User & Installation Manual

1 Installing TRUmiCount 2
1.1 Installation via Conda (Recommended) . . . 2
1.2 Manual Installation on linux or mac OS . . . 3
1.3 Running TRUmiCount . . . 4

2 Using TRUmiCount 5
2.1 Input formats & options . . . 5
2.2 Output formats & options . . . 5
2.3 Strand UMIs . . . 6
2.4 Single-cell Data . . . 7

3 Parameter Reference 8

4 Examples 10
4.1 Single-End Data . . . 10
4.2 Paired-End Data with strand UMIs . . . 10
4.3 Single-cell Data . . . 11

5 License 13

1 Installing TRUmiCount

1.1 Installation via Conda (Recommended)

Installing Conda

Conda[1] allows easy installation of a large range of software packages. See the Conda user guide for installation instructions. Briefly, (on 64-bit Linux) do[2]

```
INSTALLER=Miniconda2-latest-Linux-x86_64.sh
curl -O https://repo.continuum.io/miniconda/$INSTALLER
bash $INSTALLER -p ~/conda
```

Creating an environment

Conda allows creating multiple *environments*, each containing a different collection of packages. We create a separate environment for TRUmiCount

```
~/conda/bin/conda create -n trc
```

This environment is now *activated* to make it the target of further conda commands, and to make the installed software visible. **This must be done every time you open a new terminal window!**

```
source ~/conda/bin/activate trc
```

Installing BioConda

Conda packages are organized into so-called *channels*. We add the BioConda[3] channel which contains many common bioinformatics tools, amongst which is also TRUmiCount.

```
conda config --env --add channels defaults
conda config --env --add channels conda-forge
conda config --env --add channels bioconda
```

Installing TRUmiCount

With Conda installed and the Bioconda channels added, TRUmiCount is installed with

```
conda install trumicount
```

Installing SAMtools and UMI-Tools (Optional)

TRUmiCount relies on UMI-Tools[4] to extract the UMIs associated with a gene (or other genomic feature) from the mapped reads in a BAM file. We provide a modified version of UMI-Tools that improves the handling of paired-end reads and single-cell data in conjunction with TRUmiCount. You can install our modified version of UMI-Tools with

```
conda install -c cibiv umi_tools_tuc
```

You can also use the unmodified version of UMI-Tools together with TRUmiCount, but will have to manually post-process the output of `umi_tools group` before feeding it to TRUmiCount when working with single-cell data. To install the unmodified version, do

```
conda install umi_tools
```

While TRUmiCount does not directly require SAMtools, some of the examples in this manual rely on SAMtools being installed, so we suggest installing them with

```
conda install samtools
```

[1] See `http://conda.io` for details

[2] This will install Conda into the directory `conda` in your home directory. You can replace `~/conda` with a directory of your choice, but remember to do so everywhere

[3] See `http://bioconda.github.io/` for details

[4] Smith, T.S. *et al.* UMI-tools: Modelling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy. *Genome Research* **27**, 491-499 (2017). Software available at `http://github.com/CGATOxford/UMI-tools`

1.2 Manual Installation on linux or mac OS

Installing TRUmiCount

First, make sure you have R[5] (at least version 3.1) installed. First, we'll install a few required packages from CRAN. Amogst those is gwpcR[6] which contains the implementation of the model that TRUmiCount relies on. Start R (either on the command line, or open RStudio) and type

```
install.packages(c('statmod', 'akima', 'data.table', 'docopt', 'devtools'))
devtools::install_github("Cibiv/gwpcR", ref="latest-release")
```

Note that unless you have write privileges to your system's R installation, the packages will be installed only for your user.

With all prerequisites now available, TRUmiCount can now be installed. The commands below install TRUmiCount into your home directory. To install TRUmiCount system-wide (assuming you have the necessary privileges), don't specify `--prefix ~/.local`.

```
git clone https://github.com/Cibiv/trumicount.git trumicount
cd trumicount/
mkdir -p ~/.local/bin
./install.R --prefix ~/.local
```

To be able to run TRUmiCount, you must add `~/.local/bin` to your `PATH`. **This must be done every time you open a new terminal window!**

```
export PATH=~/.local/bin:$PATH
```

Installing UMI-Tools

TRUmiCount uses UMI-Tools[7] to extract UMIs from mapped reads and to correct for sequencing errors within the UMIs. We provide a modified version of UMI-Tools that improves the handling of paired-end reads and single-cell data in conjunction with TRUmiCount. You can install our modified version of UMI-Tools with

```
URL=https://github.com/Cibiv/UMI-tools.git
git clone -b tuc_patches $URL umi_tools
cd umi_tools
python setup.py install --user
```

To install the original version, instead do

```
URL=https://github.com/CGATOxford/UMI-tools
git clone $URL umi_tools
cd umi_tools
python setup.py install --user
```

Note that you will then have to manually post-process the output of `umi_tools group` before feeding it to TRUmiCount when working with single-cell data.

[5] R Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2017). `http://www.R-project.org/`

[6] Pflug, F.P. & von Haeseler, Arndt. TRUmiCount: Correctly counting absolute numbers of molecules using unique molecular identifiers. *Bioinformatics* (2017). DOI: `http://doi.org/10.1093/bioinformatics/bty283`

[7] Smith, T.S. *et al.* UMI-tools: Modelling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy. *Genome Research* **27**, 491-499 (2017). Software available at `http://github.com/CGATOxford/UMI-tools`

1.3 Running TRUmiCount

After installing TRUmiCount, you can display a list of options with

```
trumicount --help
```

For a detailed explanation of TRUmiCount's various modes of operation and supported file format, see *Using TRUmiCount* and *Parameter Reference*.

You can download an example input file[8] in BAM format and index it (necessary for reading it with TRUmiCount) with

```
curl -O https://cibiv.github.io/trumicount/kv_1000g.bam
samtools index kv_1000g.bam
```

From the indexed BAM file, you can let TRUmiCount compute a table containing the bias-corrected number of transcript molecules for each gene and write it to `kv_1000g.tab` with

```
trumicount --input-bam kv_1000g.bam --umi-sep ':' \
  --molecules 2 --threshold 2  --genewise-min-umis 3 \
  --output-counts kv_1000g.tab
```

For more examples of how to use TRUmiCount and a discussion of the various options used here, see *Examples*.

[8] containing a reduced version, restricted to the first 1000 genes and sub-sampled to 50%, of RNA-Seq data published in Kivioja, T. *et al.* Counting absolute numbers of molecules using unique molecular identifiers. *Nature Methods* **9**, 72-74 (2011)

2.1 Input formats & options

To be able to separate true UMIs from phantoms and to estimate and correct for the percentage of true UMIs that are lost during library preparation or data processing, TRUmiCount analyses the distribution of read counts per UMI for each gene (or any other type of genomic feature). TRUmiCount by default assumes that

- Each UMI initially had two copies. This is e.g. the case if molecules before amplification were double-stranded and the copies produced from both strands are identical. This number can be changed with "`--molecules`"
- UMIs must be supported by at least two reads to be assumed to be a true UMI and not a phantom. This threshold can be changed with "`--threshold`".

Reading UMIs from a BAM File

If a mapped BAM File is provided as input with "`--input-bam`", TRUmiCount uses UMI-Tools' "`group`" tool to extract a list of UMIs and their read counts from the specified BAM file. Sequencing errors within the UMIs are corrected by UMI-Tools by *merging* similar UMIs into one. When reading BAM file, on top of the defaults mentioned above, UMI-Tools assumes that

- The BAM file must have a corresponding index. A suitable index can be created with "`samtools index` *bamfile*".
- Sequences names are gene names. With the "`--gene-tag`" option, gene names can be read from a *tag* instead, whose name is specified with "`--umitools-option --gene-tag=`*tagname*".
- The UMI was appended to the read name, separated by '_'. A different separator can be specified with "`--umi-sep`"
- The BAM file contains single-end reads (read2 is ignored). To take the mapping position of both mates into account when grouping reads by UMI, specify "`--paired`".
- Reads with a mapping quality below 20 should be ignored. This threshold can be changed with "`--mapping-quality`".

Reading unfiltered UMIs from a tab-separated file

Instead of using UMI-Tools to extract UMI and their read counts from a BAM file, TRUmiCount is also able to read a previously computed table of UMIs with "`--input-umis`" (see *Output formats & options* for how to generate such a table). The table must be tab-separated with one row per UMI and contain at least the columns "sample", "gene", "reads". Similar UMIs are assumed to already have been merged, but filters are assumed *not* to have been applied yet. When dealing with strand UMIs (see section *Strand UMIs*), TRUmiCount also requires the columns "pos" and "end", which contain the mapping position of read1 respectively read2.

The UMI-Tools and BAM-related options "`--umi-sep`", "`--umitools`", "`--umitools-option`", "`--paired`" and "`--mapping-quality`" are ignored if "`--input-umis`" is used.

2.2 Output formats & options

Writing corrected per-gene molecule counts to a tab-separated file

The main output of TRUmiCount is a table, written to a tab-separated file specified with "`---output-counts`" containing for each gene the columns

sample : The sample identifier (e.g cell barcode)

gene : The gene identifier (see discussion in *Reading UMIs from a BAM File*)

n.umis : The number of true UMIs (i.e. after applying filters) detected

n.tot : The estimated total number of molecules, i.e. "n.umis" / (1 - "loss").

efficiency : The estimated gene-specific amplification efficiency

depth : The gene-specific sequencing depth in reads/molecule.

loss : The estimated gene-specific loss, i.e. the probability of not detecting or filtering a true UMI.

n.obs : The number of observations used to estimate "efficiency", "depth" and "loss". If "`--combine-strand-umis`" is used, this number can be larger than "n.umis", because in this case "n.umis" counts UMI *pairs* while "n.obs" counts *strand* UMIs.

Writing unfiltered UMIs to a tab-separated file

The option "`--output-umis`" (together with "`--input-bam`") writes the UMIs reported by UMI-Tools (after merging similar UMIs) and their read counts to a tab-separated file, suitable for being later read with "`--input-umis`". This can be used to avoid the overhead of running UMI-Tools multiple times if the same input BAM file is processed multiple times with TRUmiCount, e.g. to test different read count thresholds or initial molecule counts.

Writing final, filtered & strand-combined UMIs to a tab-separated file

The final list of UMIs used when fitting the model and computing corrected molecule counts can be output with "`--output-final-umis`". The format of the generated file depends on whether one of "`--filter-strand-umis`" or "`--combine-strand-umis`" is used. In the former case, the table contains one line per *strand UMI*, listing the UMI's read count in the column "reads" and it's partner UMI's read count in "reads.other". In the latter case, the table contains one line per *UMI pair*, listing the read count of the plus-strand partner UMI (arbitrarily chosed to be the one where read1's mapping position is smaller than read2's) in "reads.plus" and the read count of the minus-strand partner UMI in "reads.minus".

Diagnostic Plots

When the "`--output-plots`" option is used, TRUmiCount generates diagnostic plots as part of the analysis and writes them to the specified PDF file. The plots produced are

Distribution of reads per UMI. This plot shows the observed library-wide distribution of per-UMI read counts and the distribution predicted by TRUmiCount's model of amplification and sequencing.

Variance of the gene-specific loss estimate. Shows the observed variance of the gene-specific loss estimates as a function of the number of UMIs per gene (n_{obs}) and compares it to the interpolation curve ($s + u/n_{\text{obs}}$).

2.3 Strand UMIs

Some UMI-based library preparation protocols produce *strand UMIs* where the two strands of an initial double-stranded template molecule produce distinct (but related) UMIs. Filtering out UMIs for which the partner UMI corresponding to the second strand is not detected offers a second possibility (besides the read count threshold) for filtering our phantom UMIs.

TRUmiCount supports stranded UMIs as produced by the protocol of Shiroguchi *et al.*[9]. With this protocol, both read1 and read2 carry a separate molecular barcode. UMI pairs belonging to the same double-stranded template molecule are found by looking for pairs of UMIs whose read1 and read2 barcodes and mapping positions are swapped (mapping position here refers to the genomic coordinate of the first mapped base in *read direction*, i.e. for reverse-mapped reads this differes from the mapping position stated in the BAM file).

When working with strand UMIs, the initial molecule count should usually be set to 1 (the default is 2!), i.e. "`--molecules 1`" should be used.

[9] Shiroguchi, K., Jia, T. Z., Sims, P. A. & Xie, X. S. Digital RNA sequencing minimizes sequence-dependent bias and amplification noise with optimized single-molecule barcodes. *Proceedings of the National Academy of Sciences of the United States of America* **109**, 1347-1352 (2012).

Filtering out incomplete strand UMI pairs

With the option "`--filter-strand-umis`", UMIs are filtered out if their partner UMI cannot be detected (these UMIs are also not counted as "phantom UMIs" in the diagnostic plots!). This may happen occasionally even for true UMIs, if their partner UMI happens not to have been sequenced. The actual loss in this mode is not simply the probability $\mathbb{P}(C < T)$ of an UMI having fewer than T reads, but is instead $1 - (1 - \mathbb{P}(C < T)) \cdot (1 - \mathbb{P}(C = 0))$. TRUmiCount adjust the loss computation accordingly, and states the corrected loss also in the diagnostic plots. The plotted model distribution, however, does not take this adjustment into account, so that the stated loss is no longer simply the sum of the model probabilities for read counts less than the threshold.

Combining strand UMIs into pairs

With the option "`--combine-strand-umis`", pairs of partner UMIs stemming from the two strands of a single initial template molecules are combined, and the read count threshold is applied to both partners simultaneously (UMIs without a partner are again *not* shown as "phantom UMIs" in the diagnostic plots). The actual loss in this mode is $1 - (1 - \mathbb{P}(C < T))^2$, because pairs are dropped if *either* of the partner UMIs has a read count below the threshold. TRUmiCount adjust the loss computation accordingly, and states the corrected loss also in the diagnostic plots. The plotted model distribution, however, does not take this adjustment into account, so that the stated loss is no longer simply the sum of the model probabilities for read counts less than T.

2.4 Single-cell Data

TRUmiCount can handle single-cell data where the same library contains data from multiple individual cells, distinguishable by an additional *cell barcode* that is parts of each read in addition to the UMI. To compute corrected UMI counts per cell (i.e. for the total number of UMIs within each individual cell), specify "`--group-per cell`". To compute corrected UMI counts per gene within each cell, specify "`--group-per cell,gene`". If the grouping key contains 'cell', and "`--input-bam`" is used to read UMI directly from a BAM file with the help of UMI-Tools, the option "`--per-cell`" is passed to UMI-Tools.

Note that even when the "`--per-cell`" option is specified, the "`group`" command of the stock version of UMI-Tools (at least up to version 0.5.3) does not output the cell information in a format usable by TRUmiCount. We provide a patched version of UMI-Tools that amends the output by an additional column containing the cell barcode extracted from each read to make it possible for TRUmiCount to use that information. If you followed the installation instructions, you probably have our patched version installed already, if not see *Installing TRUmiCount* for how to obtain it.

Currently, the grouping key specified with "`--group-per`" is used to defined *both* the level on which the percentage of lost UMIs is estimated, *and* the level on which UMI counts are output. We plan to allow two separate grouping keys to be specified, so that the loss estimation can be done on a higher level (where more data to compute reliable estimates is available), will still outputting counts for each gene within each individual cell. Currently, if you desire to e.g. estimate the loss on a per-cell level, and use those estimated to computed corrected per-gene counts, you have to run TRUmiCount twice, and then use your own script to use the per-cell correction loss estimates to corrected the counts for all genes within the cell.

`--input-bam` *file*: read UMIs from *file* (uses «`umi_tools group`»)

`--input-umitools-group-out` *file*: read UMIs from *file* produced by «`umi_tools group`»

`--input-umis` *file*: read UMIs from *file* (previously produced by `--output-umis`)

`--output-counts` *file*: write bias-corrected per-group counts and models to *file*

`--output-umis` *file*: write UMIs reported by «`umi_tools group`» to *file*

`--output-final-umis` *file*: write strand-combined and filtered UMIs to *file*

`--output-readdist` *file*: write global reads/UMI distribution (before and after filtering) to *file*

`--output-plots` *plot*: write diagnostic plots in PDF format to *plot*

`--output-groupwise-fits` *file*: write group-wise model details to *file*

`--output-genewise-fits` *file*: obsolete name for `--output-groupwise-fits`

`--umitools` *umitools*: use executable *umitools* to run «`umi_tools group`» (Default: 'umi_tools')

`--umitools-option` *umitoolsopt*: pass *umitoolsopt* to «`umi_tools group`» (see «`umi_tools group --help`»)

`--umi-sep` *umisep*: assume *umisep* separates read name and UMI (passed to umi_tools) (Default: '_')

`--umipair-sep` *umipairsep*: assume *umipairsep* separates read1 and read2 UMI (see Strand UMIs) (Default: '')

`--paired` : assume BAM file contains paired reads (passed to umi_tools) (Default: No)

`--mapping-quality` *mapq*: ignored read with mapping quality below *mapq* (passed to umi_tools) (Default: 20)

`--filter-strand-umis` : filtes UMIs where only one strands was observed (Default: No)

`--combine-strand-umis` : combine UMIs strand pairs (implies `--filter-strand-umis`) (Default: No)

`--threshold` *threshold*: remove UMIs with fewer than *threshold* reads

`--threshold-quantile` Q: use quantile Q of the raw read-count distribution for *threshold* (Default: '0.2')

`--molecules` *molecules*: assume UMIs are initially represented by *molecules* copies (strands) (Default: 2)

`--group-per` *key1*,*key2*,...: counts UMIs per distinct key(s), can be "cell" and/or "gene", "cell" implies `--umitools-option --per-cell` (Default: 'gene')

`--skip-groupwise-fits` : skip group-wise model fitting, stop after plotting the global fit

`--include-filter-statistics` : add filtered and unfiltered read and UMI counts to count table (Default: No)

`--groupwise-min-umis` *minumis*: use global estimates for groups with fewer than *minumis* (strand) UMIs (Default: 5)

`--genewise-min-umis` *minumis*: obsolete name for `--groupwise-min-umis`

`--cores` *cores*: spread group-wise model fitting over *cores* cpus (Default: 1)

`--variance-estimator` *varest*: use *varest* to estimate variances, can be "lsq" or "mle" (Default: 'lsq')

`--digits` *digits*: number of digits to output (Default: 3)

`--plot-hist-bin` *plotxbin*: make read count histogram bins *plotxbin* wide

`--plot-hist-xmax` *plotxmax*: limit read count histogram plot to at most *plotxmax* reads per UMI

`--plot-skip-phantoms` **:** do not show phantom UMIs in histogram plot (Default: No)

`--plot-var-bins` *plotvarbins* **:** plot *plotvarbins* separate emprirical variances (Default: 10)

`--plot-var-logy` **:** use log scale for the variance (y) axis (Default: No)

`--verbose` **:** enable verbose output

The example data can be downloaded from `https://cibiv.github.io/trumicount/`. Before TRUmiCount can process these BAM files, they need to be indexed by running

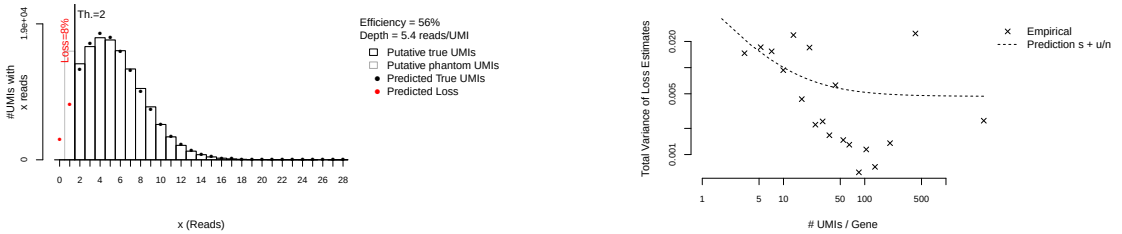
```
samtools index sg_100g.bam
samtools index kv_1000g.bam
samtools index 10xn9k_10c.bam
```

4.1 Single-End Data

The file `kv_1000g.bam` contains a reduced (restricted to the first 1000 genes, and subsamples to 50%) version of data published by Kivioja *et al.*[10].

```
trumicount --input-bam kv_1000g.bam --umi-sep ':' \
  --molecules 2 --threshold 2  --genewise-min-umis 3 \
  --output-plots kv_1000g.pdf --plot-hist-bin 1 \
  --plot-var-bins 20 --plot-var-logy \
  --output-counts kv_1000g.tab \
  --cores 4
```

This command uses UMI-Tools to read (error-corrected) UMIs and their read counts from `kv_1000g.bam` (`--input-bam`), removes UMIs with fewer than 2 reads (`--threshold`), compute gene-specific loss estimates for genes with at least 3 surviving UMIs (`--genewise-min-umis`) assuming two initial copies of each UMI (`--molecules`), writes the bias-corrected counts to `kv_1000g.tab` (`--output-counts`), and outputs diagnostic plots to `kv_1000g.pdf` (`--output-plots`). The gene-specific parameter estimation is spread over 4 cores (`--cores`). For the plots, the bin size for the read-count distribution plot is set to 1 (`--plot-hist-bin`), the number of bins for which the empirical variance is plotted is increased to 15 (`--plot-var-bins`), and the y-axis of the variance plot is log-scaled (`--plot-var-logy`).



`kv_1000g.pdf`. `kv_1000g.bam` processed in single-end mode.

4.2 Paired-End Data with strand UMIs

The file `sg_100g.bam` contains a reduced (restricted to the first 100 genes, and subsamples to 25%) version of data published by Shiroguchi *et al.*[11] which uses *strand UMIs* (see *Strand UMIs* for details).

Filtering out incomplete strand UMI pairs

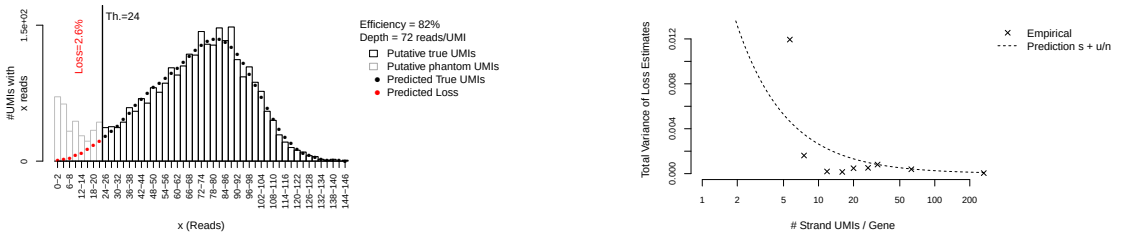
```
trumicount --input-bam sg_100g.bam --umi-sep ':' --umipair-sep '-'\
  --paired --filter-strand-umis --molecules 1 --threshold 24 \
  --output-plots sg_100g.pdf --plot-hist-bin 3 \
  --output-counts sg_100g.tab
```

This command uses UMI-Tools to read (error-corrected) UMIs and their read counts from `sg_100g.bam` (`--input-bam`) containing paired-end reads (`--paired`), removes UMIs whose partner corresponding to the initial molecule's other strand was not detected (`--filter-strand-umis`, `--umipair-sep`) or who have fewer than 24 reads (`--threshold`), compute gene-specific loss estimates assuming a single initial copy of each UMI (`--molecules`), writes the bias-corrected counts

[10] Kivioja, T. *et al.* Counting absolute numbers of molecules using unique molecular identifiers. *Nature Methods* **9**, 72-74 (2011)

[11] Shiroguchi, K., Jia, T. Z., Sims, P. A. & Xie, X. S. Digital RNA sequencing minimizes sequence-dependent bias and amplification noise with optimized single-molecule barcodes. *Proceedings of the National Academy of Sciences of the United States of America* **109**, 1347-1352 (2012).

to `sg_100g.tab` (`--output-counts`), and outputs diagnostic plots to `sg_100g.pdf` (`--output-plots`). For the plots, the bin size for the read-count distribution plot is set to 3 (`--plot-hist-bin`).

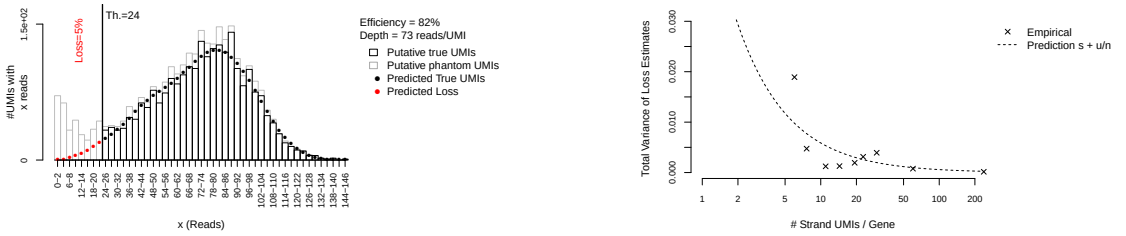


`sg_100g.pdf`. `sg_100g.bam` processed in *filter strand UMIs* mode.

Combining strand UMIs into pairs

```
trumicount --input-bam sg_100g.bam --umi-sep ':' --umipair-sep '-' \
  --paired --combine-strand-umis --molecules 1 --threshold 24 \
  --output-plots sg_100g_comb.pdf --plot-hist-bin 3 \
  --output-counts sg_100g_comb.tab
```

This command uses UMI-Tools to read (error-corrected) UMIs and their read counts from `sg_100g.bam` (`--input-bam`) containing paired-end reads (`--paired`), combines strand UMIs into UMI pairs (`--combine-strand-umis`, `--umipair-sep`), filters UMI pairs with fewer than 24 reads of either strand UMI(`--threshold`), computes gene-specific loss estimates assuming a single initial copy of each strand UMI (`--molecules`), writes the bias-corrected counts to `sg_100g.tab` (`--output-counts`), and outputs diagnostic plots to `sg_100g.pdf` (`--output-plots`). For the plots, the bin size for the read-count distribution plot is set to 3 (`--plot-hist-bin`).



`sg_100g_comb.pdf`. `sg_100g.bam` processed in *combine strand UMIs* mode. Plots show individual strand UMIs, not UMI pairs. Phantoms now appear even beyond the threshold, because strand UMIs are also considered phantoms if their *partner UMI* has a read count below the threshold.

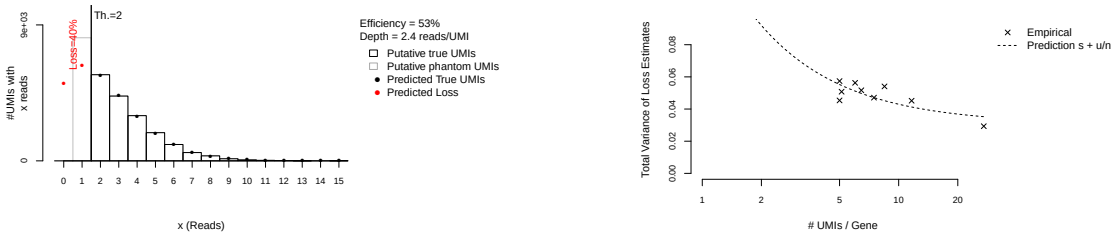
4.3 Single-cell Data

The file `10xn9k_10c.bam` contains a reduced (restricted to 10 individual cells) version of the *9k Brain Cells from an E18 Mouse*[12] public dataset provided by 10x Genomics.

```
trumicount --input-bam 10xn9k_10c.bam \
  --molecules 1 --threshold 2 --group-per cell,gene \
  --umitools-option --per-gene --umitools-option --gene-tag=XF \
  --output-plots 10xn9k_10c.pdf --output-counts 10xn9k_10c.tab
```

This command uses UMI-Tools to read (error-corrected) UMIs and their read counts from `10xn9k_10c.bam` (`--input-bam`), removes UMIs with fewer than 2 reads (`--threshold`), compute gene-specific loss estimates for each combination of gene and cell (`--group-per`) assuming a single initial copy of each UMI (`--molecules`), writes the bias-corrected counts to `10xn9k_10c.tab` (`--output-counts`), and outputs diagnostic plots to `10xn9k_10c.pdf` (`--output-plots`).

UMIs are group disregarding the mapping position, i.e. identical UMI sequences found within the same gene are assumed to belong to the same original molecule (`--umitools-option --per-gene`). The gene a particular read belongs to is read from the read's XF tag in the BAM file (`--umitools-option --gene-tag=XF`).



10xn9k_10c.pdf. 10xn9k_10c.bam processed with "--group-per cell,gene".

5 License