Using TRUmiCount

Florian G. Pflug
<florian.pflug@univie.ac.at>

Contents

1 Installing TRUmiCount

1.1 Installation via Conda (Recommended)

Installing Conda

Conda is a package manager that allows easy installation of a large range of software packages. See `https://conda.io/docs/user-guide/install/index.html` for your options of how to insteall conda. Briefly, on 64-bit linux do[1]

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

Creating an environment

Conda allows the creation of multiple *environments*, each containing different collections of packages. We will now create an environment for TRUmiCount

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

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

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

Installing BioConda

Conda packages are organized into so-called *channels*. We add the BioConda channel which provides many common tools for dealing with high-throughput sequencing data

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

Installing TRUmiCount

Finall we add the channel that supplies TRUmiCount and a modified version of umi_tools[2] with improved handling of paired-end reads[3]

```
conda config --env --add channels \
  http://tuc:tuc@www.cibiv.at/~pflug_/trumicount/
```

TRUmiCount and our version of umi_tools can now be installed

```
conda install trumicount umi_tools samtools
```

[1] You can replace `/conda` with a directory of your choice, but remember to do so in *all* the commands

[2] Smith, T.S. *et al.* UMI-tools: Modelling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy. *Genome Research* **27**, 491-499 (2017)

[3] Note that the backslash ("\") only serves to make your shell ignore the linebreak that follows it. If you enter the command as a single line, skip the backslash

1.2 Manual Installation on linux or mac OS

Installing TRUmiCount

First, make sure you have R[4] (at least version 3.1) installed. First, we'll install a few required packages from CRAN. Start R (either on the command line, or open RStuido) and type

```
install.packages(c('statmod', 'akima', 'data.table', 'docopt'))
```

Note that unless you have write privileges to your system's R installation, the packages will be installed only for your user. Having installed these packages, you can install our `gwpcR` package, which isn't available on CRAN yet. Open a terminal window (not R) and do

```
CRED=gwpcr-access:gwpcr123
URL=https://$CRED@bitbucket.org/florian_pflug/gwpcr.git
git clone $URL gwpcr
cd gwpcr/
R CMD INSTALL .
```

With all prerequisites 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`.

```
CRED=gwpcr-access:gwpcr123
URL=https://$CRED@bitbucket.org/florian_pflug/trumicount.git
git clone $URL 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 a new terminal window is opened!**

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

Installing umi_tools

TRUmiCount uses umi_tools to extract UMIs from the reads in a BAM file and to correct for sequencing errors. We provide a modified version of umi_tools which improves the handling of paired-end reads. You can install our version with

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

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

2 Using TRUmiCount

2.1 Supported input formats & options

To be able to separate true UMIs from biases 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 of 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` *bamfile*", TRUmiCount uses umi_tools's `group` tool to extract a list of UMIs and their read counts from a BAM file. Sequencing errors in the UMIs are corrected by umi_tools by *merging* similar UMIs into one. When reading BAM file, on top of the defaults mentioned above, TRUmiCount assumes that

- The BAM file must have a corresponding index. A suitable index can be created with "`samtools index bamfile`".
- The sequence name corresponds to the gene name. Alternatively, the gene names can be stored in BAM file tags – this can be changed by using umi_tool's `--gene-tag` option. To tell TRUmiCount to invoke umi_tools using that option, use "`--umitools-option--gene-tag=`*GENE_TAG*" when invoking TRUmiCount.
- The UMI was appended to the read name, and 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 UMIs from a tab-separated file

Instead of using umi_tools to extract UMI and their read counts from a BAM file, TRUmiCount can read a previously computed table of UMIs with "`--input-umis`".

The table must be tab-separated with one row per UMI and contain at least the columns "sample", "gene", "reads". When dealing with strand UMIs (see section *Strand UMIs*), TRUmiCount will also use columns "pos" and "end" containing 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.

The option "`--output-umis`" (together with "`--input-bam`") produces a suitable input file for "`--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.

2.2 Output

The main output of TRUmiCount is a table containing for each gene to 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$/(1 - \text{loss})$.

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**.

Diagnostic Plots

When the "`--output-plots`" option is used, TRUmiCount generates diagnostic plots as part of the analysis. 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 interpolated 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 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.*[5]. 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.

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 (again, UMIs without a partner *are not* not counted as "phantom UMIs", but we *do* count UMIs as phantoms whose partner's read count lies below the threshold!). The actual loss in this mode is $1 - (1 - \mathbb{P}(C < T))^2$. 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.

[5]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).

3 Parameter Reference

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

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

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

`--output-counts` *countstab*: write bias-corrected per-gene counts and models to *countstab*

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

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

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

`--output-genewise-fits` *genefitstab*: write gene-wise model details to *genefitstab*

`--umitools` *umitools*: path to umitools (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: '*true*')

`--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: '*false*')

`--combine-strand-umis` : combine UMIs strand pairs (implies --filter-strand-umis) (Default: '*false*')

`--threshold` *threshold*: remove UMIs with fewer than *threshold* reads (Default: 2)

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

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

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

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

`--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: *'false'*)

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

`--plot-var-logy`: use log scale for the variance (y) axis (Default: *'false'*)

`--verbose`: enable verbose output

4 Examples

The example data can be downloaded from `http://tuc:tuc@www.cibiv.at/~pflug_/trumicount`. Before TRUmiCount can process these BAM files, they need to be indexed by running

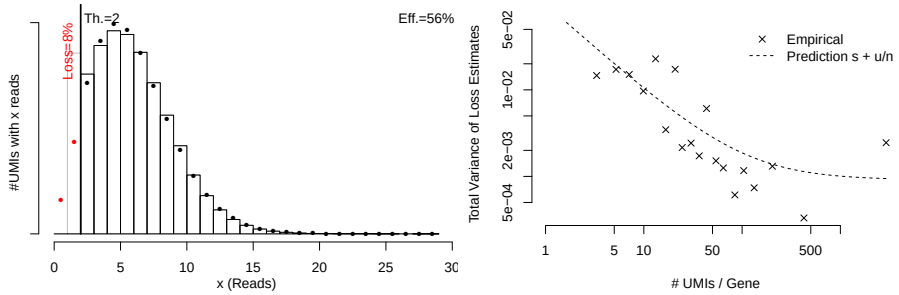
```
samtools index sg_100g.bam
samtools index kv_1000g.bam
```

4.1 Single-End Data

The file `kv_1000g.bam` contains a reduced (restricted to the first 0100 genes, and subsamples to 50%) version of data published by Kivioja *et al.*[6].

```
trumicount --input-bam kv_1000g.bam \
  --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`, and outputs diagnostic plots to `kv_1000g.pdf`. 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.

[6]Kivioja, T. *et al.* Counting absolute numbers of molecules using unique molecular identifiers. *Nature Methods* **9**, 72âĂŞ74 (2011)

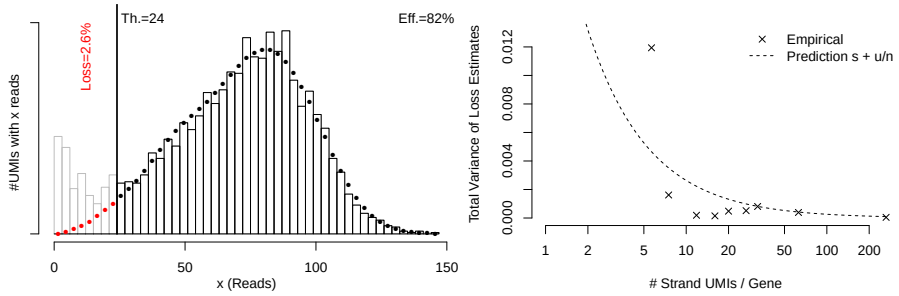
4.2 Paired-End Data with strand UMIs

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.*[7] which uses *strand UMIs* (see section *Strand UMIs* for details).

Filtering out incomplete strand UMI pairs

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



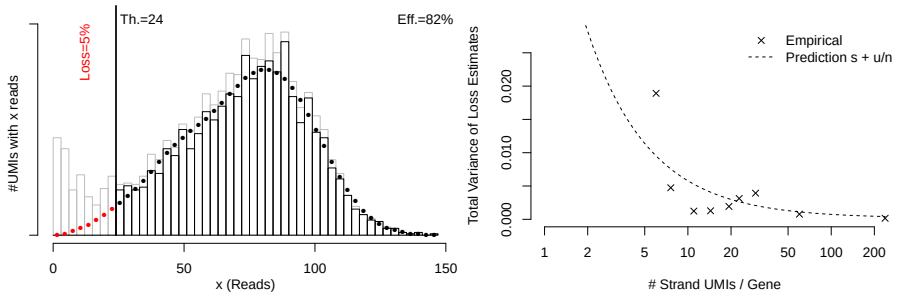
`sg_100g.pdf`. `sg_100g.bam` processed in *filter strand UMIs* mode.

[7] 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).

Combining strand UMIs into pairs

```
trumicount --input-bam sg_100g.bam --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`, and outputs diagnostic plots to `sg_100g.pdf`. 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.