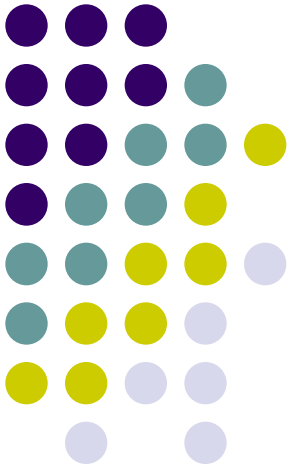
“Near-optimal RNA-Seq quantification with kallisto”



kallisto is:
a *transcript-oriented*
RNAseq quantification tool
that does *not require alignment*

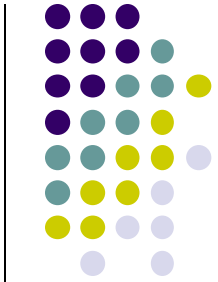
presented for Byte Club by

Anna Battenhouse

Iyer Lab

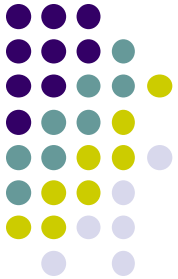
September 21, 2015

Resources



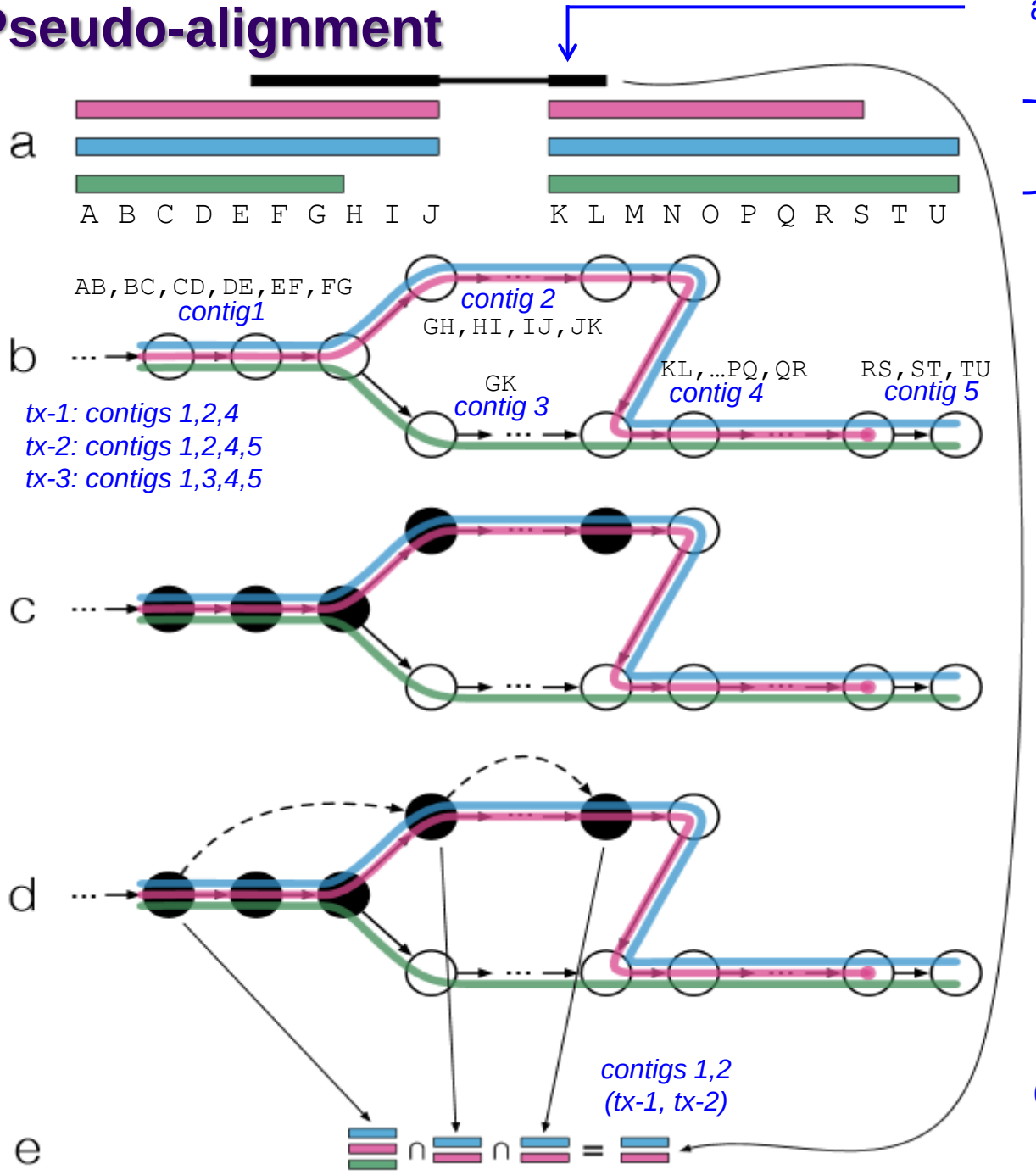
- Lior Pachter's blog on **kallisto**:
  https://liorpachter.wordpress.com/2015/05/10/near-optimal-rna-seq-quantification-with-kallisto/
- **kallisto** paper at arxiv:
  http://arxiv.org/ftp/arxiv/papers/1505/1505.02710.pdf
- Github project:
  - Getting started
    http://pachterlab.github.io/kallisto/starting.html
  - "Manual"
    http://pachterlab.github.io/kallisto/manual.html
  - ***Warning: tricky to build for non-Mac***
- Previous paper on **cuffdiff2**:
  http://www.nature.com/nbt/journal/v31/n1/full/nbt.2450.html

Kallisto overview



- No explicit alignment to reference genome or transciptome
- Instead, uses "*pseudoalignment*" to transcriptome
  - for each read, determine ***not where*** in each transcript it aligns, but rather which transcripts it is ***compatible*** with
  - simultaneously addresses 2 aspects of "multi-mapping" reads in traditional RNAseq pipelines
    - multiple possible genomic loci (addressed during alignment)
    - multiple possible transcripts of origin (addressed during quantification)
- Pseudoalignments are sufficient to quantify transcript abundances
  - Expectation Maximization (EM) algorithm is applied to a "simple" RNAseq Likelihood function
  - report estimated abundances as Transcripts per Million (TPM) + counts
- No *P-value* reported or differential expression (DE) support, but…
  - **kallisto** re-runs EM on multiple bootstrap re-samples to estimate variance
    - bootstraps are re-sampling ***with replacement*** from original sample, with same size
  - then **kallisto** bootstraps are used by add-on **sleuth** DE package

Pseudo-alignment



a read from a spliced message

3 transcripts, 6 exons
(provided as continuous fasta sequences)

Transcriptome de Bruijn Graph (T-DBG)

- nodes are k-mers
- each transcript is a path
- "transciptome path cover induces a k-compatibility class for each k-mer"

Read k-mers are hashed onto the T-DBG
(exact match)

Skip redundant T-DBG nodes
(for efficiency)

Find read's equivalence class
(intersect k-compatibility classes of read's k-mers)

Quantification

- RNAseq likelihood function ***L(α)***
  - parameters are the α_t , probability of selecting fragments from transcript ***t***
  - ***F*** – set of fragments/reads
  - ***T*** – set of transcripts
  - l_t – effective transcript length
  - $y_{f,t}$ – binary compatibility matrix (1 if fragment ***f*** compatible with transcript ***t***)
- RHS written as product over equivalence classes
  - ***E*** – set of equivalence classes
  - c^e – counts observed from equivalence class ***e***
    - sufficient statistics for the factorization
  - *very fast & efficient because # equivalence classes << # fragments/reads*
- Function iteratively optimized via Expectation Maximization to find the α_t
  - until all estimated counts > 0.01 change less than 1%

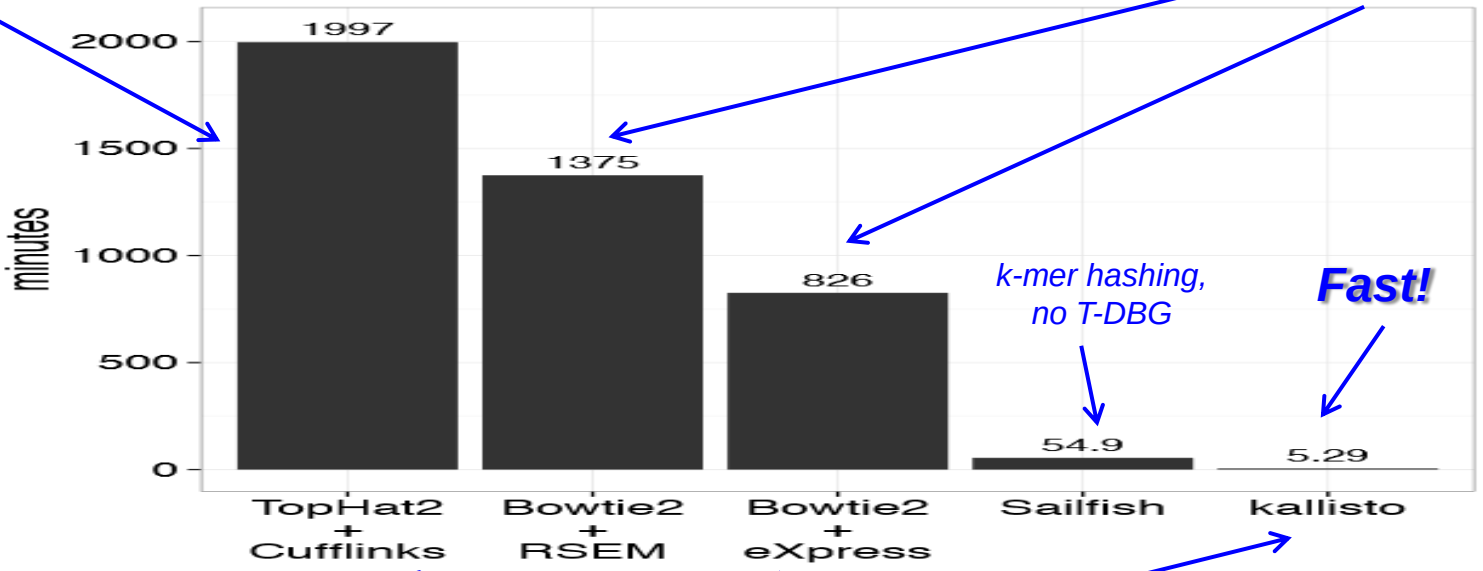
$$L(\alpha) \propto \prod_{f \in F} \sum_{t \in T} y_{f,t} \frac{\alpha_t}{l_t} = \prod_{e \in E} \left(\sum_{t \in e} \frac{\alpha_t}{l_t} \right)^{c_e}$$

Comparison with other transcript-oriented tools

20 simulated datasets, 30 M reads each, from 216M PE read superset

genome alignment + transcript quantification via EM

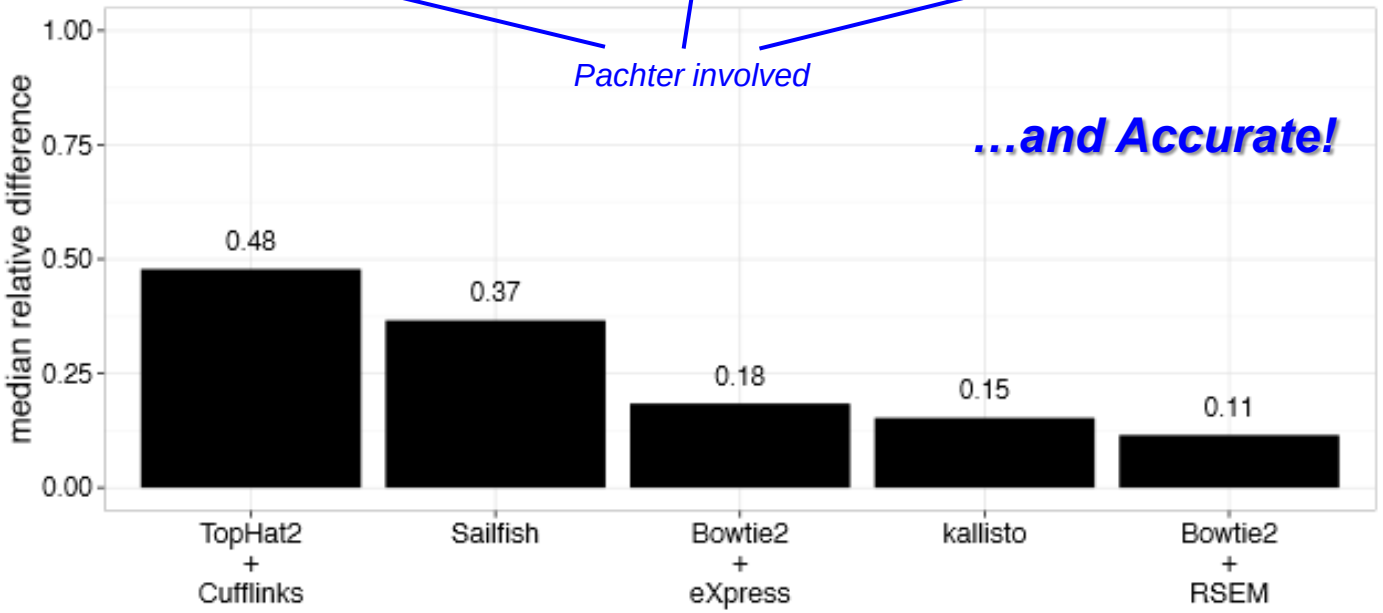
transcriptome alignment + quantification via EM



k-mer hashing, no T-DBG

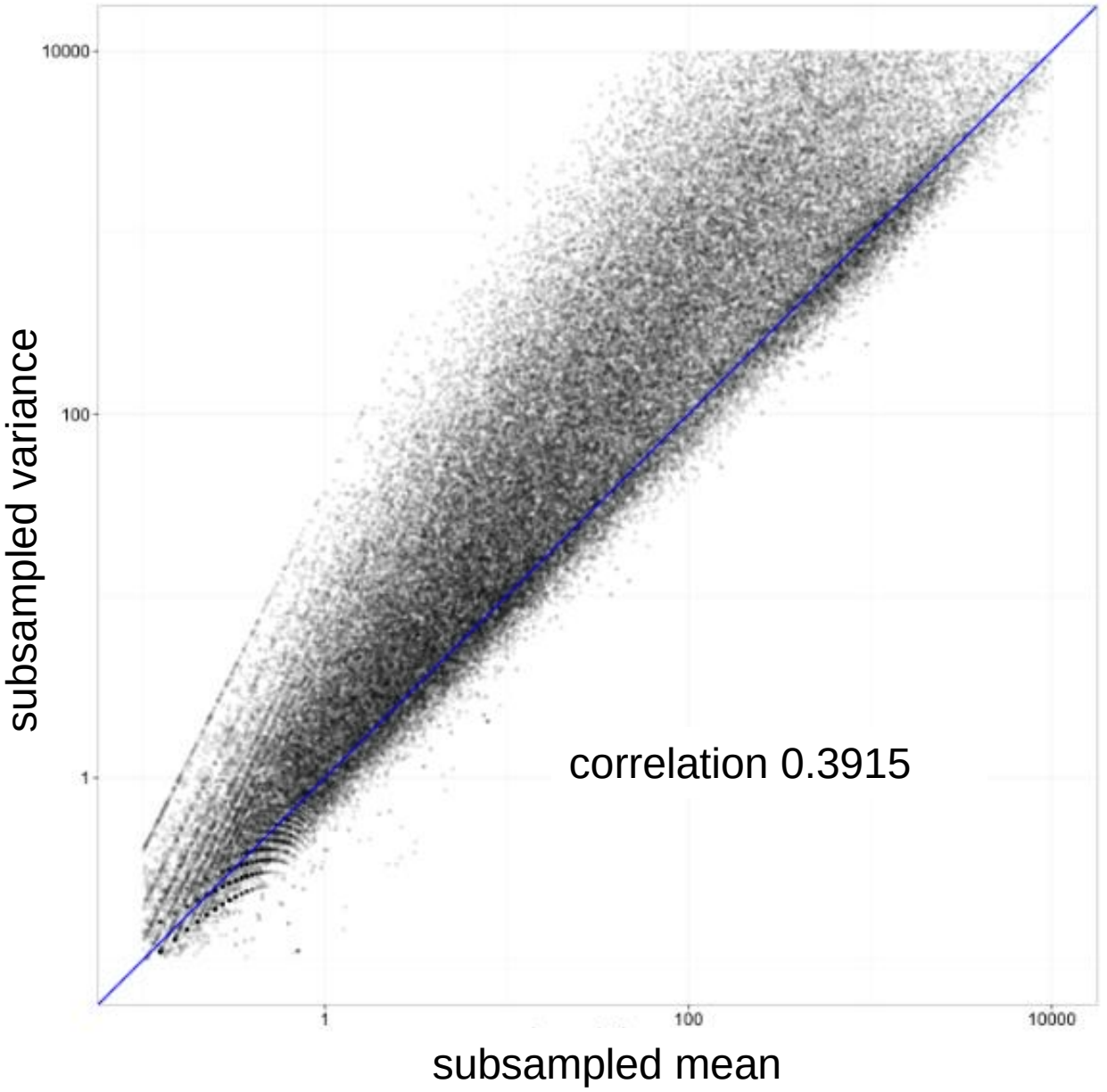
Fast!

Pachter involved



…and Accurate!

P-values derived from Poisson statistics may be misleading



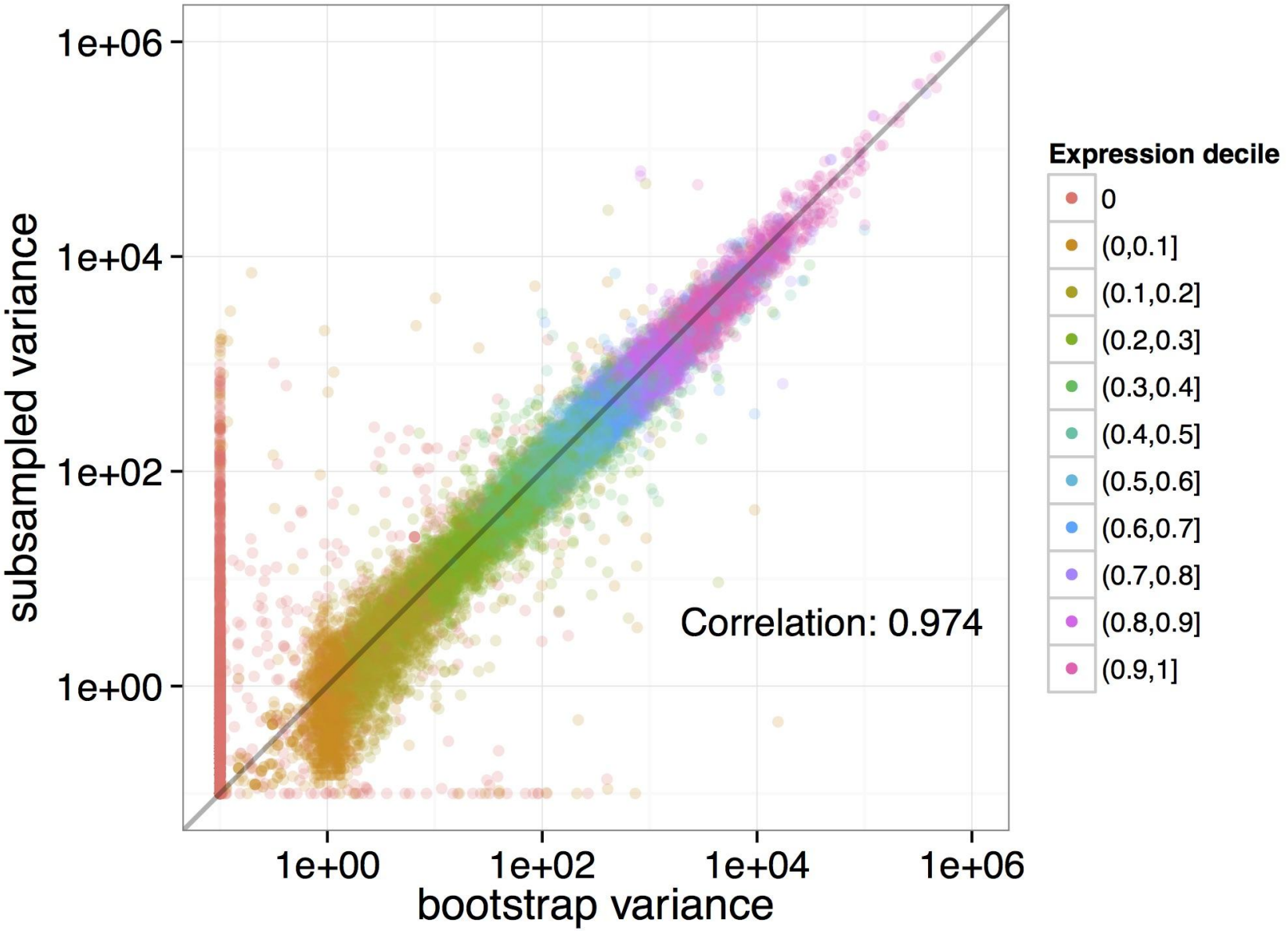
Count estimate variance does ***not*** follow Poisson (variance $\propto$ mean)

Variance is ***greater*** than expected from Poisson

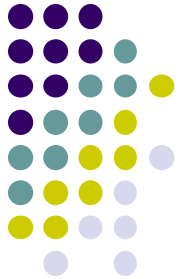
Here variance vs mean (log space) are plotted for each transcript across 40 subsamples (30M reads each), drawn from a superset of 216M PE reads.

Counts estimated using their EM on subsamples.

Variance calculated via EM on 40 bootstrap re-samples of 1 subsample correlates highly with variances based on 40 random subsamples (216M PE reads)



Running Kallisto



- Prepare the **kallisto** executable
  - Mac – download the pre-built binaries
  - Non-Mac: ***tricky!***
    - download the source from GitHub
    - install correct versions pre-requisite s/w & libraries
      - g++ version >= 4.8, **CMake** >= 2.8.12, **HDF5** C library version >= 18.12, **zlib**
    - build the binary
- Construct the T-DBG index (one-time)
  - obtain fasta sequences for your target transcriptome
    - e.g. from Gencode or RefSeq
  - use **kallisto index** to construct the index
    - much faster than building reference index for an aligner
- Run **kallisto quant** command on your fastq

Ran kallisto on real RNAseq data

Gave it raw fastq files (no adapter trimming) for one sample, 2 PE sequencing lanes

```
time(
kallisto quant -t 4 --bias -i ~/ref/kallisto/gencode_19.idx -o sd39 -b 100 \
    ./fq/SDMC_0039_GCCAAT_L004_R1_001.fastq.gz ./fq/SDMC_0039_GCCAAT_L004_R2_001.fastq.gz \
    ./fq/SDMC_0039_GCCAAT_L005_R1_001.fastq.gz ./fq/SDMC_0039_GCCAAT_L005_R2_001.fastq.gz \
)
# [quant] fragment length distribution will be estimated from the data
# [index] k-mer length: 31
# [index] number of targets: 95,309
# [index] number of k-mers: 75,956,643
# [index] number of equivalence classes: 346,220
# [quant] running in paired-end mode
# [quant] will process pair 1: ./fq/SDMC_0039_GCCAAT_L004_R1_001.fastq.gz
#                              ./fq/SDMC_0039_GCCAAT_L004_R2_001.fastq.gz
# [quant] will process pair 2: ./fq/SDMC_0039_GCCAAT_L005_R1_001.fastq.gz
#                              ./fq/SDMC_0039_GCCAAT_L005_R2_001.fastq.gz
# [quant] finding pseudoalignments for the reads ... done
# [quant] learning parameters for sequence specific bias
# [quant] processed 14,163,753 reads, 3,586,582 reads pseudoaligned
# [quant] estimated average fragment length: 79.4949
# [   em] quantifying the abundances ... done
# [   em] the Expectation-Maximization algorithm ran for 2,160 rounds
# [bstrp] number of EM bootstraps complete: 100

# real    14m0.542s
# user    34m6.400s
# sys     0m2.765s
```

← # transcripts in Gencode transcriptome fasta

~ 3.5 equiv. classes per transcript

~ 25%

~ 15 min clock time using 4 cores

~ 35 core-minutes total

Ran kallisto on a very large RNAseq dataset

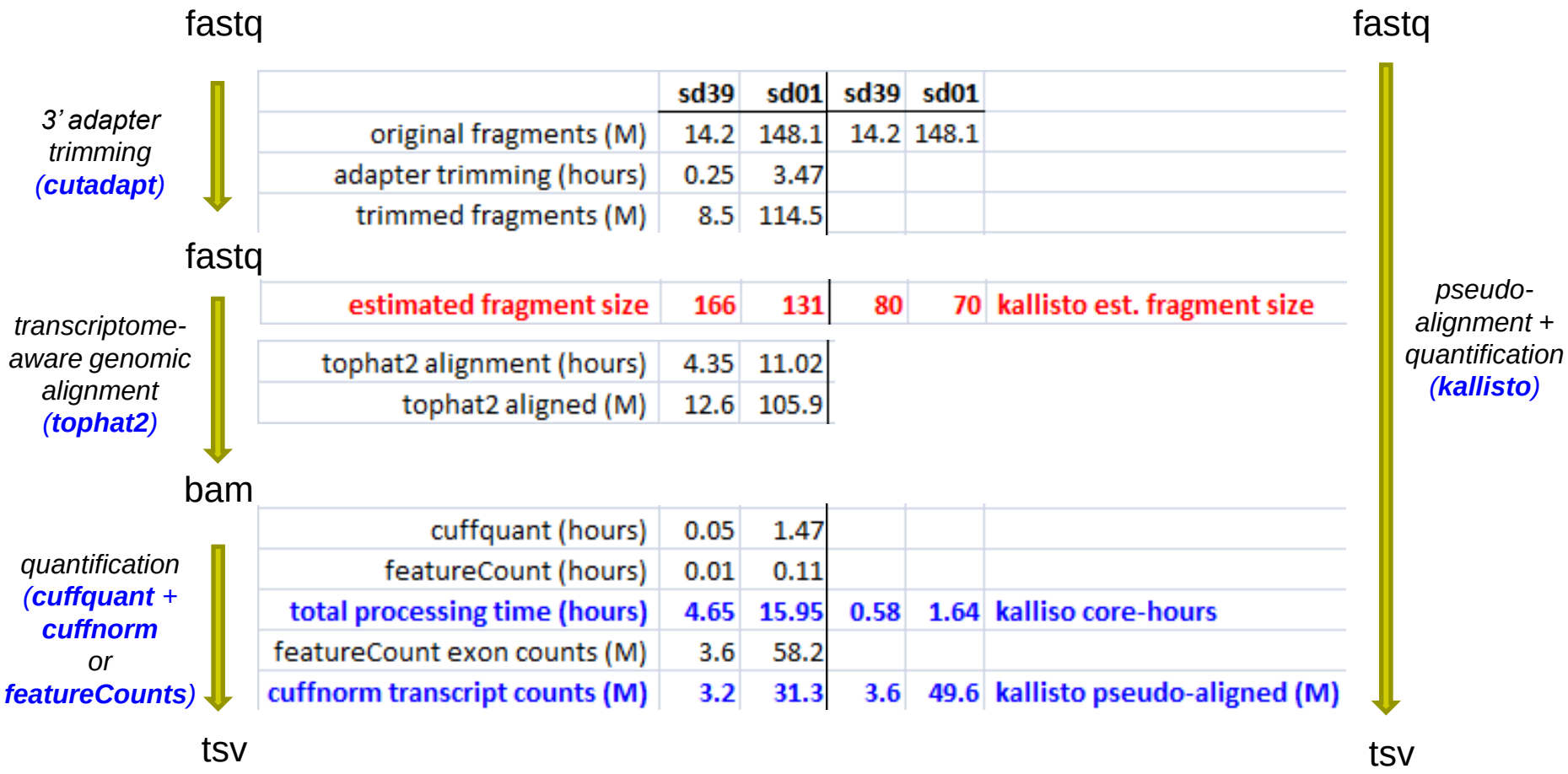
```
time(
kallisto quant -t 4 --bias -i ~/ref/kallisto/gencode_19.idx -o sd01 -b 100 \
    ./fq/SD_CLS_001_totalRNA_GGCTAC_L001_R1_001.fastq.gz \
    ./fq/SD_CLS_001_totalRNA_GGCTAC_L001_R2_001.fastq.gz \
    ./fq/SD_CLS_001_totalRNA_GGCTAC_L002_R1_001.fastq.gz \
    ./fq/SD_CLS_001_totalRNA_GGCTAC_L002_R2_001.fastq.gz )
# [quant] fragment length distribution will be estimated from the data
# [index] k-mer length: 31
# [index] number of targets: 95,309
# [index] number of k-mers: 75,956,643
# [index] number of equivalence classes: 346,220
# [quant] running in paired-end mode
# [quant] will process pair 1: ./fq/SD_CLS_001_totalRNA_GGCTAC_L001_R1_001.fastq.gz
#                              ./fq/SD_CLS_001_totalRNA_GGCTAC_L001_R2_001.fastq.gz
# [quant] will process pair 2: ./fq/SD_CLS_001_totalRNA_GGCTAC_L002_R1_001.fastq.gz
#                              ./fq/SD_CLS_001_totalRNA_GGCTAC_L002_R2_001.fastq.gz
# [quant] finding pseudoalignments for the reads ... done
# [quant] learning parameters for sequence specific bias
# [quant] processed 164,071,599 reads, 49,610,367 reads pseudoaligned
# [quant] estimated average fragment length: 69.8159
# [   em] quantifying the abundances ... done
# [   em] the Expectation-Maximization algorithm ran for 8,372 rounds
# [bstrp] number of EM bootstraps complete: 100

# real    57m12.449s
# user    98m37.788s
# sys     0m52.660s
```

~ 30%

1 hour clock time using 4 cores!
< 2 core-hours total

Traditional RNAseq pipeline

kallisto



	sd39	sd01	sd39	sd01	
original fragments (M)	14.2	148.1	14.2	148.1	
adapter trimming (hours)	0.25	3.47			
trimmed fragments (M)	8.5	114.5			
estimated fragment size	**166**	**131**	**80**	**70**	**kallisto est. fragment size**
tophat2 alignment (hours)	4.35	11.02			
tophat2 aligned (M)	12.6	105.9			
cuffquant (hours)	0.05	1.47			
featureCount (hours)	0.01	0.11			
total processing time (hours)	**4.65**	**15.95**	**0.58**	**1.64**	**kalliso core-hours**
featureCount exon counts (M)	3.6	58.2			
cuffnorm transcript counts (M)	**3.2**	**31.3**	**3.6**	**49.6**	**kallisto pseudo-aligned (M)**

Limitations

- Transcriptome orientation limitations
  - common to all transcriptome-oriented RNAseq tools (RSEM, sailfish, etc.)
    - global quantitation only as good as annotations
    - cannot measure RNA signal *outside* of annotated transcripts
    - cannot detect novel isoforms (**cufflinks**)
- Algorithmic limitations
  - quantifications are quite sensitive to estimated fragment length
    - especially for shorter transcripts
  - **kallisto** statistical methods do not detect most biases
    - or characterize their sources

Why focus on transcripts?

"Raw fragment counts inaccurately estimate changes in expression"

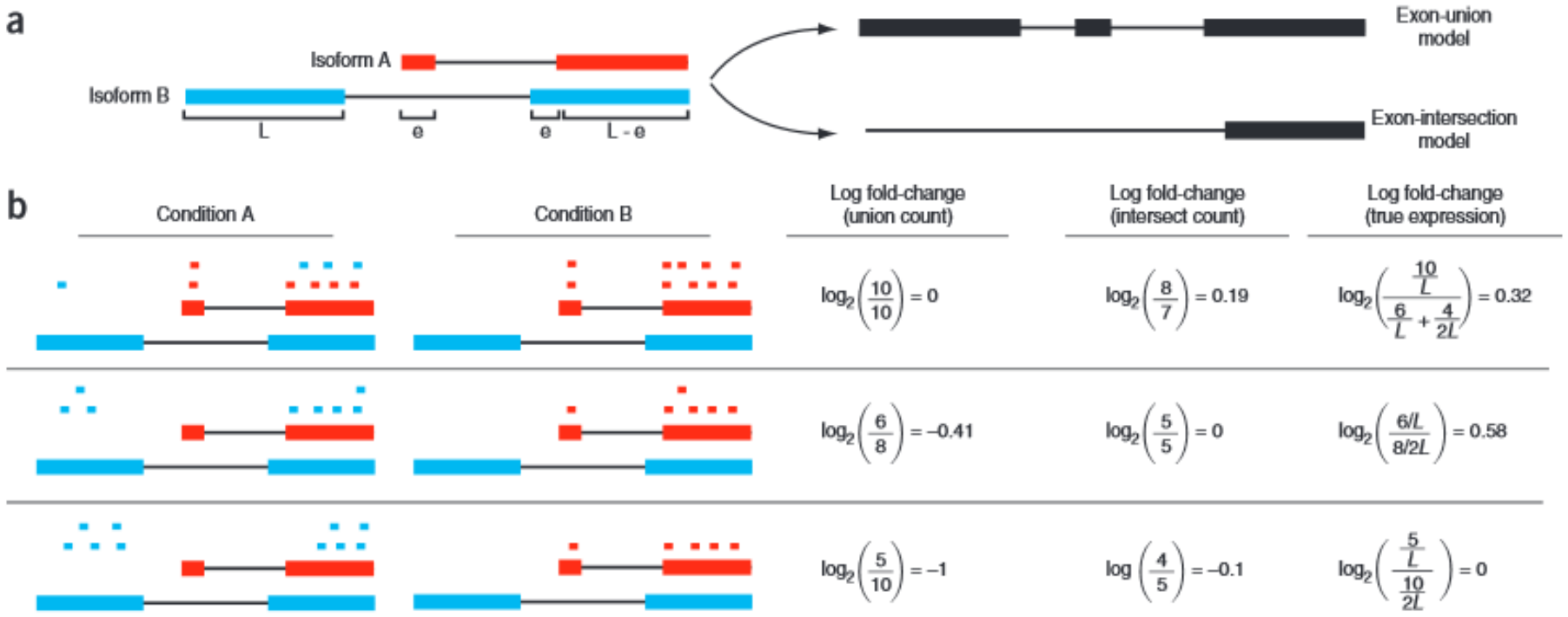


Figure 1 Changes in fragment count for a gene does not necessarily equal a change in expression. (**a**) Simple read-counting schemes sum the fragments incident on a gene's exons. The exon-union model counts reads falling on any of a gene's exons, whereas the exon-intersection model counts only reads on constitutive exons. (**b**) Both of the exon-union and exon-intersection counting schemes may incorrectly estimate a change in expression in genes with multiple isoforms. The true expression is estimated by the sum of the length-normalized isoform read counts. The discrepancy between a change in the union or intersection count and a change in gene expression is driven by a change in the abundance of the isoforms with respect to one another. In the top row, the gene generates the same number of reads in conditions A and B, but in condition B, all of the reads come from the shorter of the two isoforms, and thus the true expression for the gene is higher in condition B. The intersection count scheme underestimates the true change in gene expression, and the union scheme fails to detect the change entirely. In the middle row, the intersection count fails to detect a change driven by a shift in the dominant isoform for the gene. The union scheme detects a shift in the wrong direction. In the bottom row, the gene's expression is constant, but the isoforms undergo a complete switch between conditions A and B. Both simplified counting schemes register a change in count that does not reflect a change in gene expression.

Trapnell *et al*, *Nature Biotechnology*, 2013 (**cuffdiff2** paper)