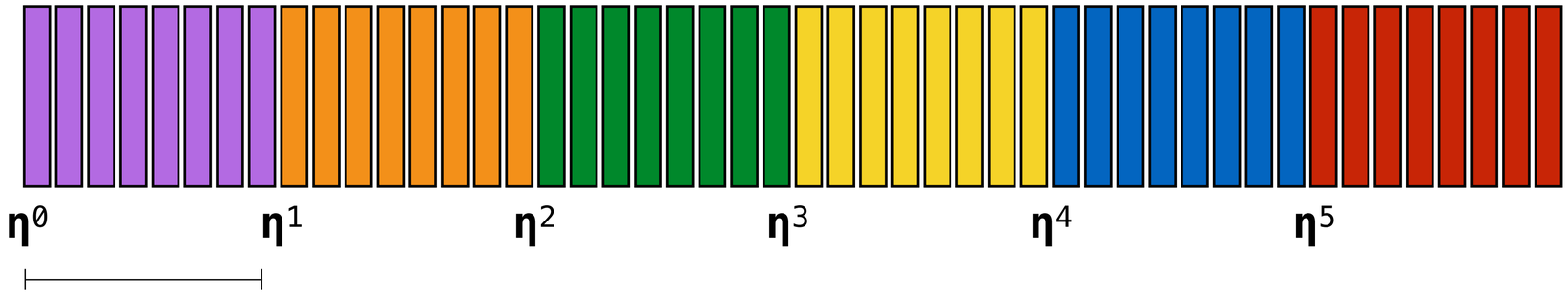
Refines the abundance estimates using a reduced representation.



Phase 1: Online Inference

Based on: Foulds et al. Stochastic collapsed variational Bayesian inference for latent Dirichlet allocation. ACM SIGKDD, 2013.

Process fragments in batches:



Compute local η' using η^{t-1} & current "bias" model to allocate fragments

Update global nucleotide fractions: $\eta^t = \eta^{t-1} + a^t \eta'$

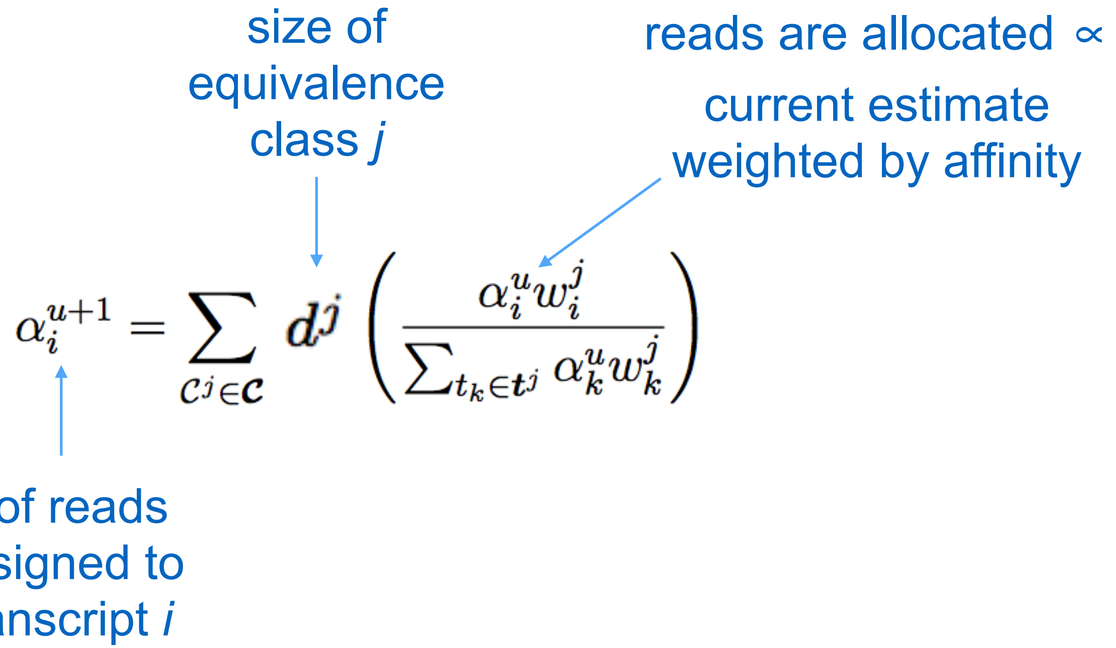
Update "bias" model

Weighting factor that decays over time

Often converges very quickly.
Compare-And-Swap (CAS) for synchronizing updates of different batches

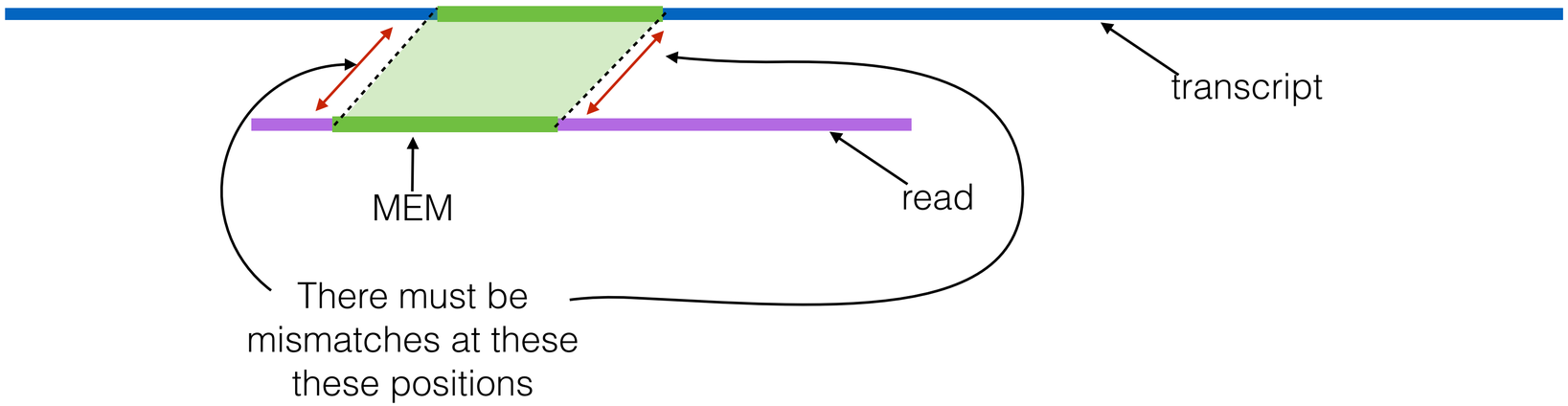
Phase 2: Offline Inference

Repeatedly reallocate fragments according to current abundance estimates & “bias” model until convergence:



Salmon's Original Lightweight Alignment

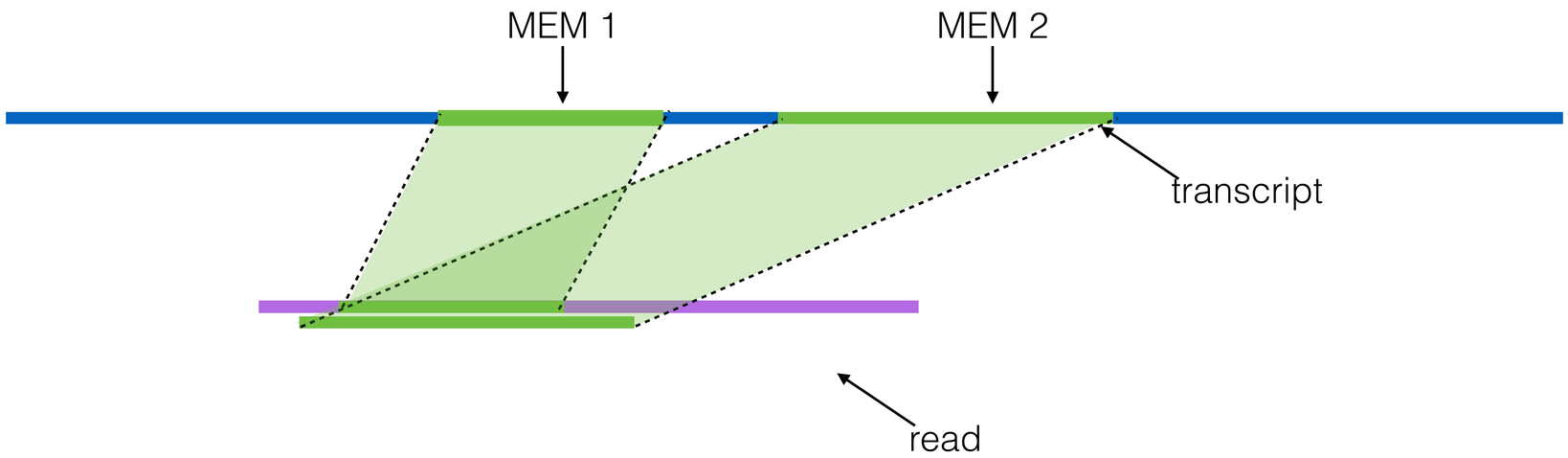
- Salmon replaces the time-consuming read alignment step with a new approach that quickly finds chains of "maximal exact matches":



A maximal exact match is an exact match between the read and a transcript that can't be extended in either direction.

SMEMs

A **super** maximal exact match is a MEM that is not contained in any other MEM in either the query or the reference:

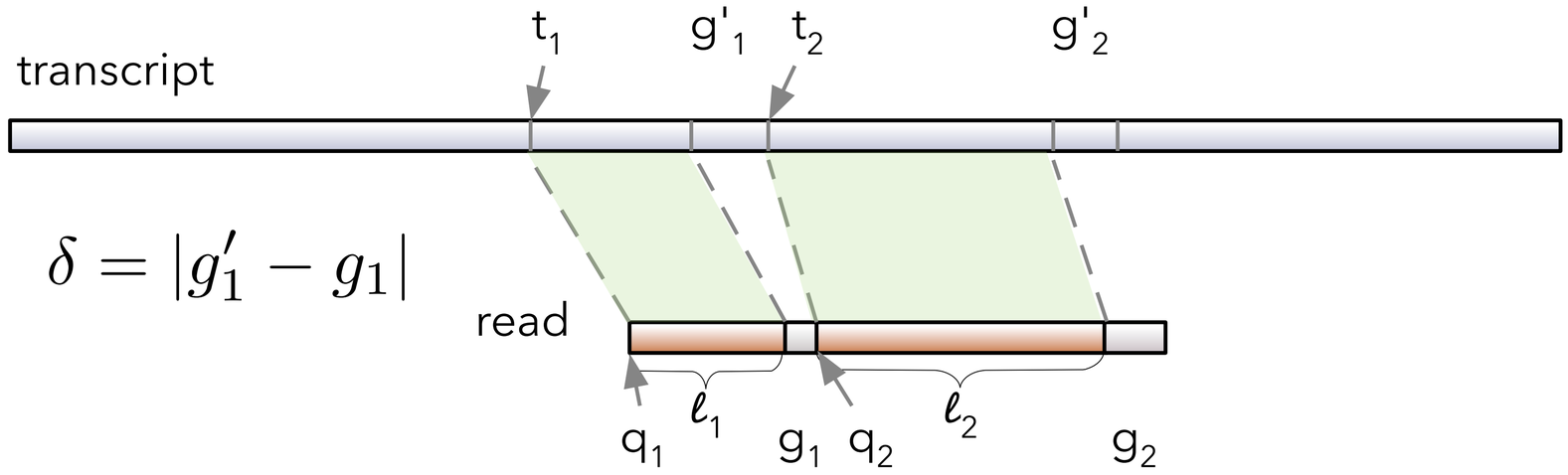


MEM 1 is not an SMEM, while MEM 2 is.

Lightweight alignment

Lightweight alignment looks for δ-consistent chains of SMEMs.

A chain of SMEMs is δ-consistent if the total difference in gap sizes between the SMEMs is ≤ δ



Salmon requires the SMEMs to cover at least 65% of the read.

Revising the Challenges

- finding locations of reads (mapping) is traditionally slow

→ Use lightweight alignment

- alternative splicing creates ambiguity about where reads came from

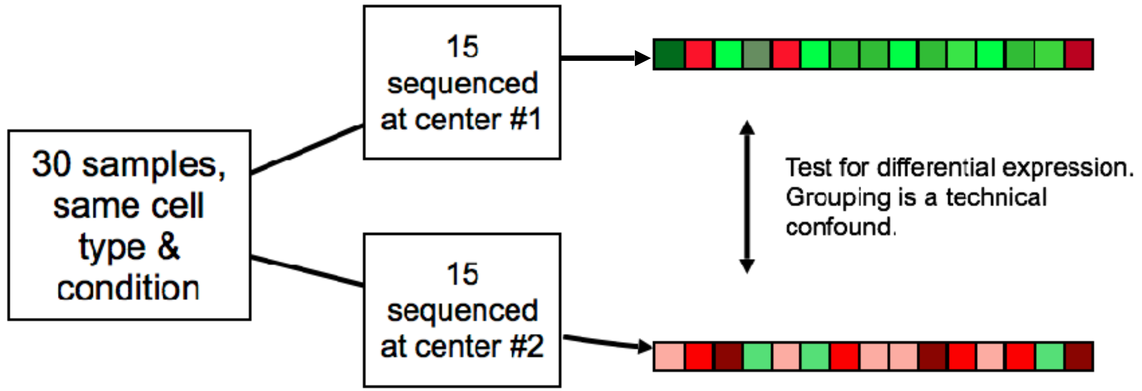
→ Use 2-phase EM inference algorithm

- sampling of reads is not uniform

→ Use bias model learned from data

We developed the first fast & accurate method for gene expression quantification

- Convert 10s of millions of short reads into expression levels of all gene transcripts.
- Uses probabilistic models and advanced inference algorithms.
- Downloaded > 650,000 times.



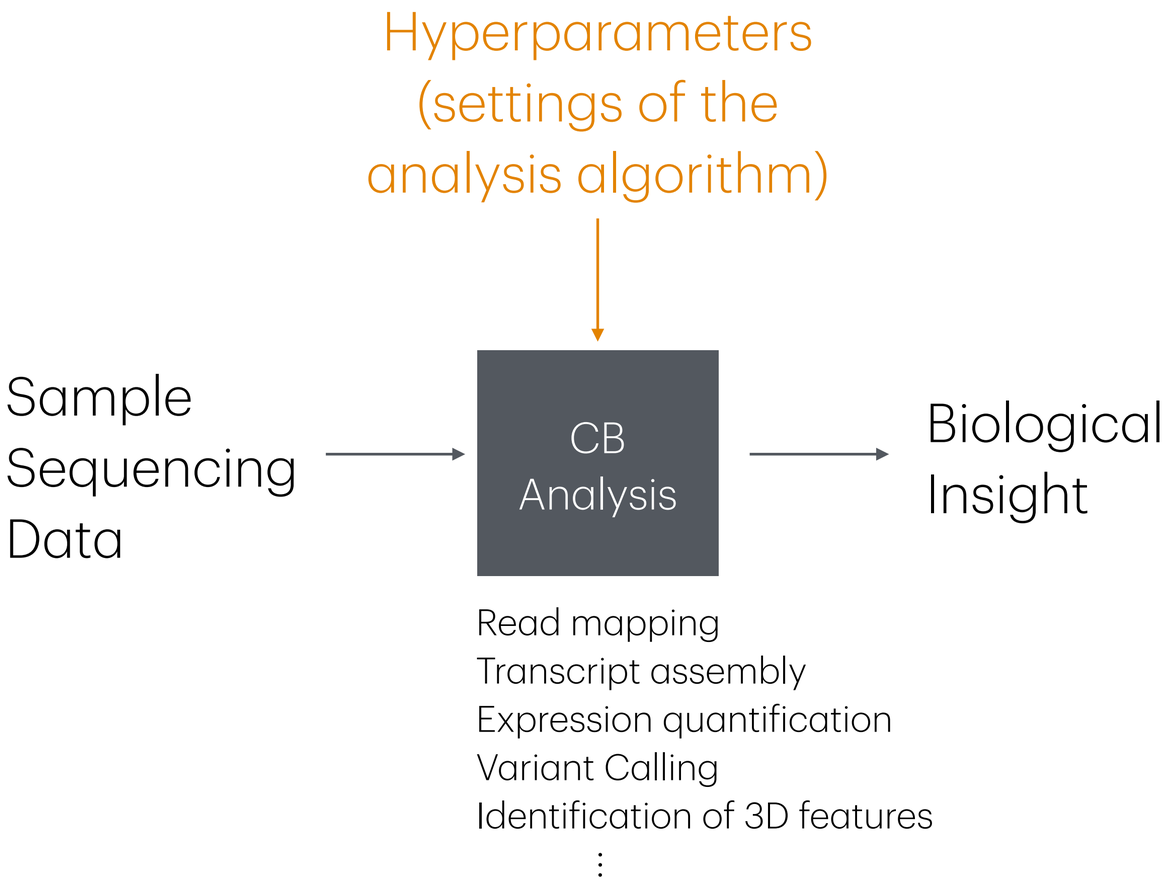
Gene Type	Salmon	kallisto	eXpress
All genes	1171	2620	2472
2-isoform genes	224	545	531

Patro et al., Nature Methods (2017)
Patro et al., Nature Biotechnology (2014)

Part 2: Learning sample-specific hyperparameters

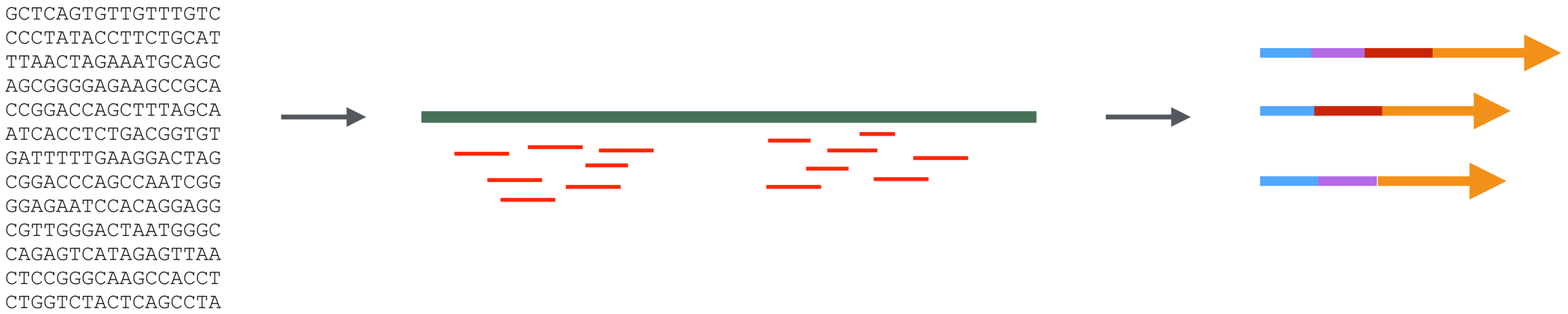
Adaptive, sample-specific parameter selection for more accurate transcript assembly.
Yihang Shen, Zhiwen Yan, Carl Kingsford.
https://www.biorxiv.org/content/10.1101/2024.01.25.577290

The challenge of automated bioinformatics



- Selection of analysis hyperparameters is non-trivial since the analysis steps can be complex algorithms.
- The choice of parameters significantly affects the of accuracy many analyses.
- Manual setting of the parameters takes costly time.
- Hand-setting parameters reduces reproducibility & may introduce biases.

Transcript assembly is the problem of inferring new transcripts from RNA sequencing

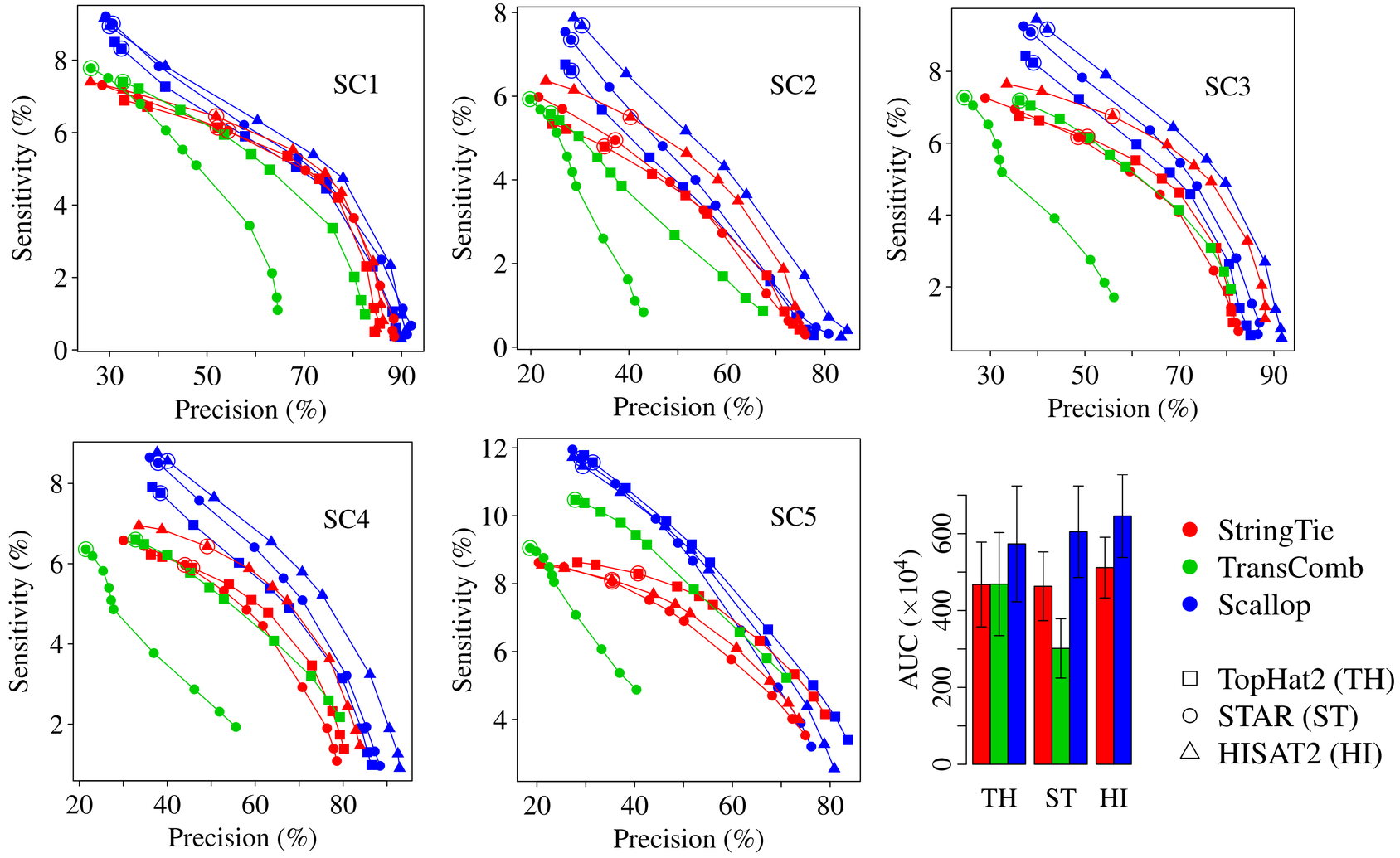


RNA-seq reads

Mapped to a reference genome

Infer expressed transcripts

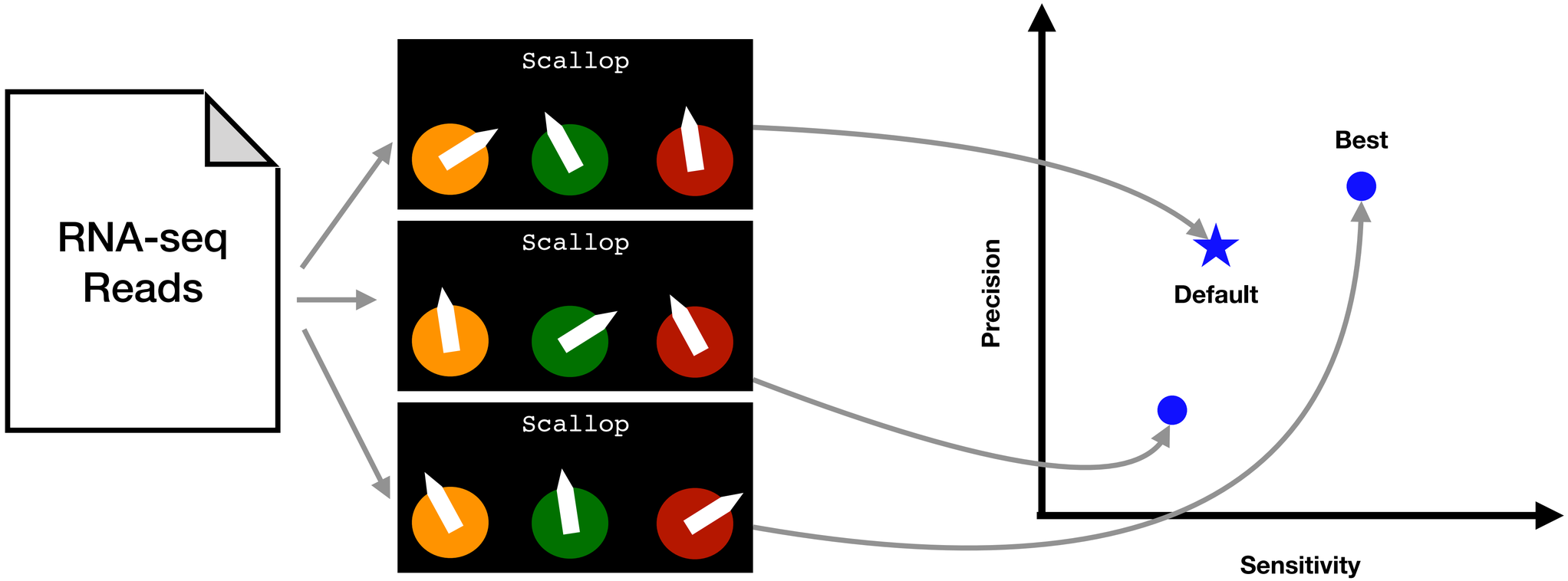
Transcript assembly remains a challenging problem



(Shao, Kingsford, 2017)

See also Tung, Shao, Kingsford, (2019) on assembling long reads

Selecting better parameters leads to greater accuracy



Application defaults are best on average, but possibly poor for some input.

Increase accuracy by selecting a parameter choice for each input.

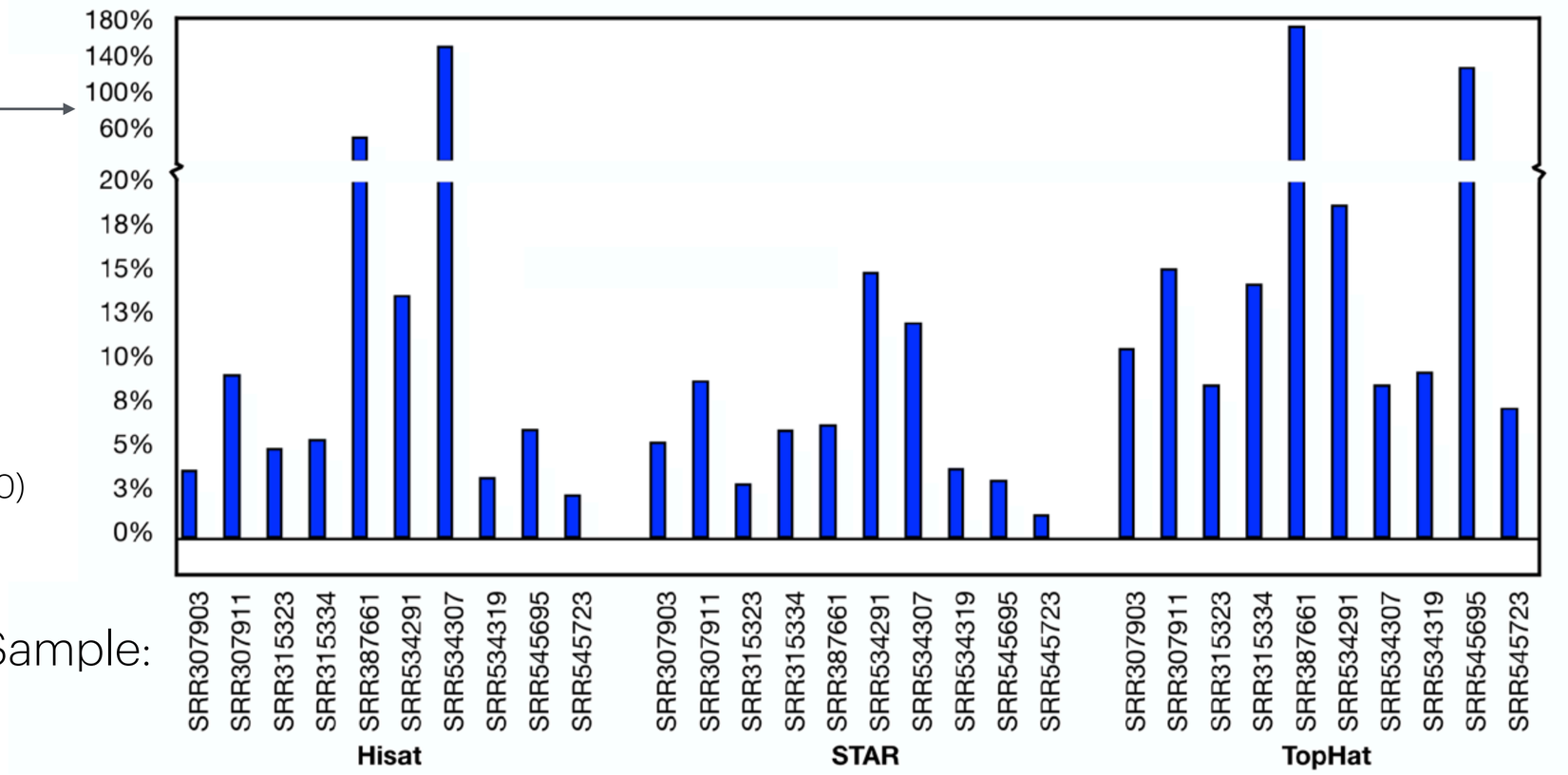
Scallop has parameters of mixed type

Parameter	Default value	Type
uniquely_mapped_only	False	binary
use_second_alignment	False	binary
max_dp_table_size	10000	integer
max_edit_distance	10	integer
max_num_exons	1000	integer
min_bundle_gap	50	integer
min_exon_length	20	integer
min_flank_length	3	integer
min_mapping_quality	1	integer
min_num_hits_in_bundle	20	integer
min_router_count	1	integer
min_splice_boundary_hits	1	integer
min_subregion_gap	3	integer
min_subregion_length	15	integer
min_transcript_length_base	150	integer
min_transcript_length_increase	50	integer
max_intron_contamination_coverage	2	float
min_subregion_overlap	1.5	float

Fixed parameter advising improves transcript assembly

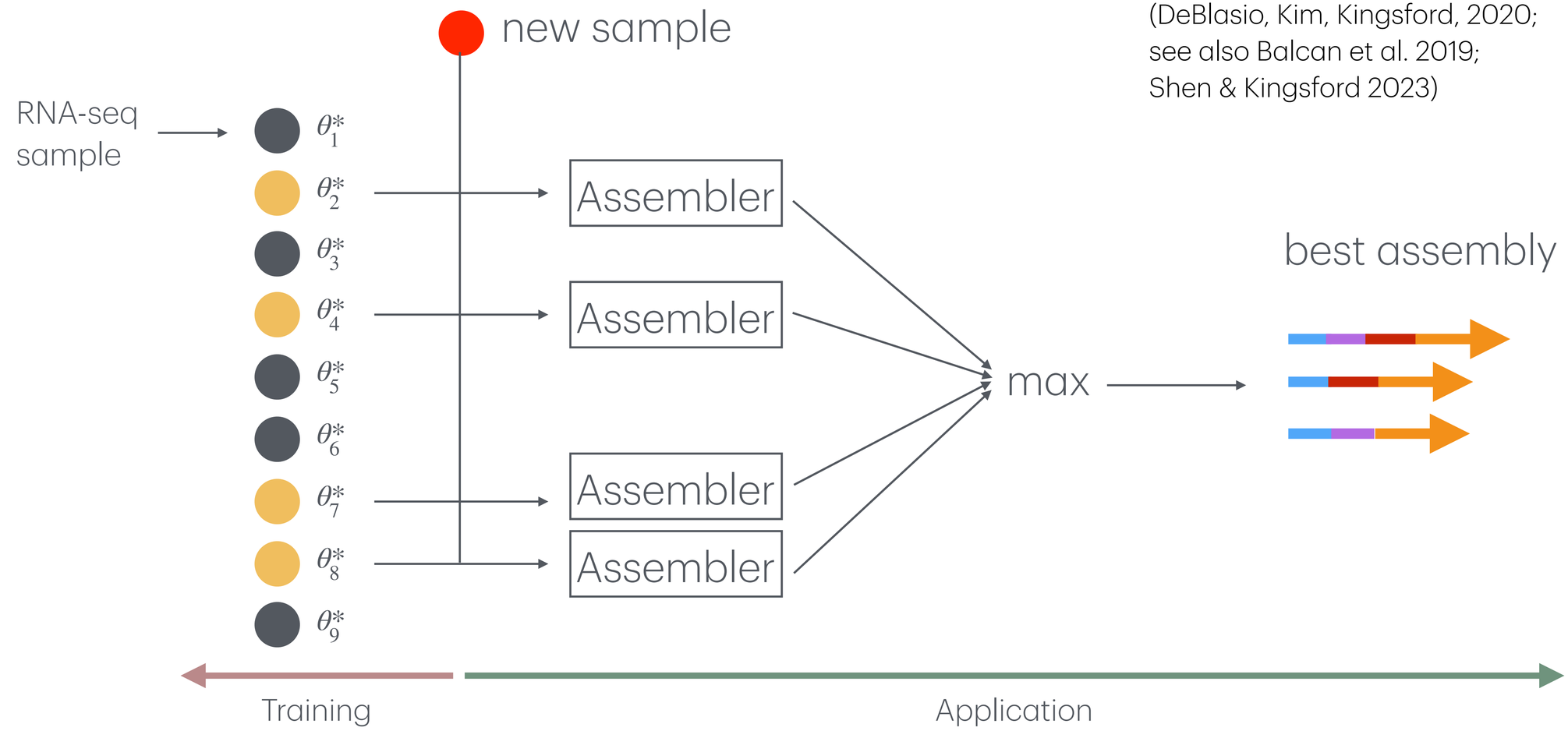
Increase in transcript assembly accuracy when using automatically a single set of 30 parameter vectors

(DeBlasio, Kim, Kingsford, 2020)



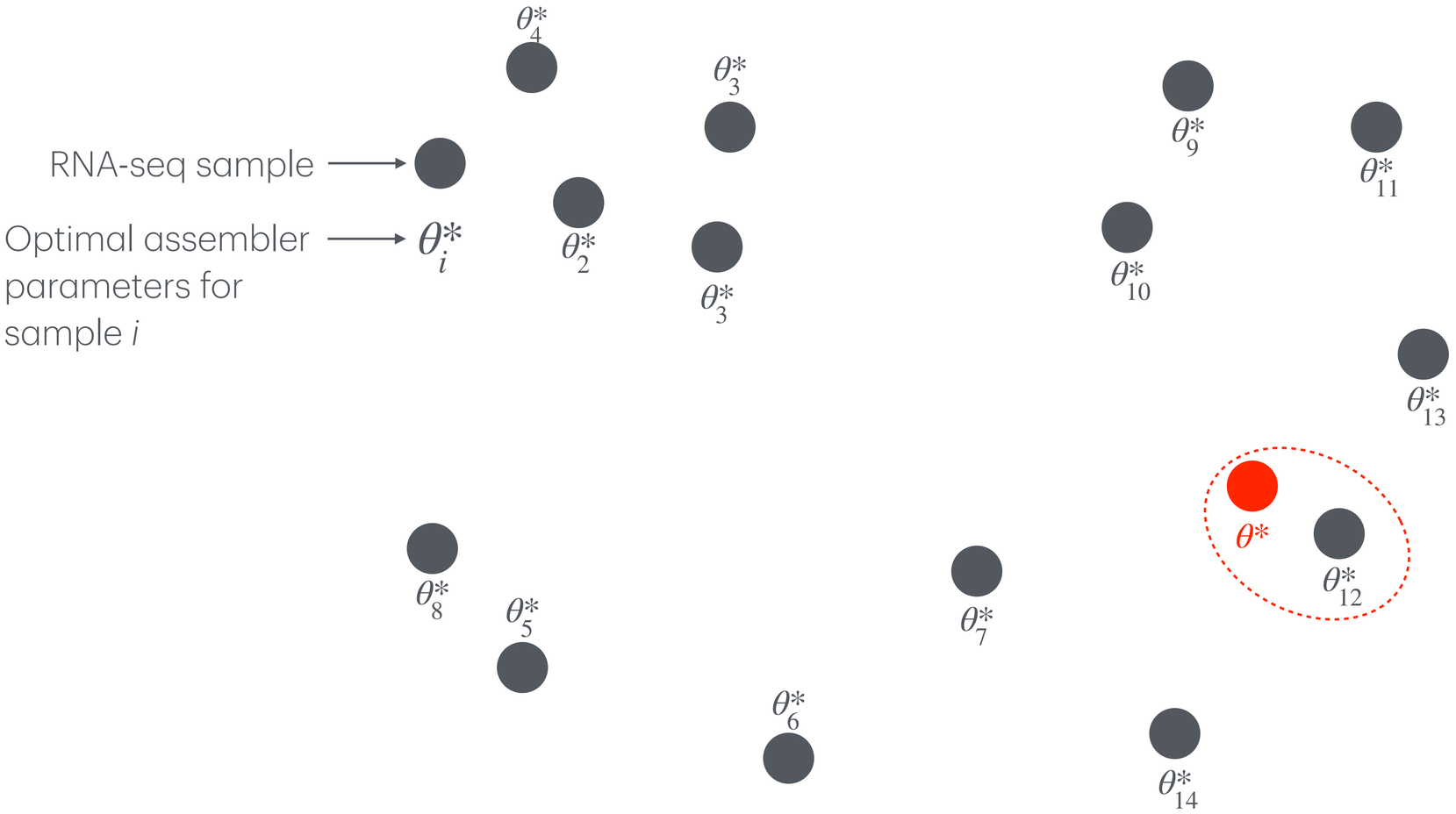
Aligner used to create the input to the assembler

Fixed parameter sets are computationally slow and non-adaptive

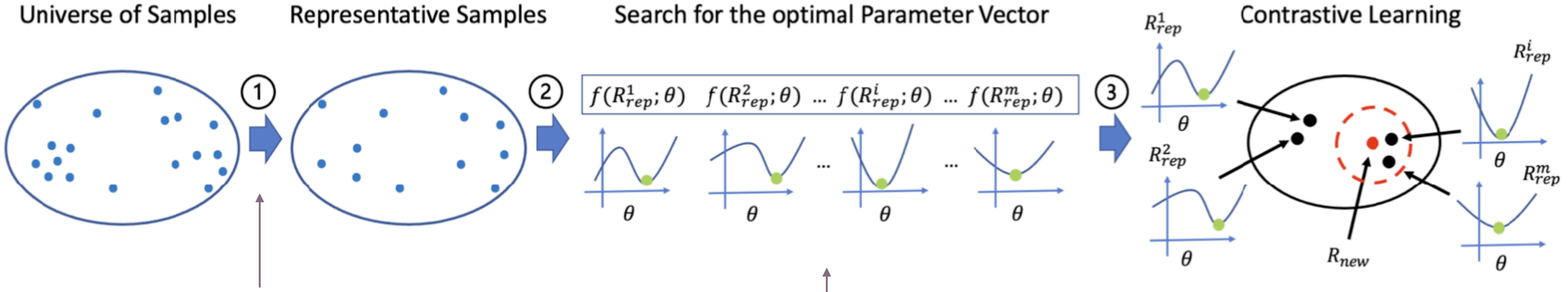


(DeBlasio, Kim, Kingsford, 2020; see also Balcan et al. 2019; Shen & Kingsford 2023)

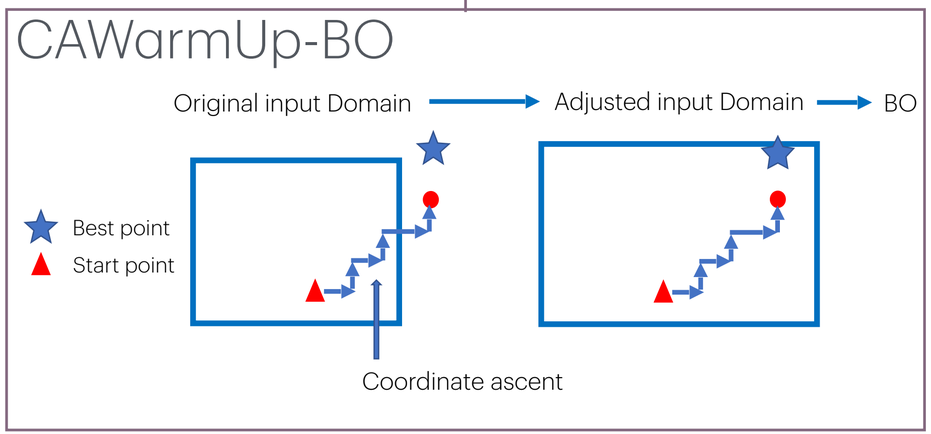
“Similar” samples should have similar parameters



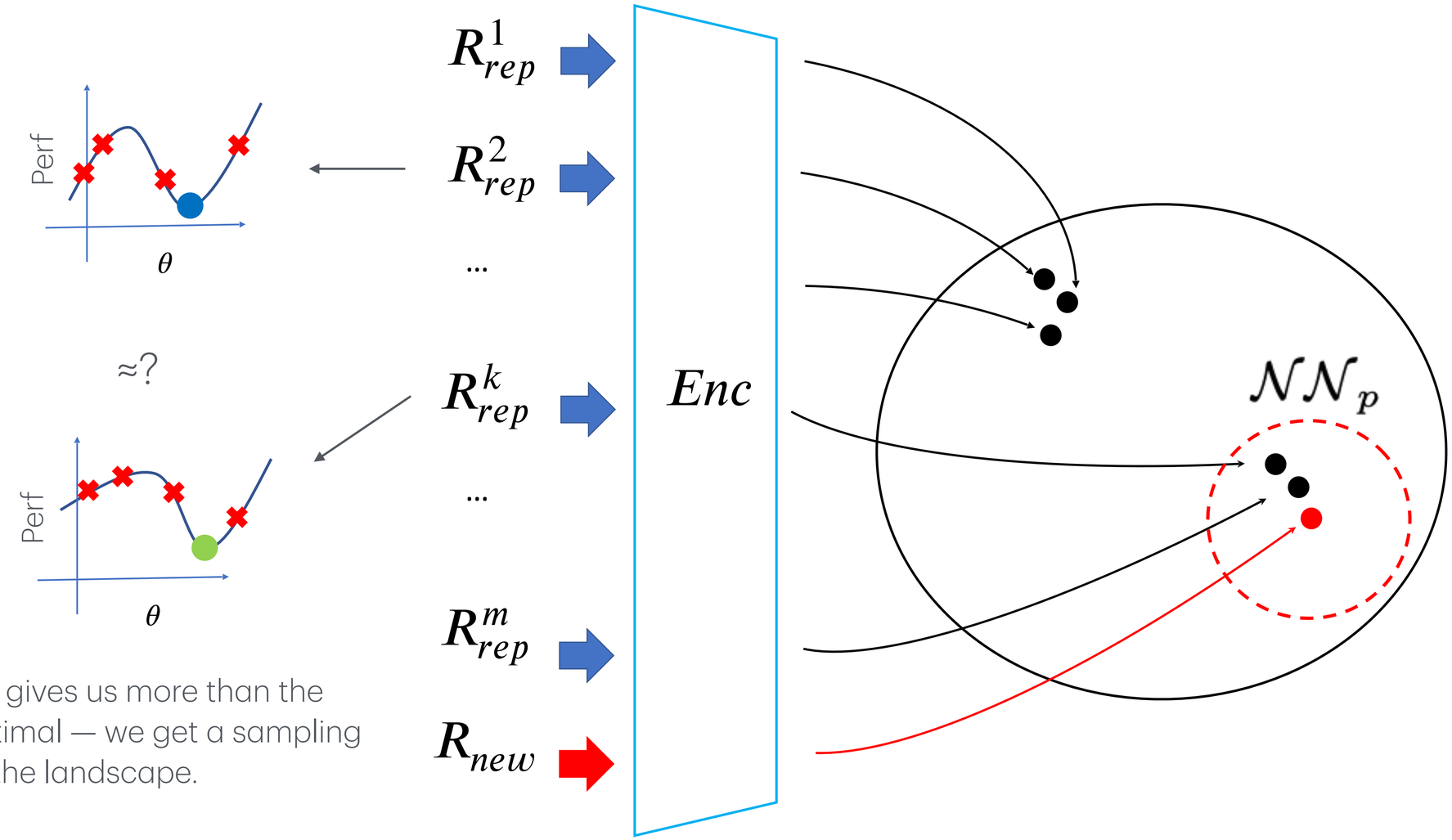
Combining Bayesian Optimization and Contrastive Learning leads to a performant framework



Tung & Kingsford, 2021



Contrastive learning provides an approach to learn a good embedding

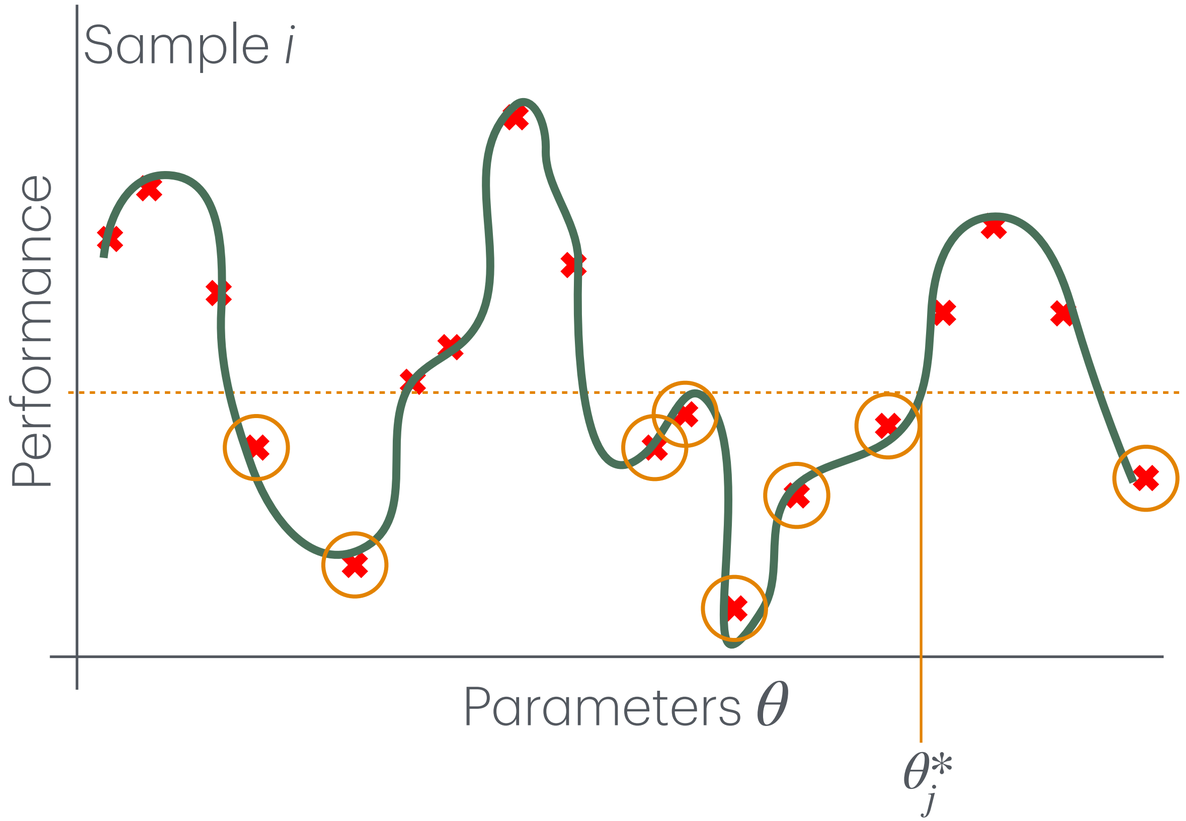


BO gives us more than the optimal — we get a sampling of the landscape.

Want: samples with similar good parameters to embed close together.

For new samples, we can use the parameters for the nearest neighbors.

We measure the similarity between parameter landscapes



Parameters χ_i sampled by BO for sample *i*

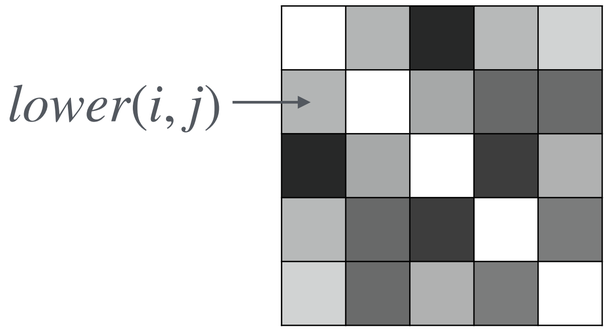
Gaussian Process model fit μ_i

Performance better than the optimal of sample *j*

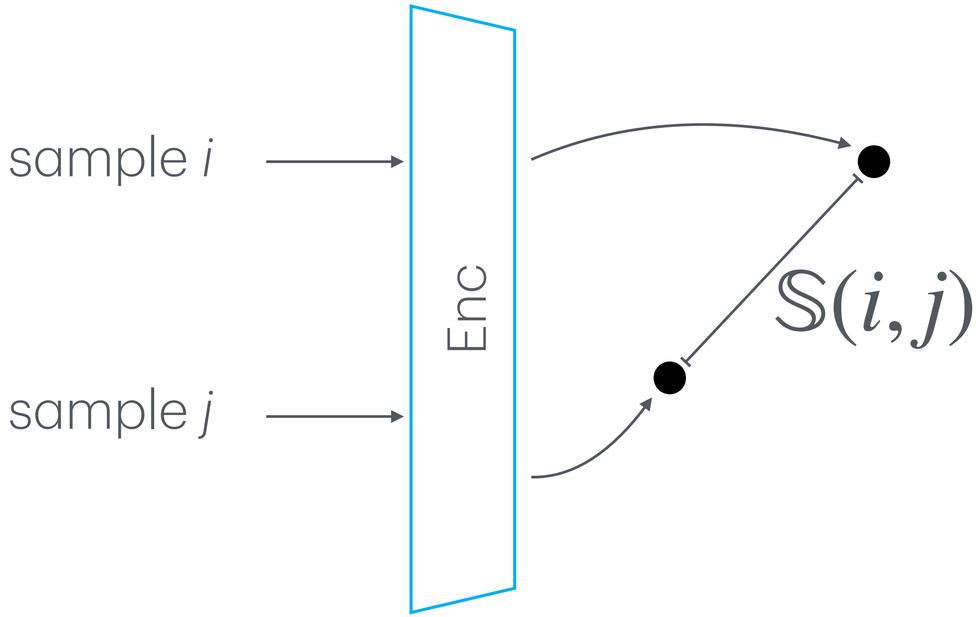
$$lower'(i,j) = \frac{\left|\left\{\left(\theta_i^g, y_i^g\right) \in \chi_i \mid \mu_i(\theta_i^g) < \mu_i(\theta_j^*)\right\}\right|}{|\chi_i|}$$

$$lower(i,j) = average\left(lower'(i,j), lower'(j,i)\right)$$

We use this similarity to train an encoder



Shortest path using $lower(i,j)$ → $\mathbb{D}(i,j)$ → $\mathbb{S}(i,j) = 1 - 2\mathbb{D}(i,j)$



We use a permutation-invariant contrastive learning architecture

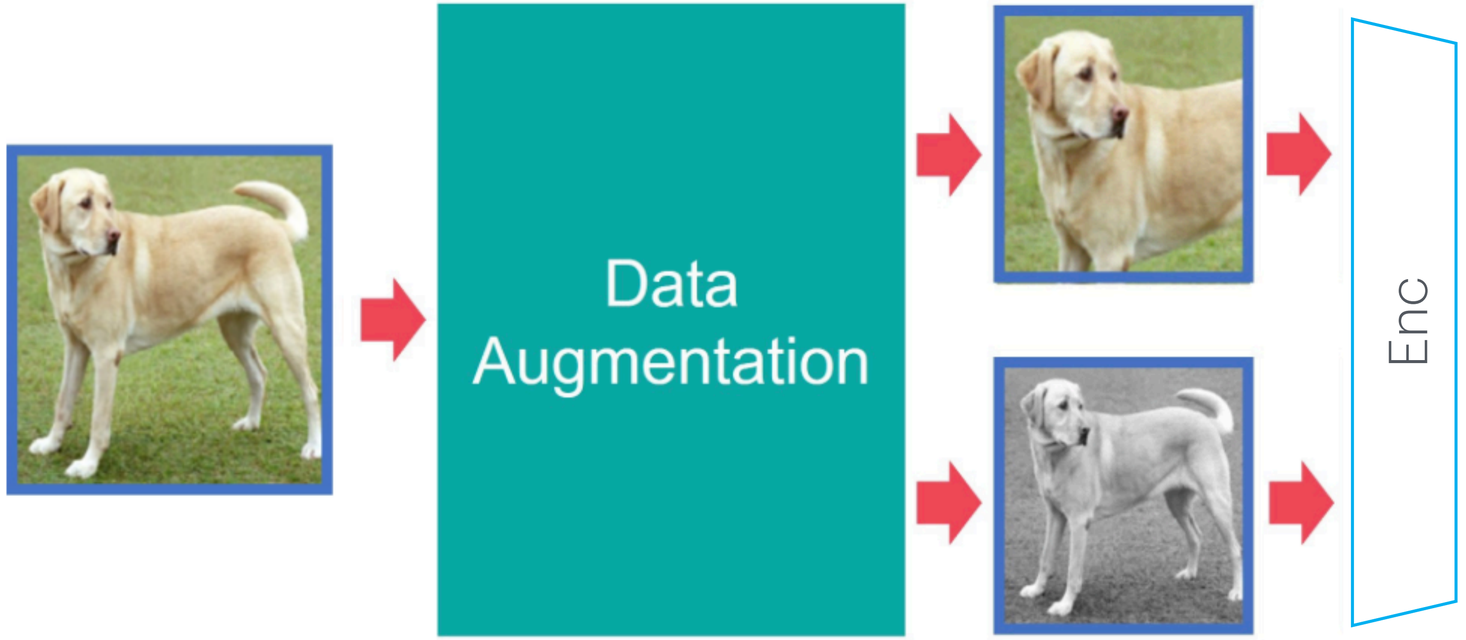
Sample representation:  —MinHash→ $\{x_i\}_{i=1}^{s}$

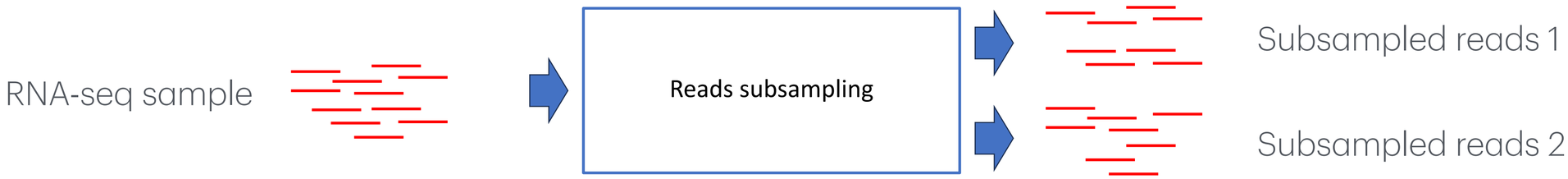
Encoder design: $$z = Enc_{\phi}\left(\{x_i\}_{i=1}^{s}\right) = g_{\psi}\left(\frac{1}{s}\sum_{i=1}^{s} h_{\varphi}(x_i)\right)$$

Permutation-invariant similar to Zander et al. (2017)

Loss function: $$\mathscr{L} = \sum_{i,j}\left(\frac{z_i^T z_j}{\|z_i\|\,\|z_j\|} - \mathbb{S}(i,j)\right)^2$$

Data augmentation will improve encoder training

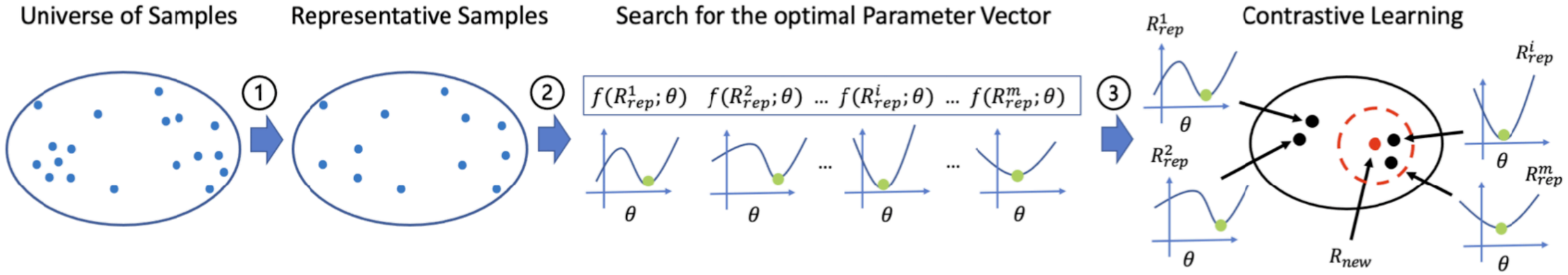




$$\mathbb{S}(i_{\text{sub}}, j_{\text{sub}}) = \mathbb{S}(i, j)$$

Figure credit: https://zhenyuanlu.com/projects/clVR/

Our experiments draw from nearly 200,000 human RNA-seq samples



196,523 human RNA-seq samples from Sequence Read Archive

~1500 samples

Transcript assemblers: Scallop & StringTie
CAWarm-BO with
CASMOPLITAN (Wan et al., 2021)

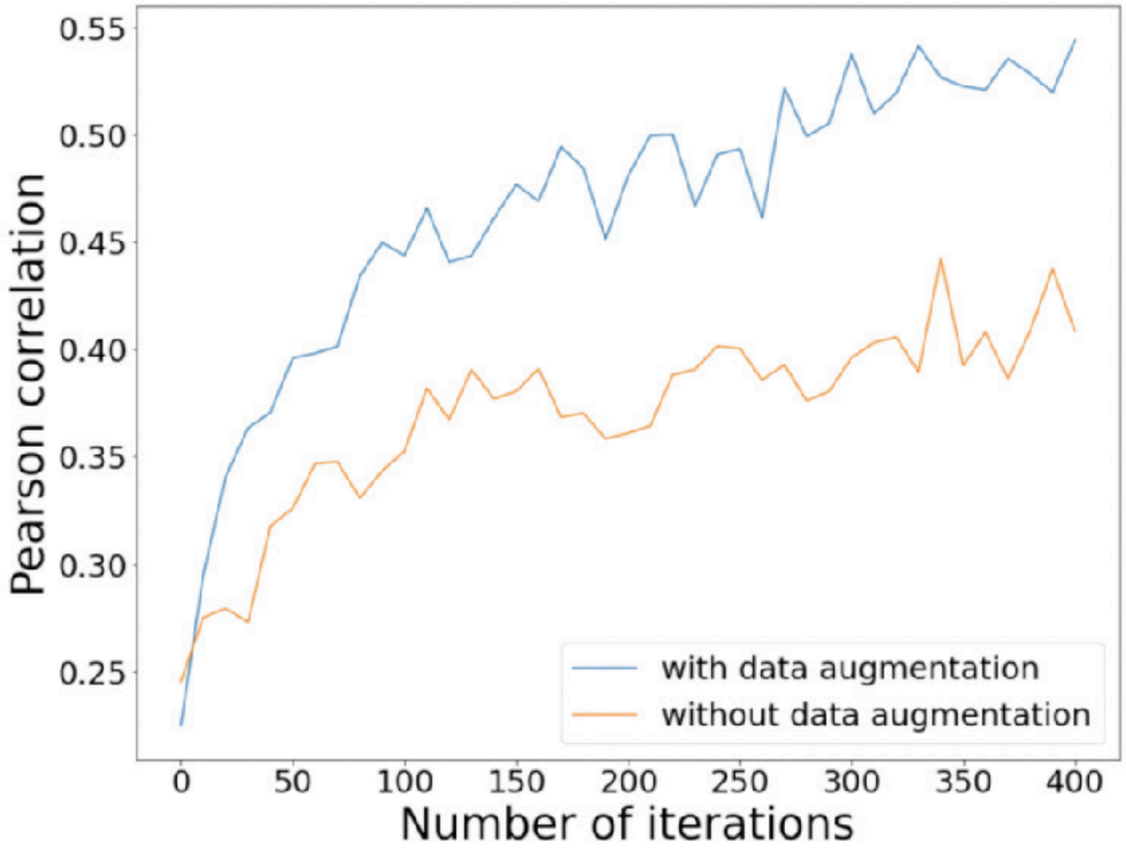
~200 iterations

Fully connected NNs
ADAM: 1×10^{-3}

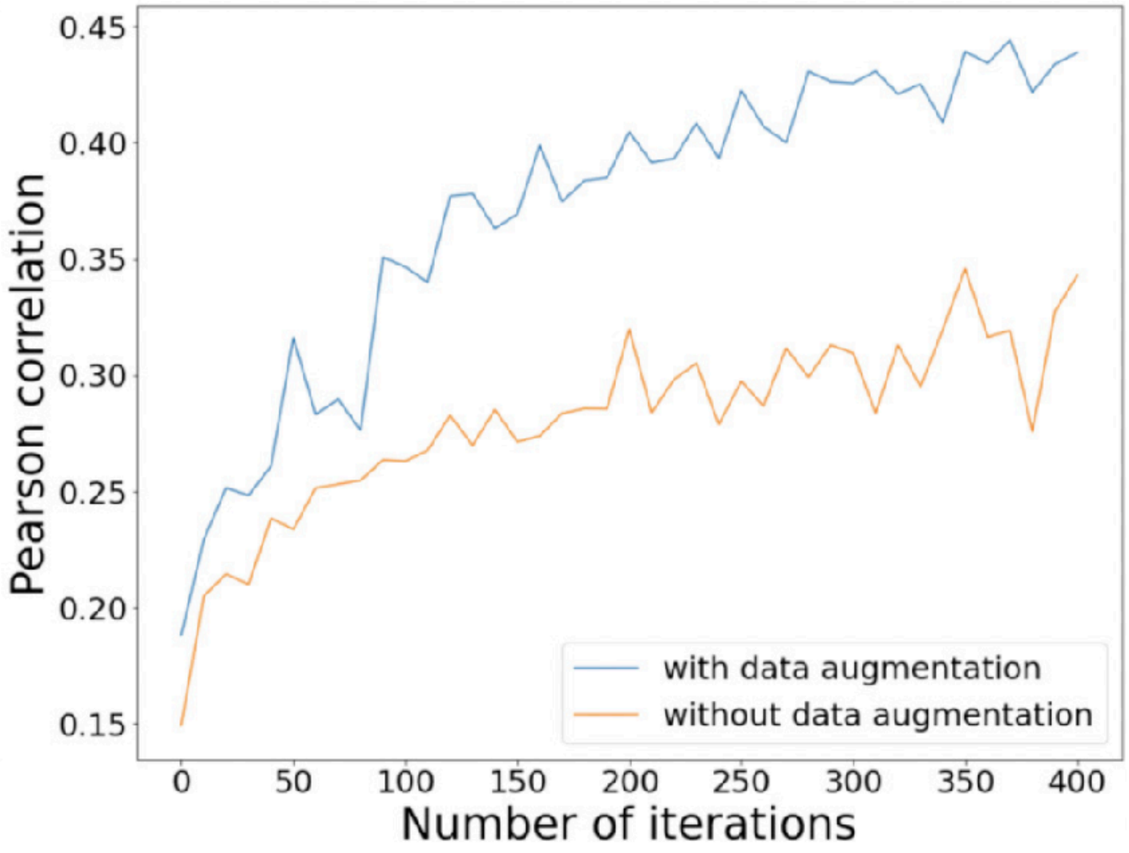
≈17 days using around 300 CPUs

≈ 10 mins to train, < 10 mins to apply on 1 GPU

Data augmentation significantly improves accuracy



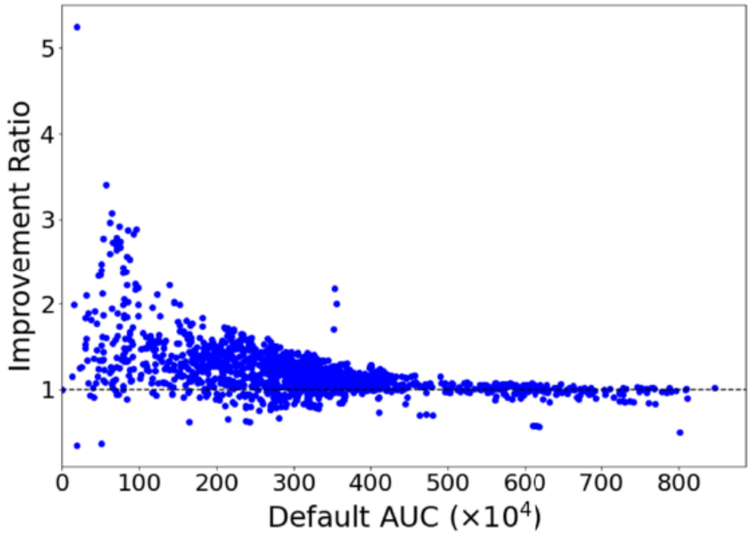
Scallop



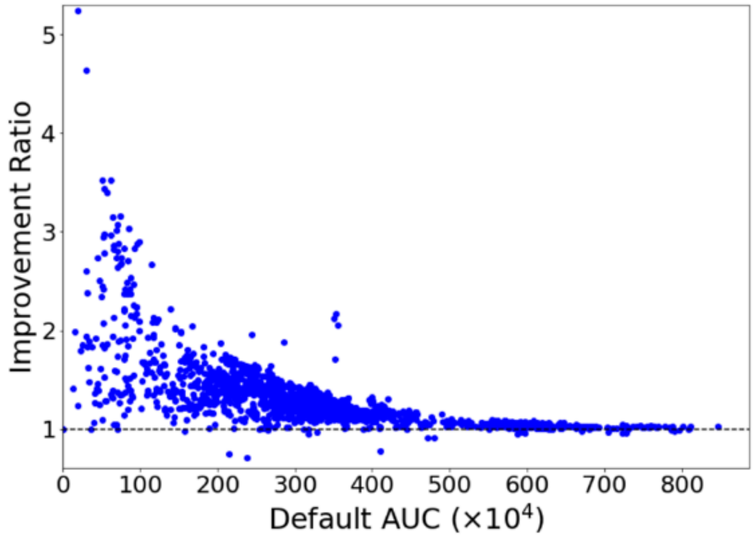
StringTie

The framework significantly improves accuracy of transcript assembly

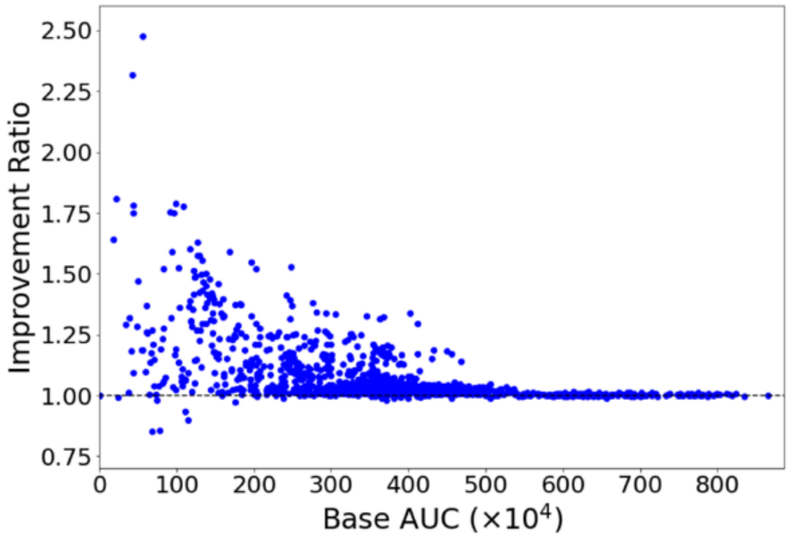
1595 RNA-seq samples
Using the Scallop assembler (Shao & Kingsford, 2017)



Top 1



Top 5



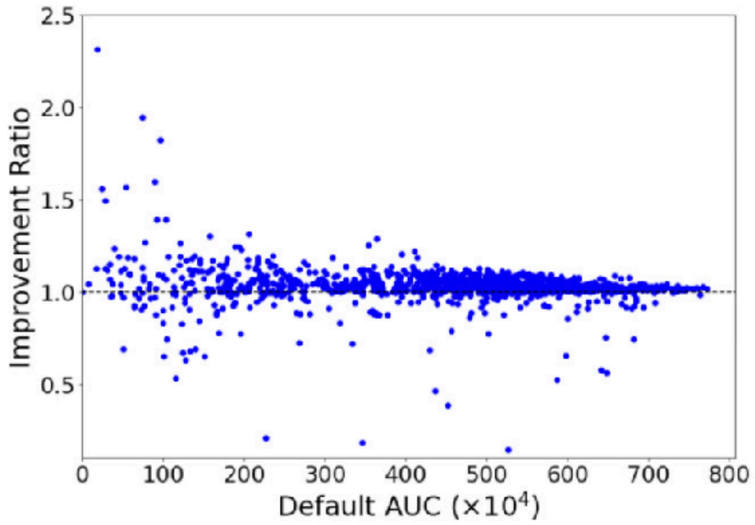
Top 30

Comparison to Default Parameters

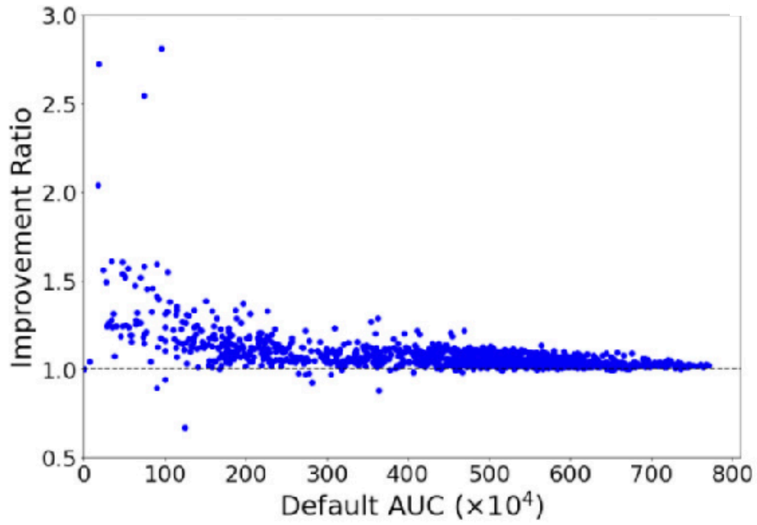
Comparison to DeBlasio et al. 2020

Improvements are observed across multiple assemblers

1595 RNA-seq samples
Using the **StringTie** assembler (Pertea et al., 2015)

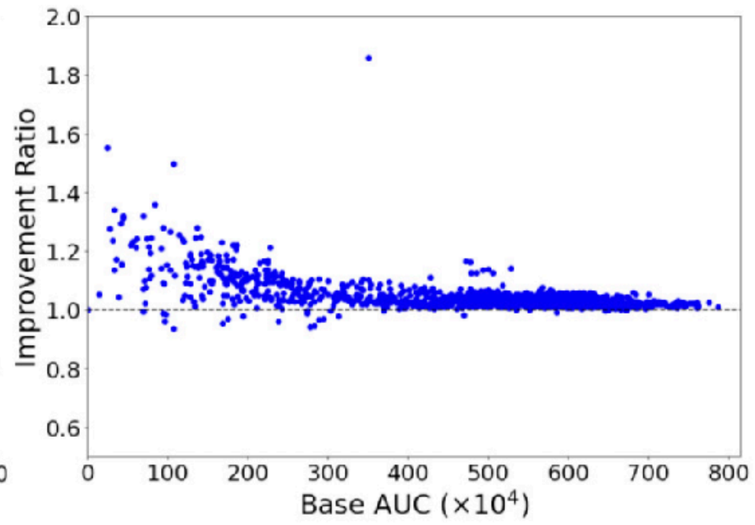


Top 1



Top 5

Comparison to Default Parameters



Top 30

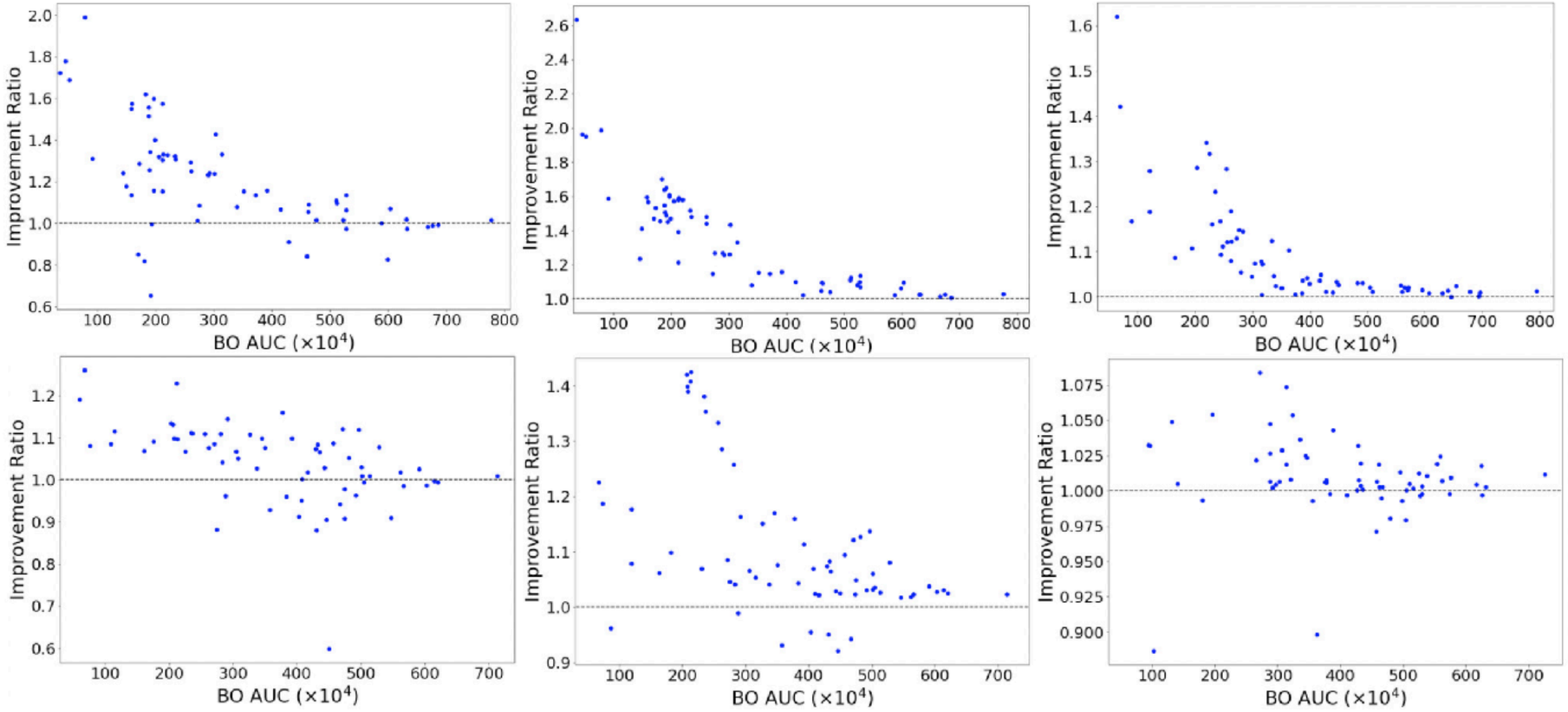
Comparison to DeBlasio et al. 2020

The framework provides improvements beyond simple application of Bayesian Optimization

65 ENCODE samples (not in training set).

Scallop

StringTie



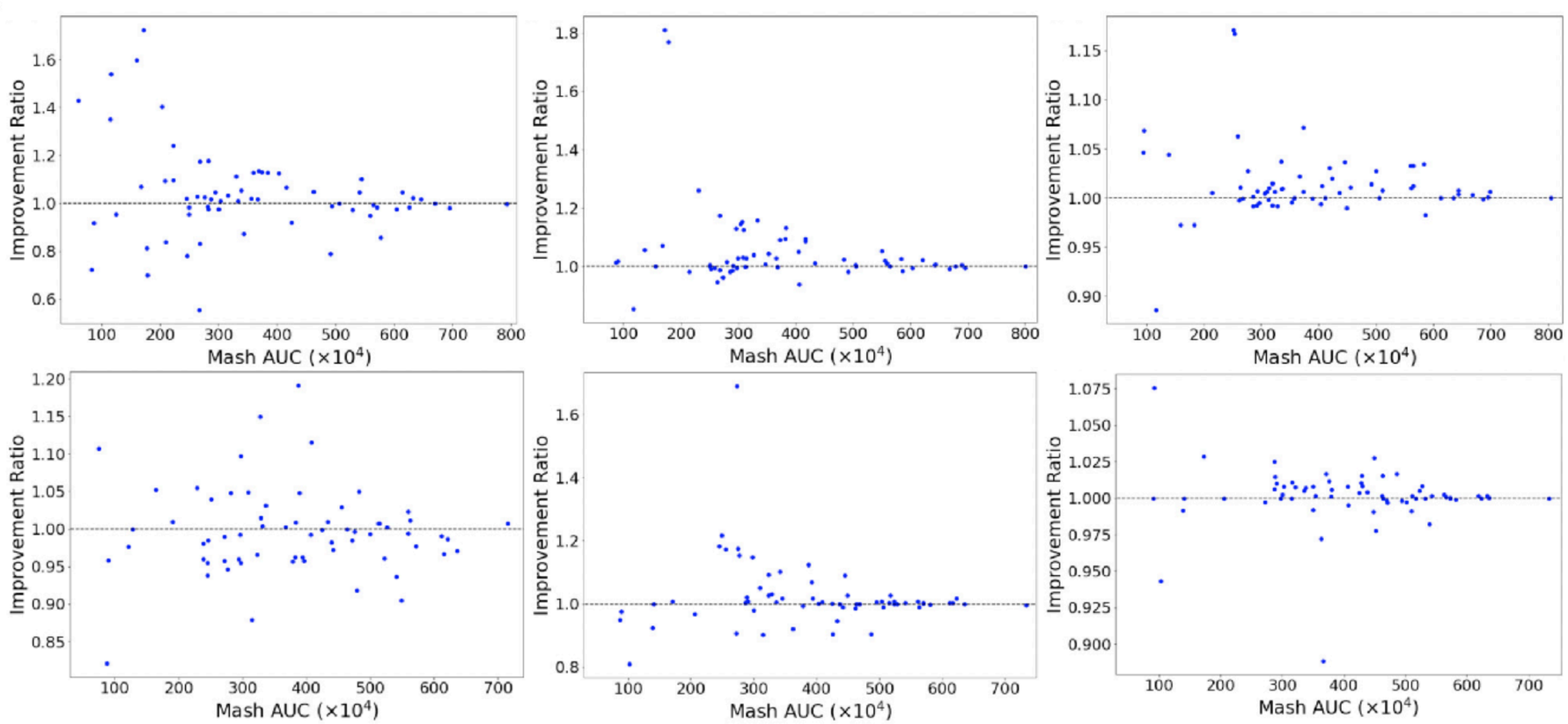
Top 1 Top 5 Top 30

The framework outperforms Mash-based nearest neighbor

65 ENCODE samples (not in training set).

Scallop

StringTie



Top 1 Top 5 Top 30

Take aways from Part 2

- New CA-Warmup-BO approach for adjusting the search domain for Bayesian Optimization.
- Framework for using large numbers of experiments to inform the choice of hyperparameters for sequencing analyses (esp. transcript assembly).
- New method to use traces from BO to define an distance measure on samples that is specific to a given objective function.
- Leads to sometimes huge gains in accuracy (multiples of the accuracy) across several assemblers.
- Outperforms natural baselines and simpler approaches.