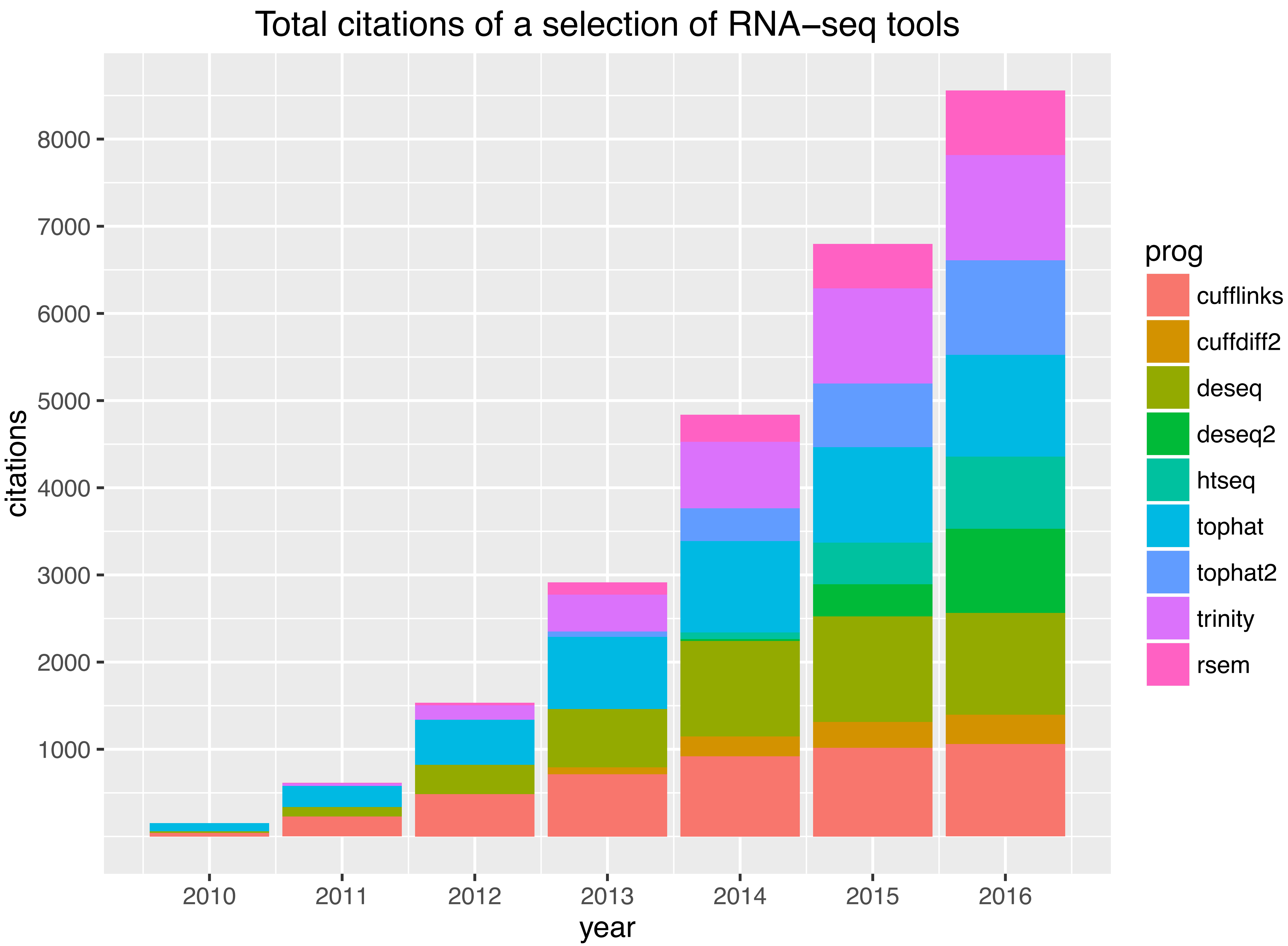
Pseudoalignment & kallisto

Nicolas Bray

The growth of RNA-seq



Deluge of data

- Datasets are growing not only in number but in size and complexity
- Consortia like GTEx generate thousands of samples, while individual biologists can easily generate hundreds of millions of reads worth of RNA-seq data
- Traditional analysis of this data is very computationally intensive, often involving expensive computational resources

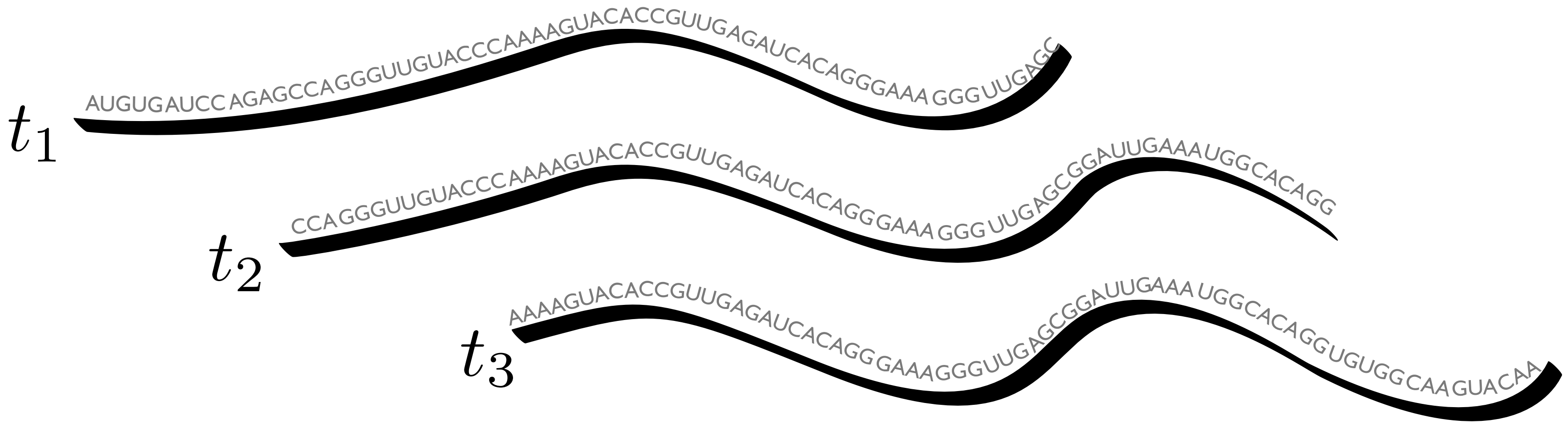
Democratizing analysis

- When analysis requires computational power beyond what's easily available to the average biologist, this adds a barrier between them and their data
- The ability to analyze their own data can reduce dependence on external support
- The ability to explore their data computational can lead to new questions and new discoveries

Usability

- We want data analysis to be not just *possible* but *usable*
- Analysis should make it easy to say “What if…”
- When an analysis takes huge amounts of computer time, it limits exploration
- Personal anecdote: I once waited two weeks for an analysis of a particularly large RNA-seq dataset to finish, only to have a new transcriptome annotation be released the next day

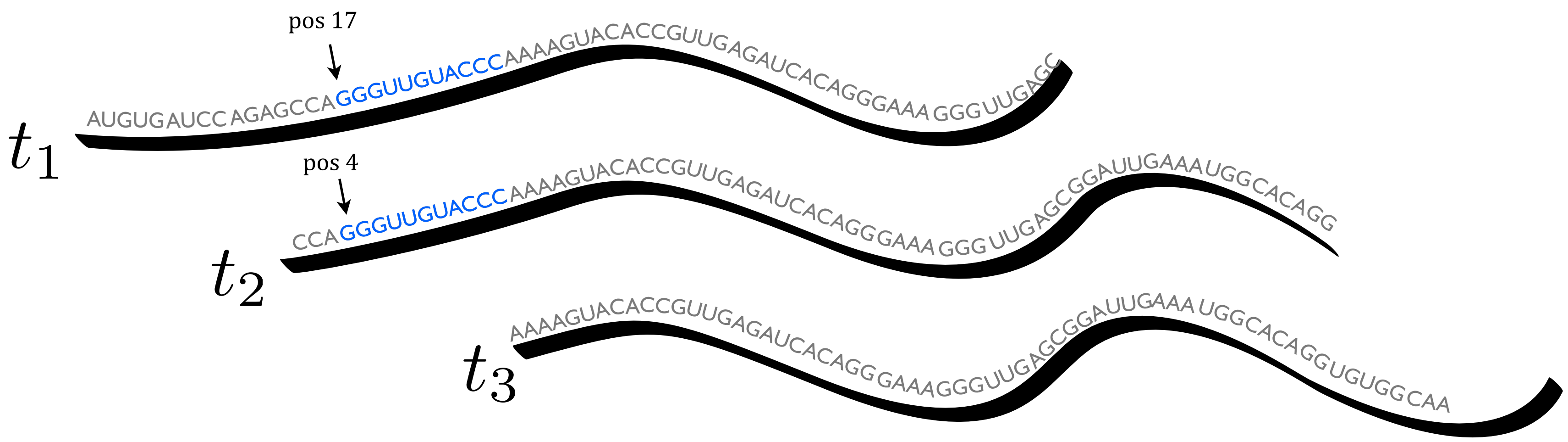
Alignment based analysis


t_1
AUGUGAUCCAGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGC
t_2
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGG
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGGUGUGGCAAGUACAA

read alignment information

read 1	GGGTTGTACCC
read 2	ATGTGATCC
read 3	CCGTTG
read 4	GAAAGGGTTG
read 5	CACAGGTGTGG

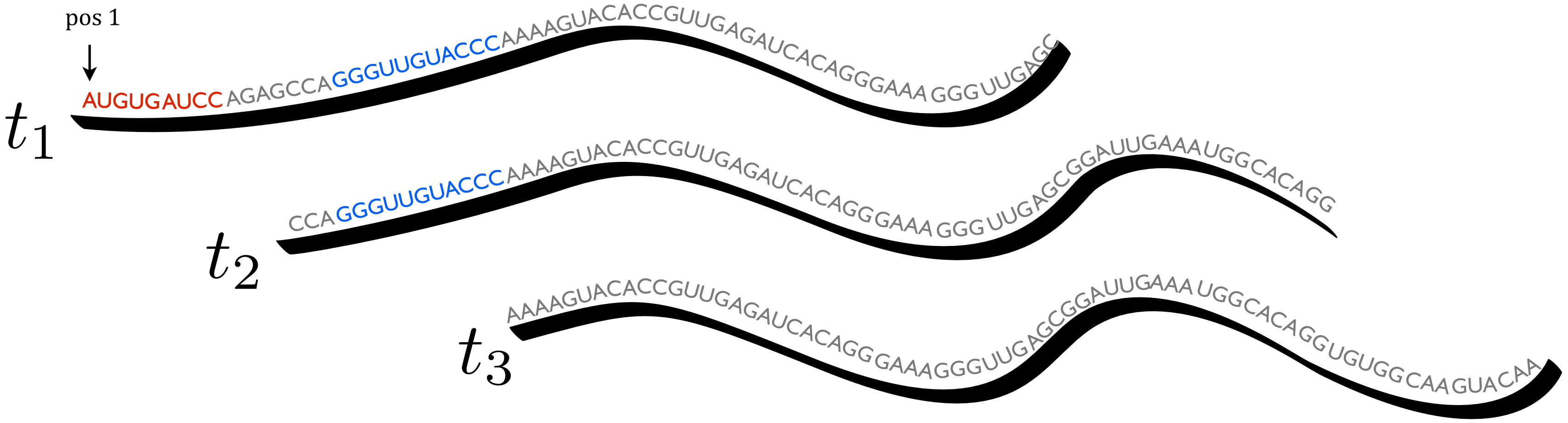
Alignment based analysis


pos 17
AUGUGAUCCAGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGC
t_1
pos 4
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGG
t_2
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGGUGUGGCAA
t_3

read alignment information

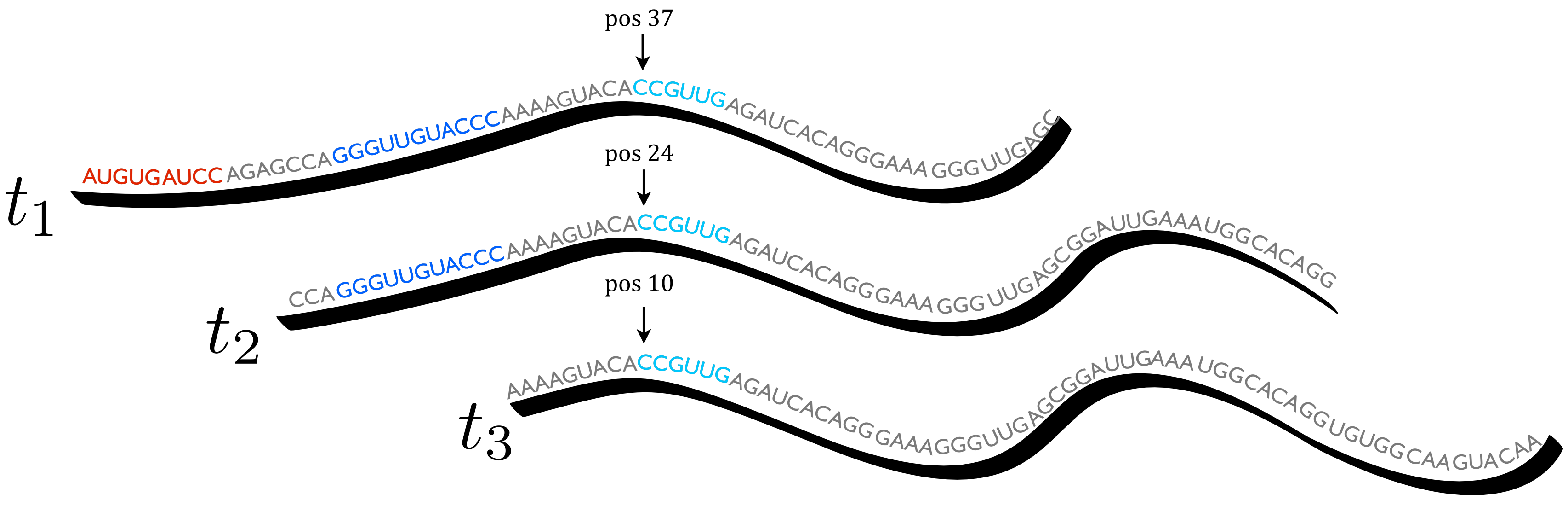
read 1	GGGTTGTACCC	t_1 @position 17, t_2 @position 4
read 2	ATGTGATCC	
read 3	CCGTTG	
read 4	GAAAGGGTTG	
read 5	CACAGGTGTGG	

Alignment based analysis


pos 1
t_1
AUGUGAUCC AGAGCCA GGGUUGUACCC AAAAGUACACCGUUGAGAUCACAGGGAAA GGG UUGAGC
t_2
CCA GGGUUGUACCC AAAAGUACACCGUUGAGAUCACAGGGAAA GGG UUGAGCGGAUUGAAAUGGCACAGG
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAA UGGCACAGGUGUGGCAAGUACAA

		read alignment information
read 1	GGGTTGTACCC	t1 @position 17, t2 @position 4
read 2	ATGTGATCC	t1 @position 1
read 3	CCGTTG	
read 4	GAAAGGGTTG	
read 5	CACAGGTGTGG	

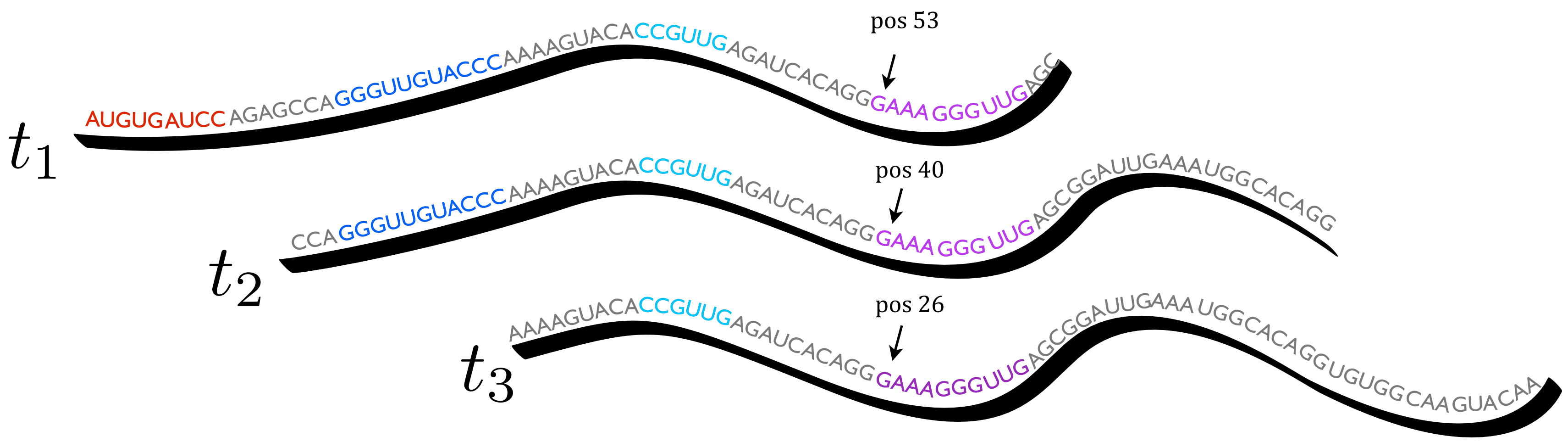
Alignment based analysis


pos 37
pos 24
pos 10
t_1
AUGUGAUCC AGAGCCA GGGUUGUACCC AAAAGUACA CCGUUG AGAUCACAGGGAAA GGGUUGAGC
t_2
CCA GGGUUGUACCC AAAAGUACA CCGUUG AGAUCACAGGGAAA GGG UUGAGCGGAUUGAAAUGGCACAGG
t_3
AAAAGUACA CCGUUG AGAUCACAGG GAAAGGGUUGAGCGGAUUGAAA UGGCACAGGUGUGGCAAGUACAA

read alignment information

read 1	GGGTTGTACCC	t1 @position 17, t2 @position 4
read 2	ATGTGATCC	t1 @position 1
read 3	CCGTTG	t1 @position 37, t2 @position 24, t3 @position 10
read 4	GAAAGGGTTG	
read 5	CACAGGTGTGG	

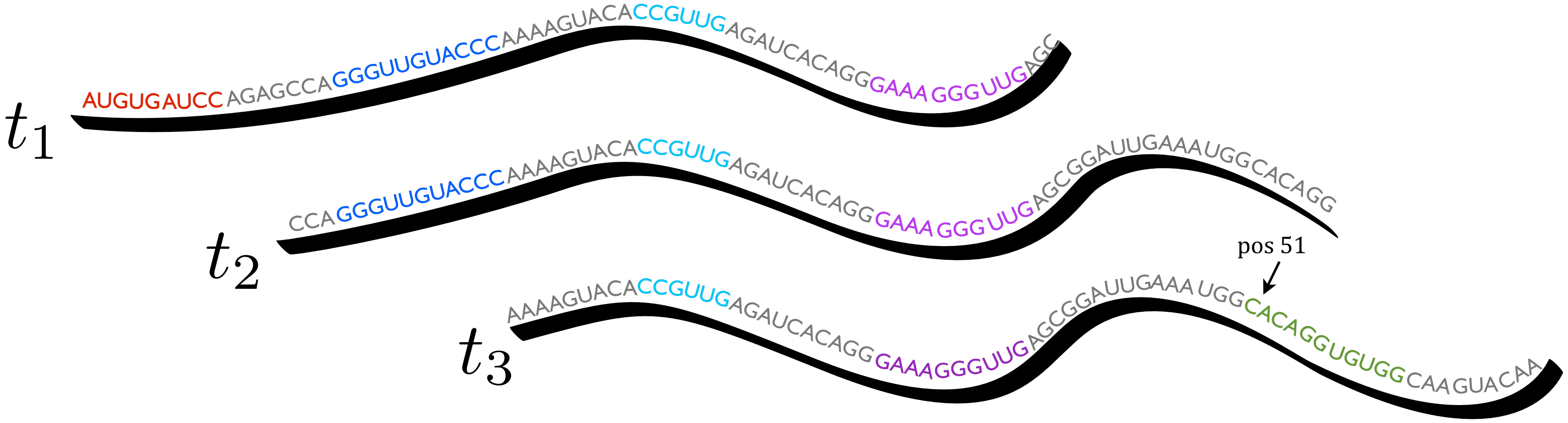
Alignment based analysis


t_1
AUGUGAUCC AGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAA GGGUUGAGC
pos 53
t_2
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAA GGG UUGAGCGGAUUGAAAUGGCACAGG
pos 40
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAA UGGCACAGGUGUGGCAAGUACAA
pos 26

read alignment information

read 1	GGGTTGTACCC	t_1 @position 17, t_2 @position 4
read 2	ATGTGATCC	t_1 @position 1
read 3	CCGTTG	t_1 @position 37, t_2 @position 24, t_3 @position 10
read 4	GAAAGGGTTG	t_1 @position 53, t_2 @position 40, t_3 @position 26
read 5	CACAGGTGTGG	

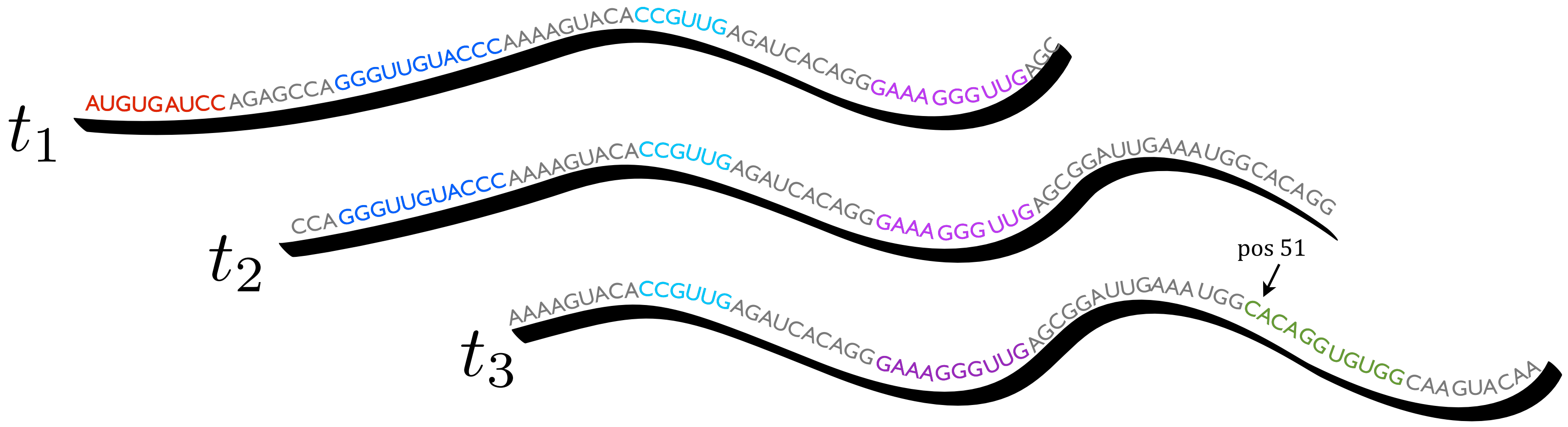
Alignment based analysis


t_1
AUGUGAUCCAGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGC
t_2
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGG
pos 51
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGGUGUGGCAAGUACAA

read alignment information

read 1	GGGTTGTACCC	t_1 @position 17,	t_2 @position 4	
read 2	ATGTGATCC	t_1 @position 1		
read 3	CCGTTG	t_1 @position 37,	t_2 @position 24,	t_3 @position 10
read 4	GAAAGGGTTG	t_1 @position 53,	t_2 @position 40,	t_3 @position 26
read 5	CACAGGTGTGG			t_3 @position 51

Alignment based analysis

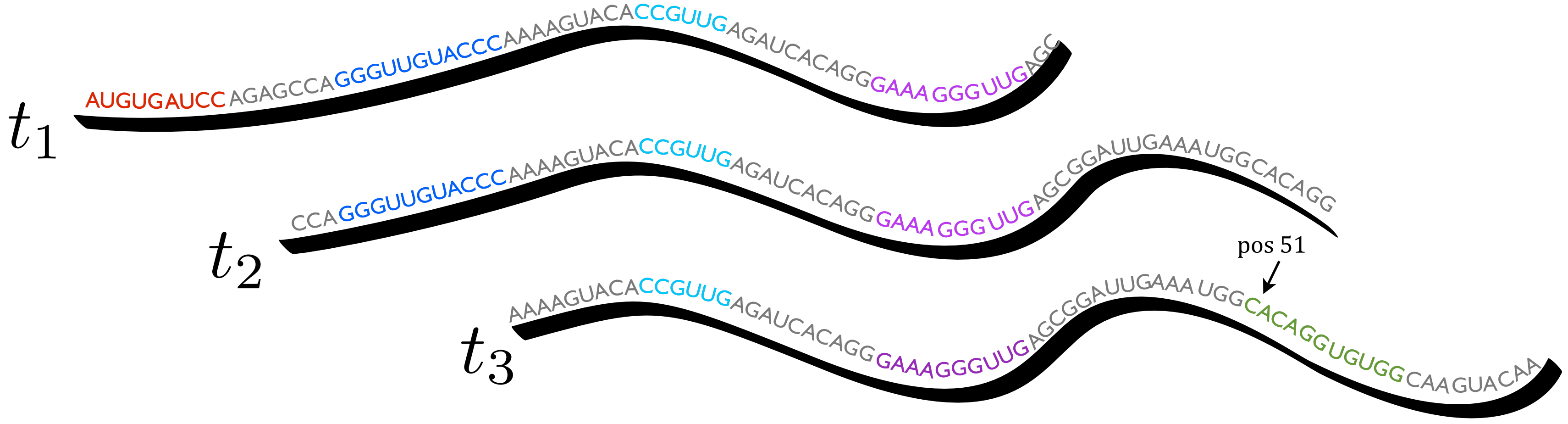

t_1
AUGUGAUCC AGAGCCA GGGUUGUACCC AAAAGUACA CCGUUG AGAUCACAGG GAAA GGG UUG AGC
t_2
CCA GGGUUGUACCC AAAAGUACA CCGUUG AGAUCACAGG GAAA GGG UUG AGCGGAUUGAAAUGGCACAGG
t_3
AAAAGUACA CCGUUG AGAUCACAGG GAAAGGGUUG AGCGGAUUGAAA UGG CACAGGUGUGG CAAGUACAA
pos 51

read alignment information

read 1	GGGTTGTACCC	t_1 @position 17, t_2 @position 4
read 2	ATGTGATCC	t_1 @position 1
read 3	CCGTTG	t_1 @position 37, t_2 @position 24, t_3 @position 10
read 4	GAAAGGGTTG	t_1 @position 53, t_2 @position 40, t_3 @position 26
read 5	CACAGGTGTGG	t_3 @position 51

- Even ultra-fast alignment is still pretty slow
- Alignments contain information that we don’t usually care about.

The kallisto mantra

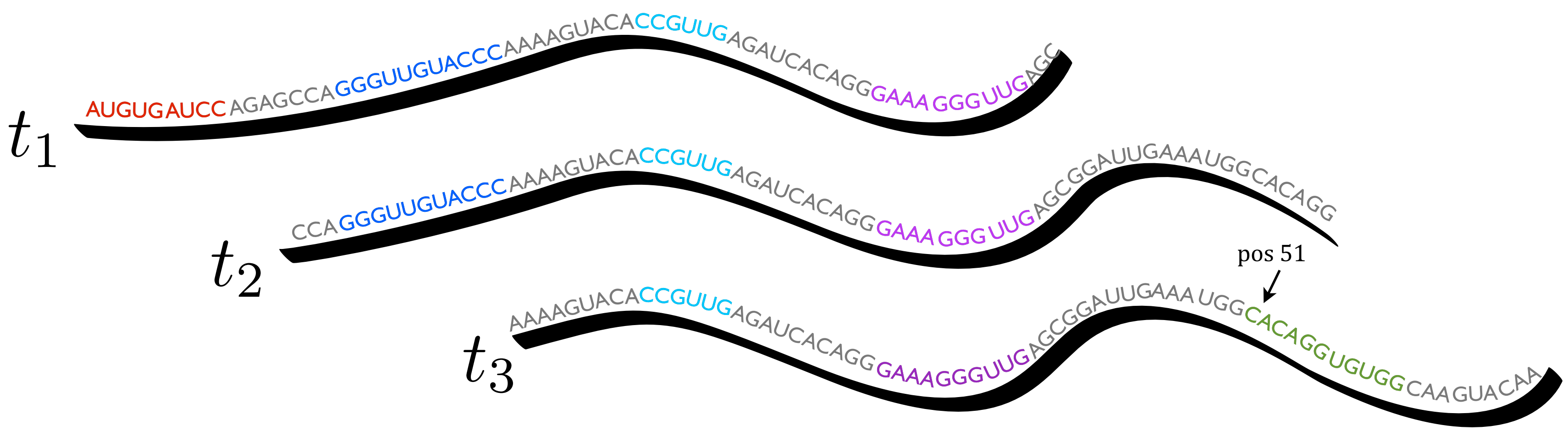

t_1
AUGUGAUCCAGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGC
t_2
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGG
pos 51
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGGUGUGGCAAGUACAA

read alignment information

read 1	GGGTTGTACCC	t1 @position 17, t2 @position 4
read 2	ATGTGATCC	t1 @position 1
read 3	CCGTTG	t1 @position 37, t2 @position 24, t3 @position 10
read 4	GAAAGGGTTG	t1 @position 53, t2 @position 40, t3 @position 26
read 5	CACAGGTGTGG	t3 @position 51

Do as much as you can, with as little as you can.

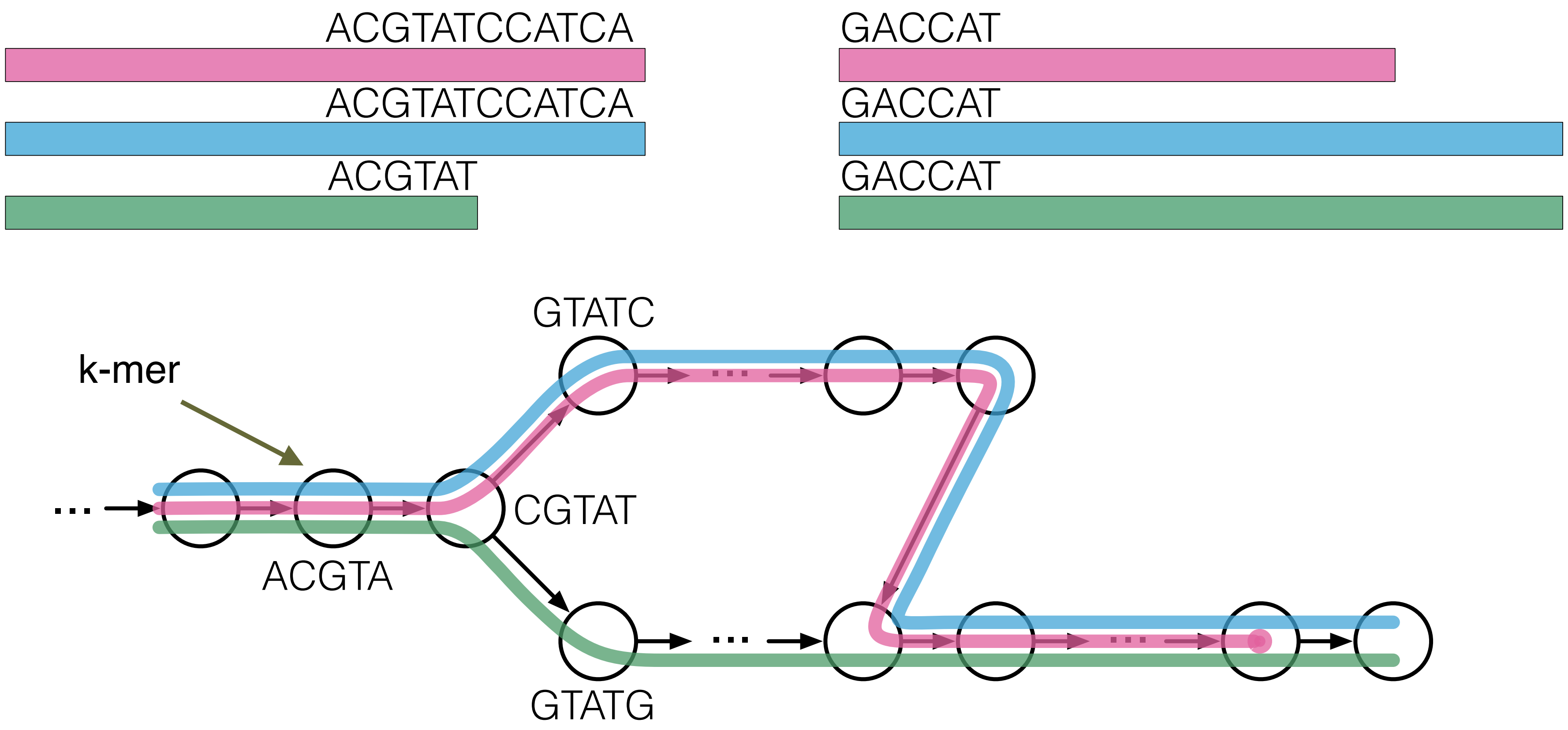
The kallisto mantra


t_1
AUGUGAUCCAGAGCCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGC
t_2
CCAGGGUUGUACCCAAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGG
pos 51
t_3
AAAAGUACACCGUUGAGAUCACAGGGAAAGGGUUGAGCGGAUUGAAAUGGCACAGGUGUGGCAAGUACAA

		read alignment information	equivalence class	read counts in each class
read 1	GGGTTGTACCC	t1 @position 17, t2 @position 4	{t1, t2}	S1: 1
read 2	ATGTGATCC	t1 @position 1	{t1}	S2: 1
read 3	CCGTTG	t1 @position 37, t2 @position 24, t3 @position 10	{t1, t2, t3}	S3: 2
read 4	GAAAGGGTTG	t1 @position 53, t2 @position 40, t3 @position 26	{t1, t2, t3}	S4: 0
read 5	CACAGGTGTGG	t3 @position 51	{t3}	S5: 1 ←

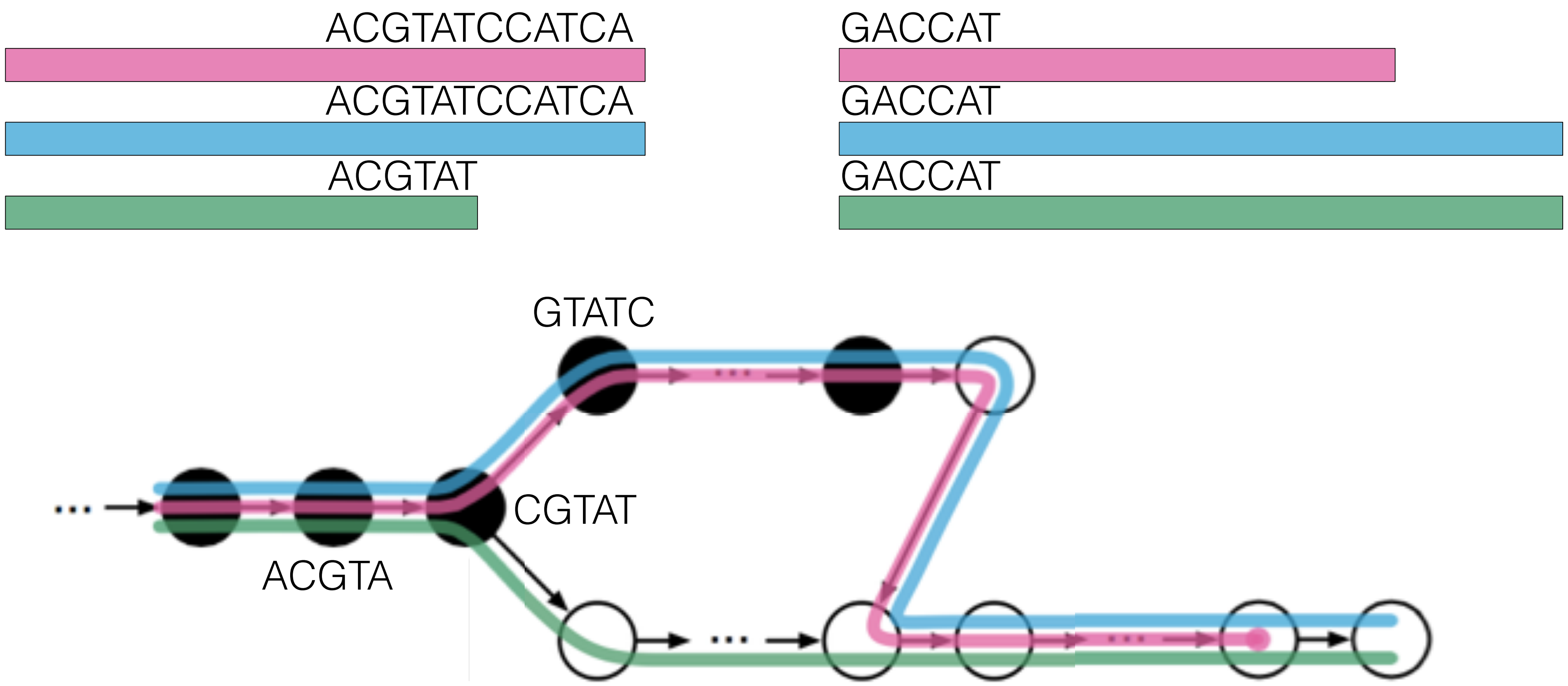
- for computing transcript abundances, the set of transcripts a read is compatible with tells you almost everything about it
- **idea**: let's compute that directly rather than a basepair-level alignment that has more information than we need

How kallisto computes pseudoaligments



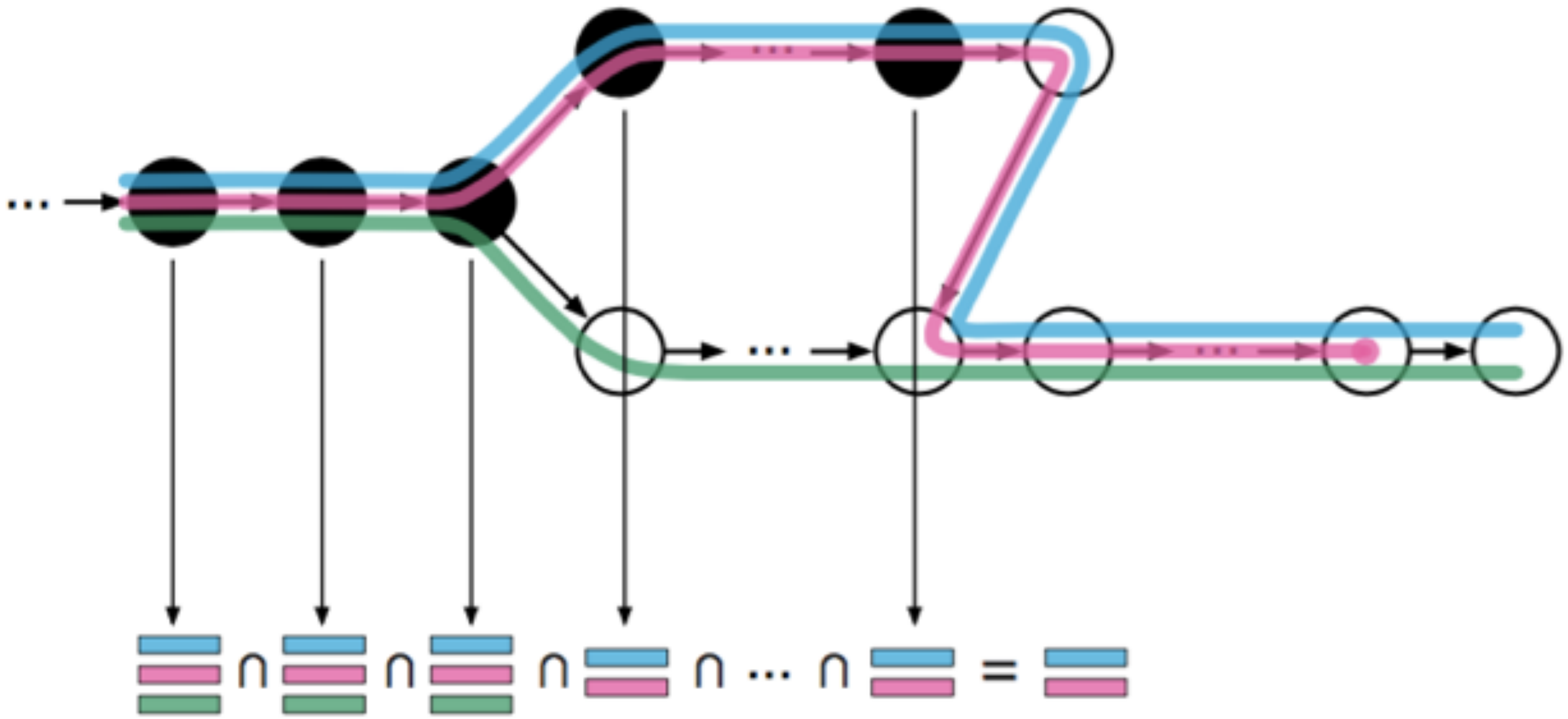
- Given our reference transcriptome, we first construct its *target de Bruijn Graph (T-DBG)*
- This encodes the transcript sequences but also provides information about how they overlap with each other
- Only has to be done *once* per transcriptome (and is fast)

How kallisto computes pseudoaligments



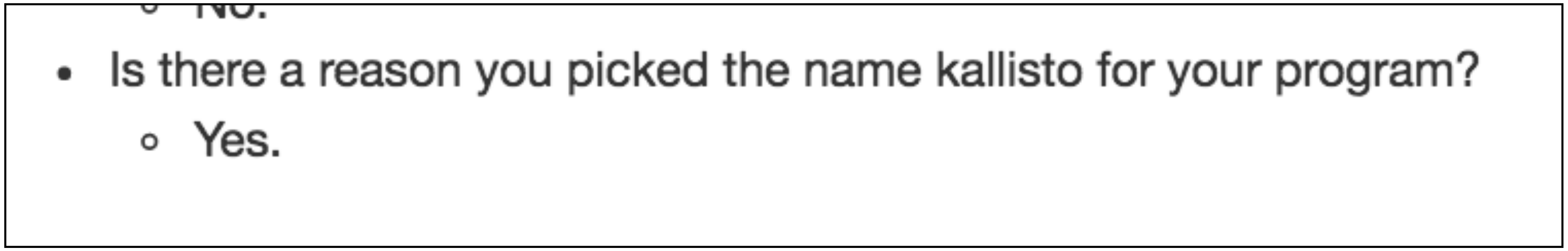
- Given a read, finding its constitutive *k*-mers in the T-DBG gives you information about where the read could have come from
- This can be done *very* fast
- **But individual *k*-mers might be more ambiguous than the read as a whole**

How kallisto computes pseudoaligments



- Combining information across the k-mers can recover lost information
- For each k-mer we have the set of transcripts it could have come from. Intersecting them gives the set of transcripts that *all* k-mers could have come from
- It's possible for their combination to have information equivalent to the entire read, even if no single k-mer does by itself

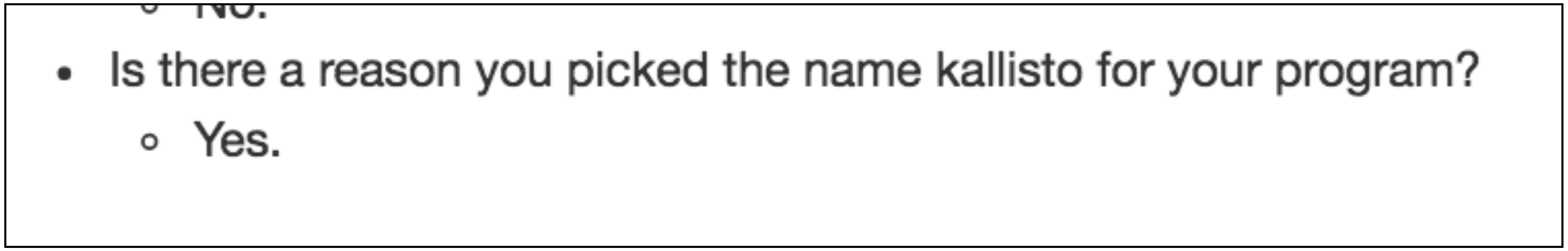
How kallisto computes pseudoaligments



  - No.
- Is there a reason you picked the name kallisto for your program?
  - Yes.

- **k**-mers **a**lone **l**ose **l**ots of **i**nformation; **s**trong **t**ogether **o**nly

How kallisto computes pseudoaligments

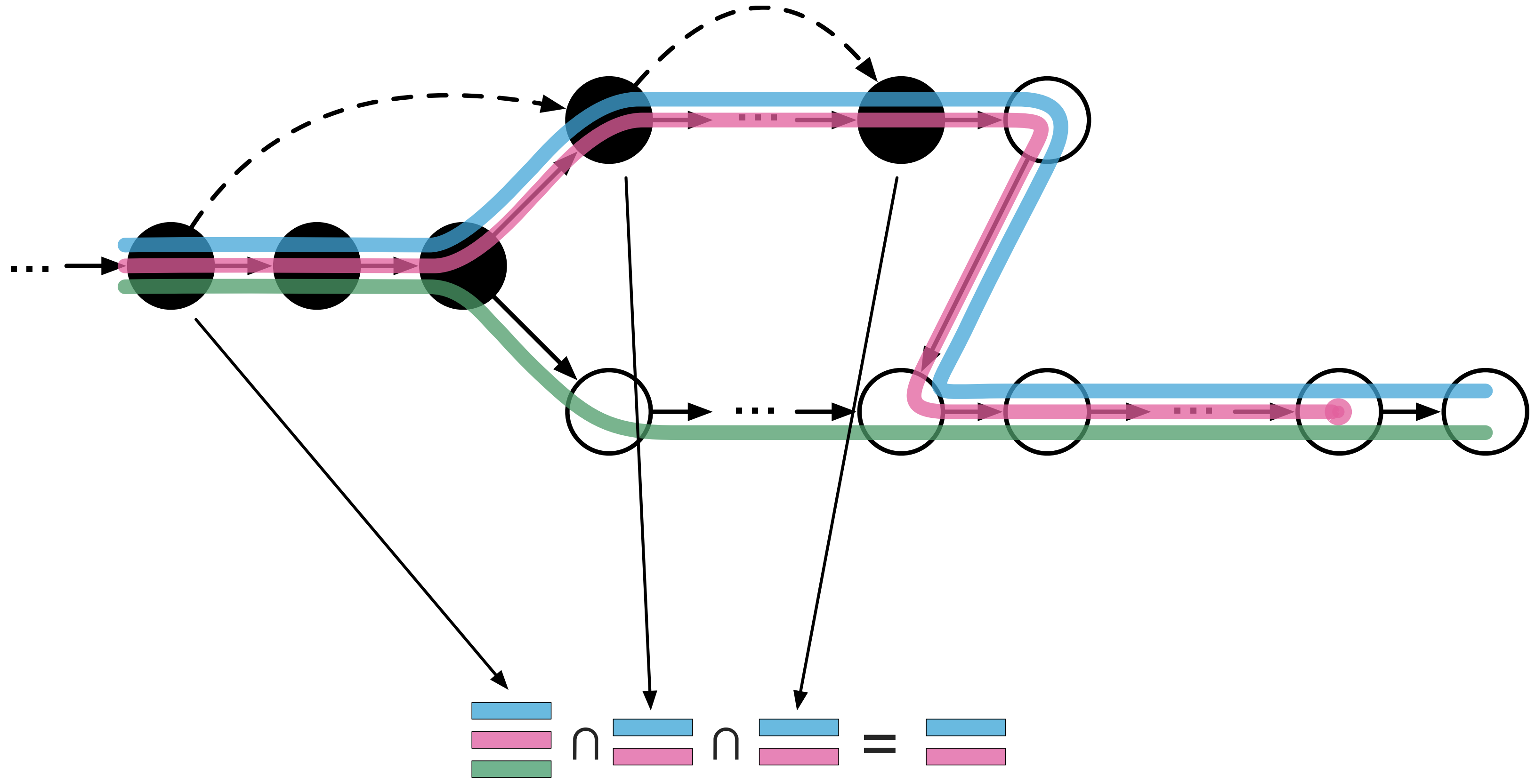


  - No.
- Is there a reason you picked the name kallisto for your program?
  - Yes.

- **k**-mers **a**lone **l**ose **l**ots of **i**nformation; **s**trong **t**ogether **o**nly

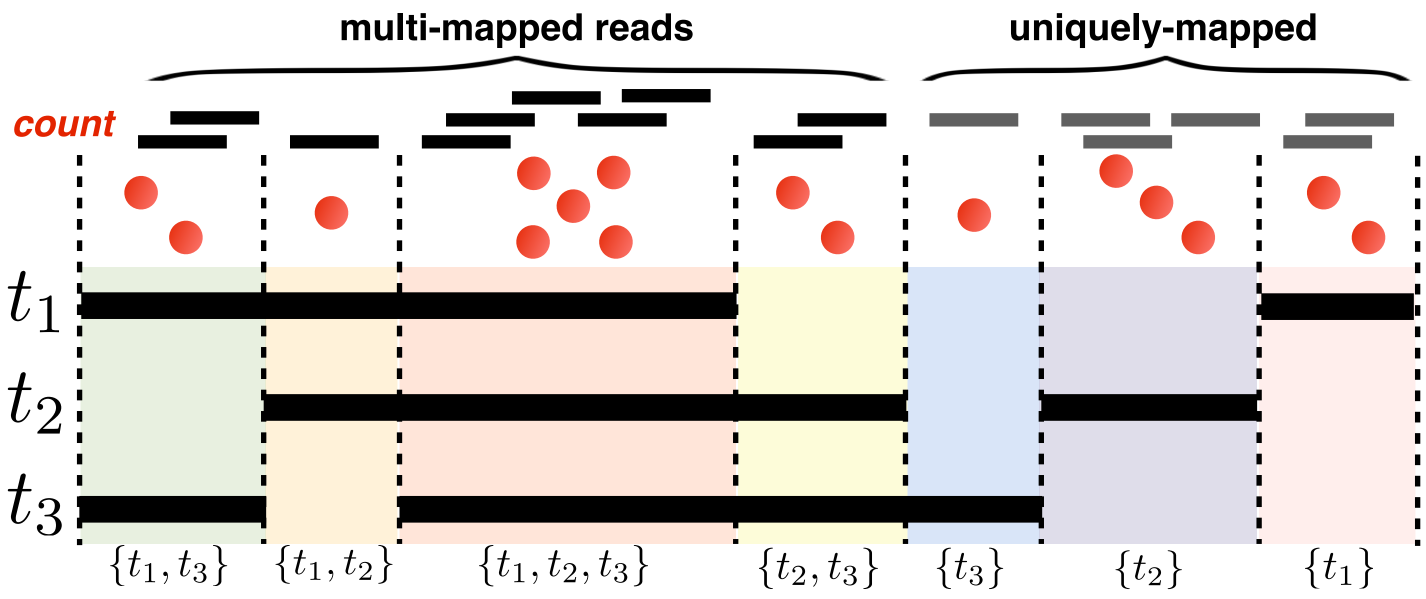
kallisto

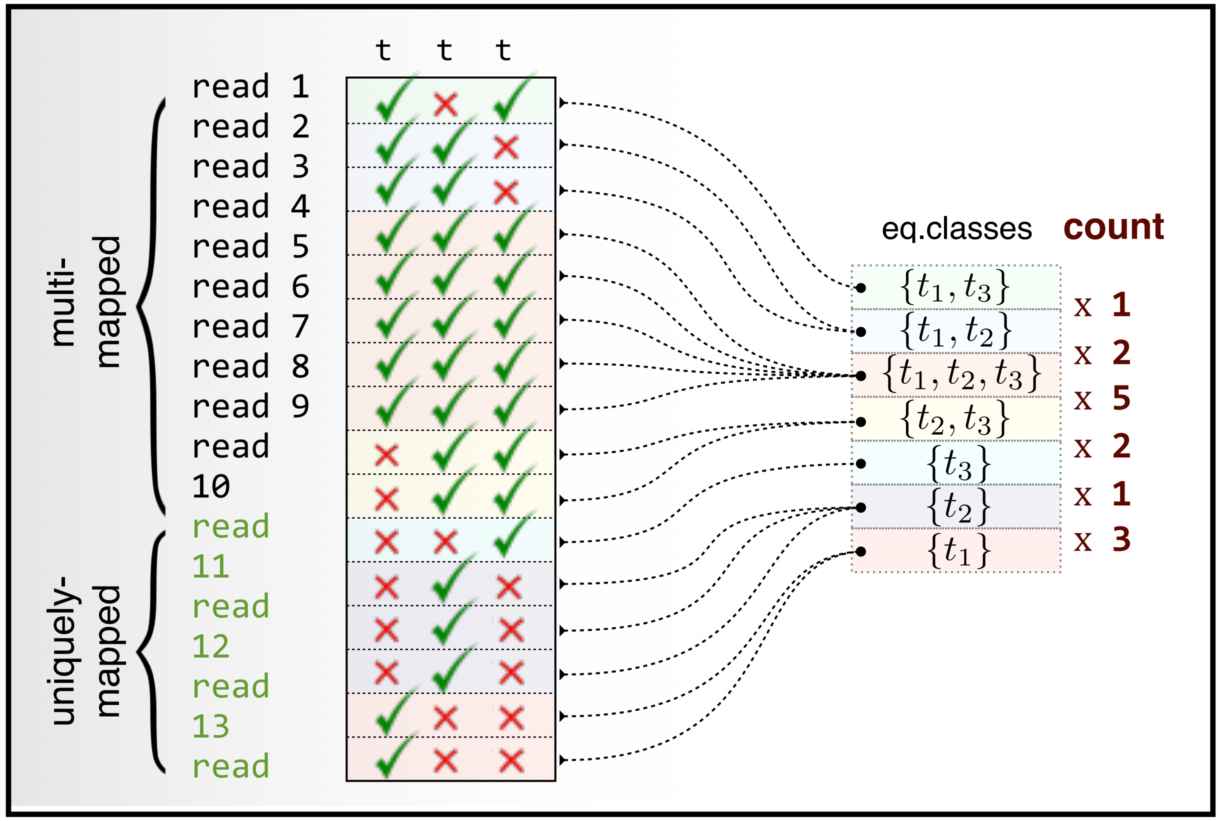
How kallisto computes pseudoaligments



- Knowing the T-DBG, we can predict ahead of time which k-mers will be potentially interesting
- By only processing those k-mers, kallisto runs ~8 times faster

Transcript compatibility counts

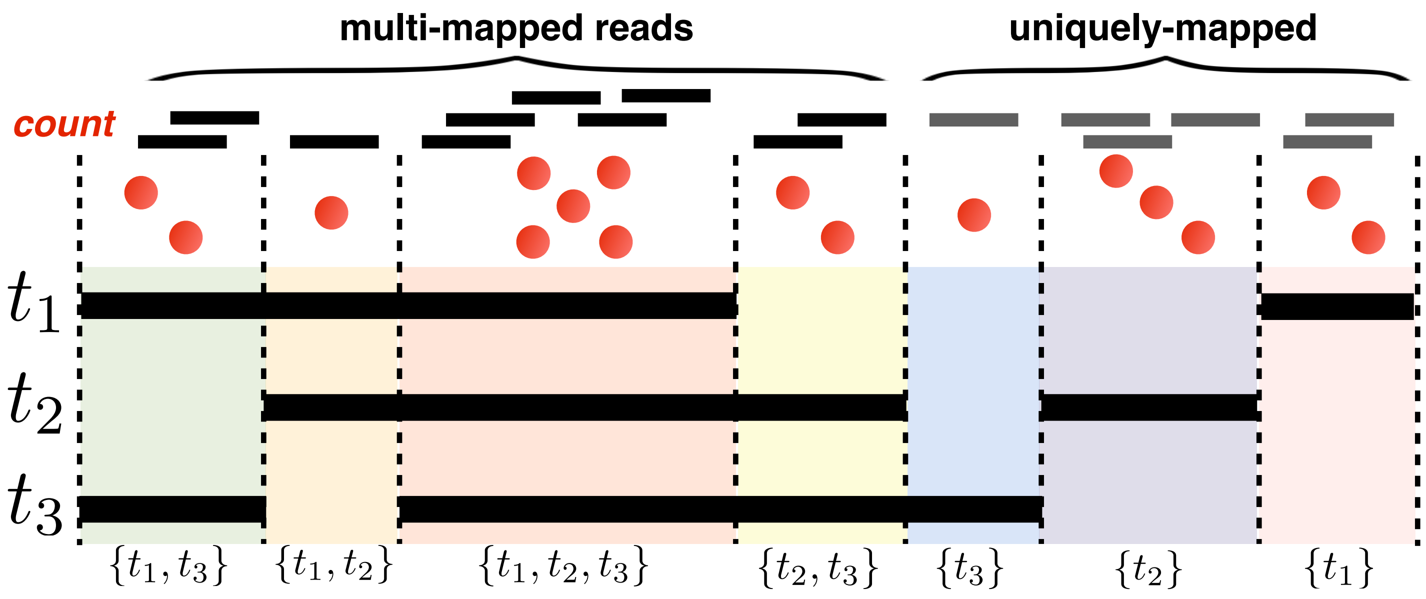


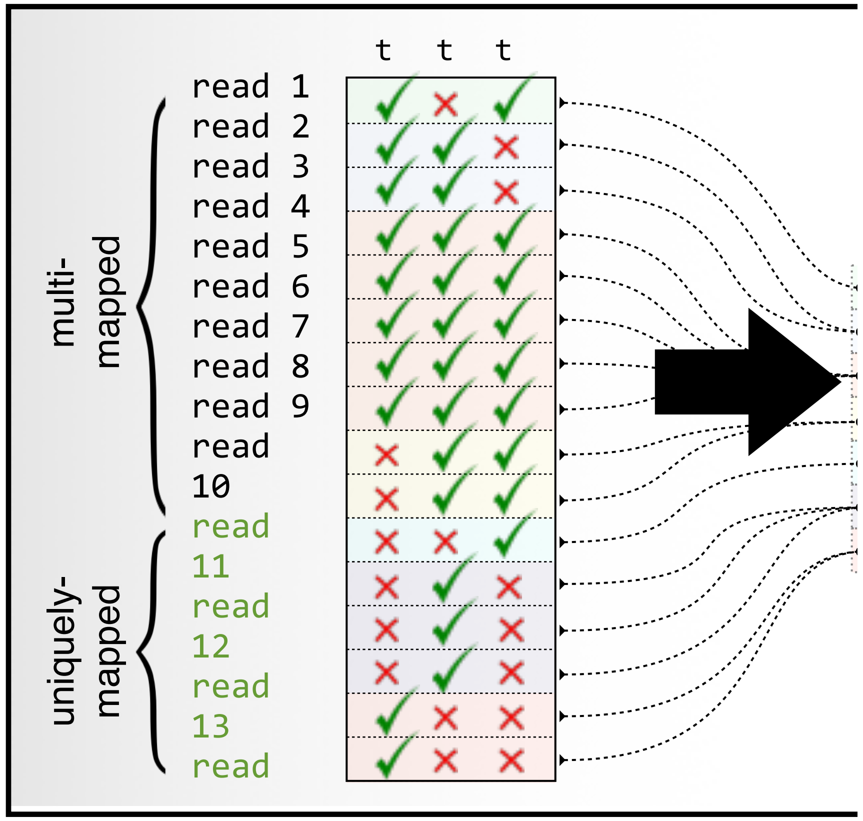


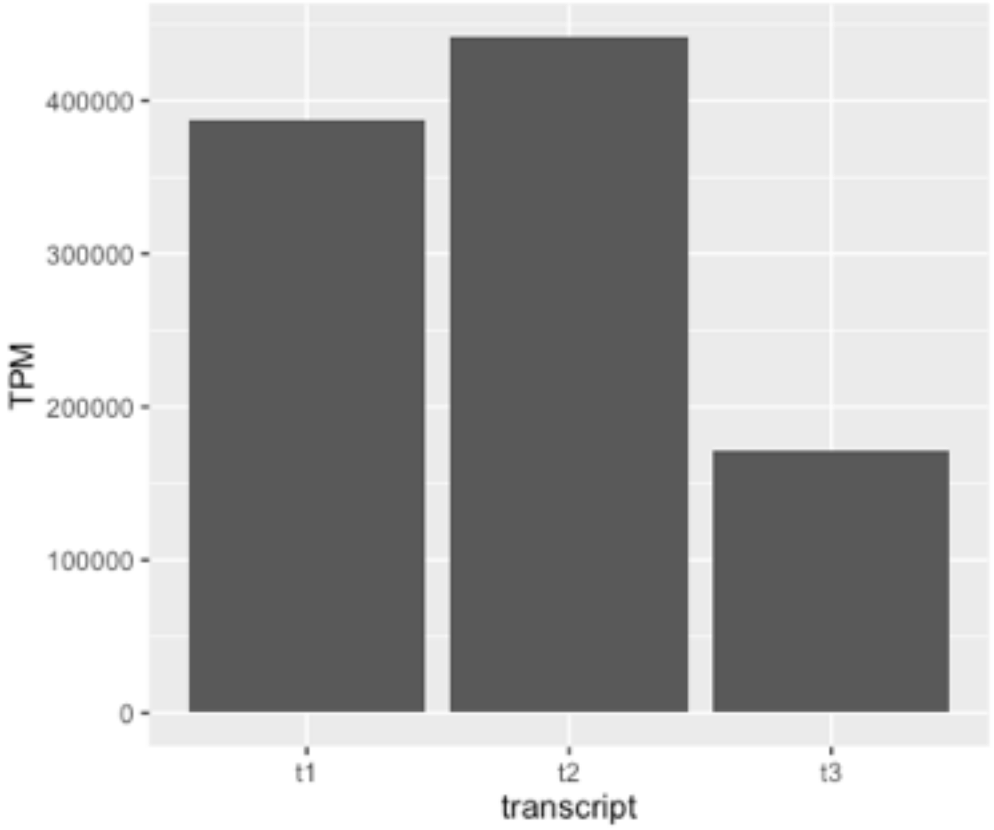
t	t	t
✓	✗	✓
✓	✓	✗
✓	✓	✗
✓	✓	✓
✓	✓	✓
✓	✓	✓
✓	✓	✓
✓	✓	✓
✗	✓	✓
✗	✓	✓
✗	✗	✓
✗	✓	✗
✗	✓	✗
✗	✓	✗
✓	✗	✗
✓	✗	✗

eq.classes	count
$\{t_1, t_3\}$	x 1
$\{t_1, t_2\}$	x 2
$\{t_1, t_2, t_3\}$	x 5
$\{t_2, t_3\}$	x 2
$\{t_3\}$	x 1
$\{t_2\}$	x 3
$\{t_1\}$	

Quantifying transcript abundances







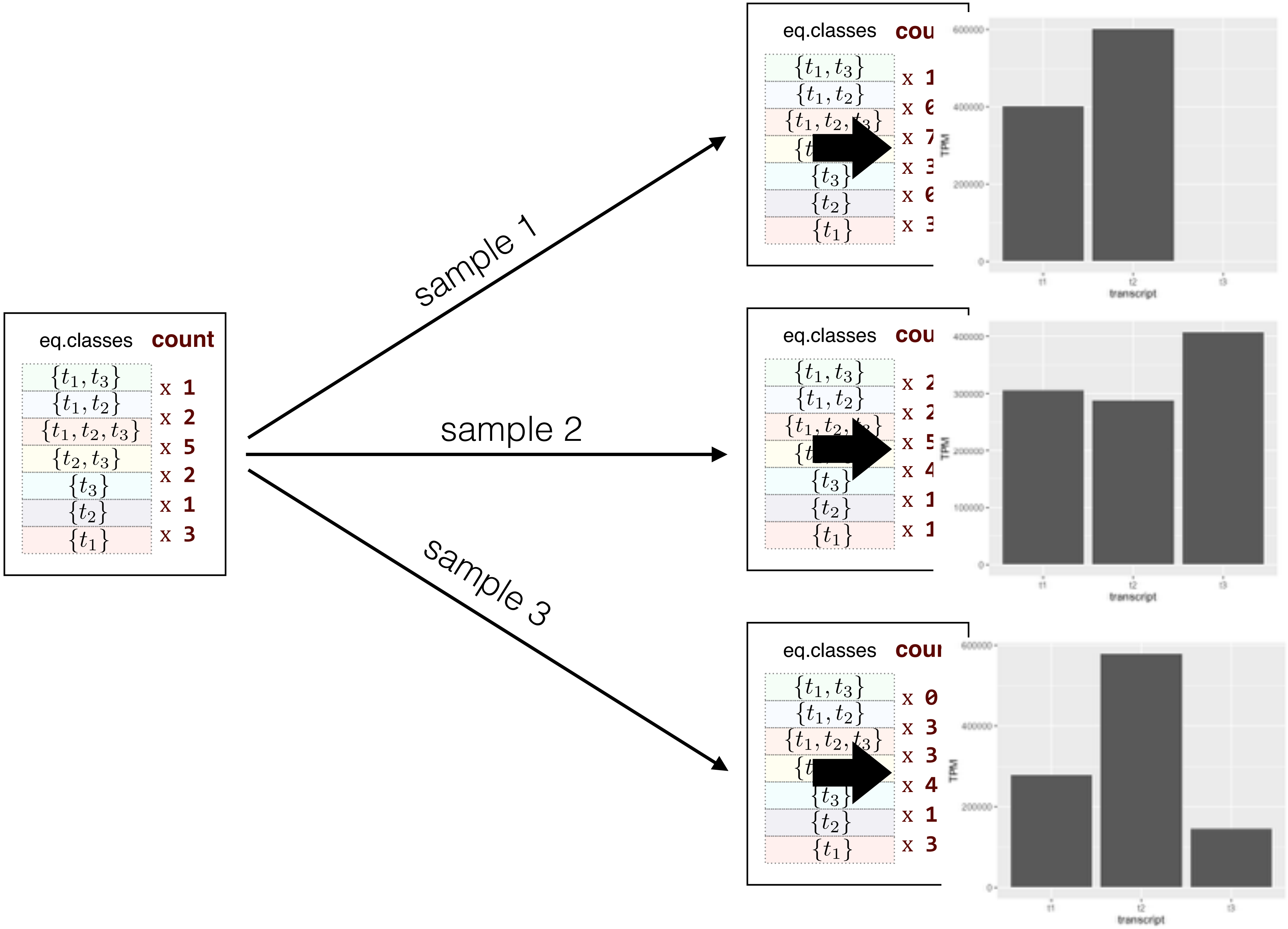
Estimating uncertainty

- “What are the abundances of the different transcripts in my sample?”
- kallisto gives *an* answer but how sure should you be of it?
- In an alternate universe, your sample prep and sequencing might have produced slightly different data for no real biological reason
- What would that data look like?

Estimating uncertainty

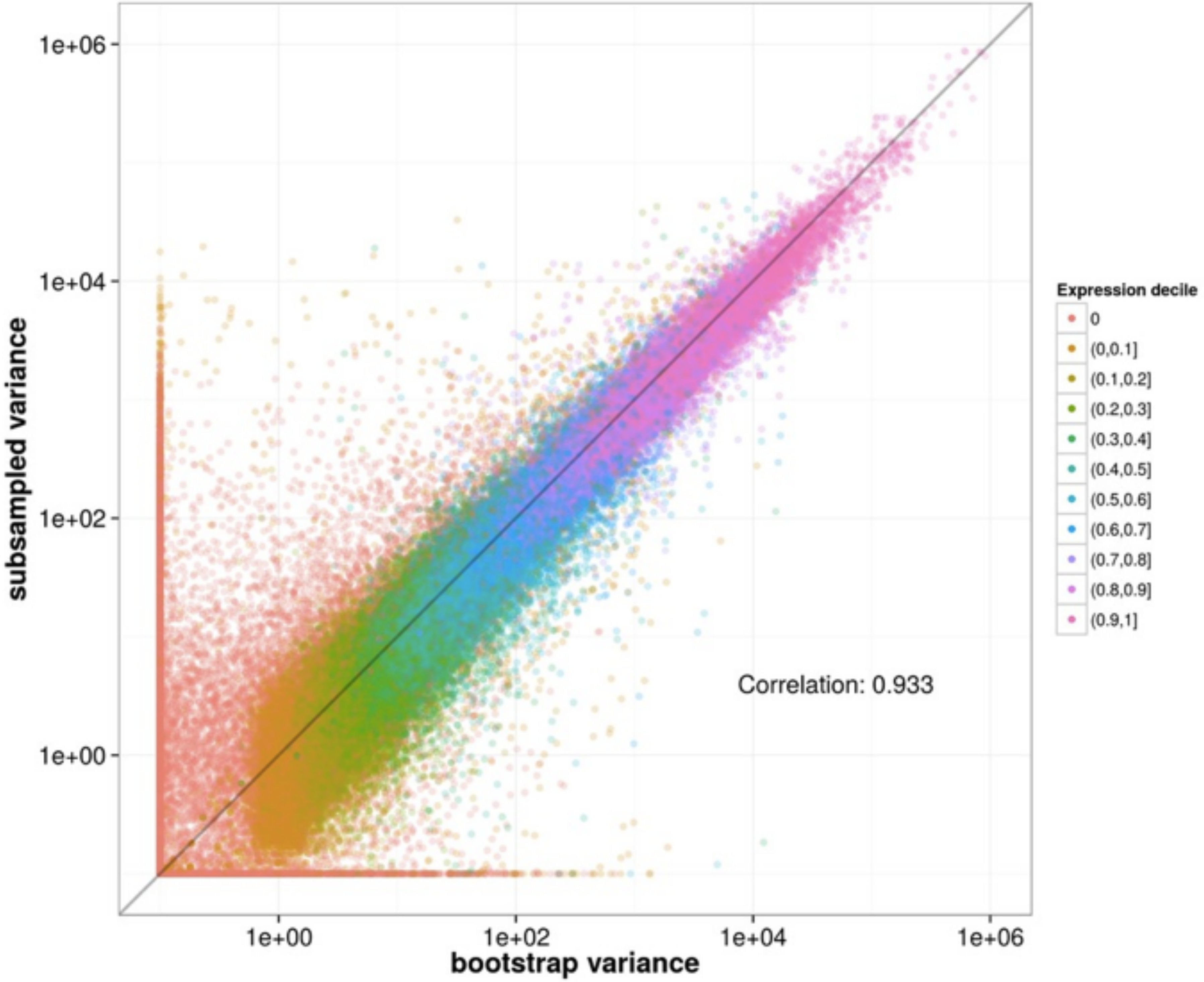
- The simplicity of the kallisto method allows us to apply a classic statistical tool known as the *bootstrap.*
- We can’t access alternate universes, but we can try to simulate them as best we can
- Alternate datasets are constructed by resampling from the original dataset
- Each alternate dataset can then be analyzed with kallisto allowing us to gain some insight into the variability inherent in the data

Estimating uncertainty



eq.classes	count
$\{t_1, t_3\}$	x 1
$\{t_1, t_2\}$	x 2
$\{t_1, t_2, t_3\}$	x 5
$\{t_2, t_3\}$	x 2
$\{t_3\}$	x 1
$\{t_2\}$	x 1
$\{t_1\}$	x 3

Testing the bootstrap



pachterlab.github.io/kallisto/