STARsolo: mapping, demultiplexing and quantification for single cell RNA-seq

Major update in STAR 2.7.9a (2021/05/05)

- Counting *multi-gene* (multimapping) reads

Major updates in STAR 2.7.8a (2021/02/20)

- Cell calling (filtering) similar to CellRanger:
  - `--soloCellFilter EmptyDrops_CR` option for cell filtering (calling) nearly identical to that of CellRanger 3 and 4
  - `--runMode soloCellFiltering` option for cell filtering (calling) of the raw count matrix, without re-mapping
- Input from BAM files for STARsolo:
  - Input from unmapped or mapped SAM/BAM for STARsolo, with options `--soloInputSAMattrBarcodeSeq` and `--soloInputSAMattrBarcodeQual` to specify SAM tags for the barcode read sequence and qualities
- Read trimming similar to CellRanger4:
  - `--clipAdapterType CellRanger4` option for 5' TSO adapter and 3' polyA-tail clipping of the reads to better match CellRanger >= 4.0.0 mapping results
- Support for barcodes embedded in mates (such as 10X 5' protocol):
  - `--soloBarcodeMate` to support scRNA-seq protocols in which one of the paired-end mates contains both barcode sequence and cDNA (e.g. 10X 5' protocol)

STARsolo

STARsolo is a turnkey solution for analyzing droplet single cell RNA sequencing data (e.g. 10X Genomics Chromium System) built directly into STAR code. STARsolo inputs the raw FASTQ reads files, and performs the following operations

- error correction and demultiplexing of cell barcodes using user-input whitelist
- mapping the reads to the reference genome using the standard STAR spliced read alignment algorithm
- error correction and collapsing (deduplication) of Unique Molecular Identifiers (UMIa)
- quantification of per-cell gene expression by counting the number of reads per gene
- quantification of other transcriptomic features: splice junctions; pre-mRNA; spliced/unspliced reads similar to Velocyto

STARsolo output is designed to be a drop-in replacement for 10X CellRanger gene quantification output. It follows CellRanger logic for cell barcode whitelisting and UMI deduplication, and produces nearly identical gene counts in the same format. At the same time STARsolo is ~10 times faster than the CellRanger.

Running STARsolo for 10X Chromium scRNA-seq data

- STARsolo is run the same way as normal STAR run, with addition of several STARsolo parameters:

  ```
  /path/to/STAR --genomeDir /path/to/genome/dir/ --readFilesIn ...  [...other parameters...] --soloType ... --soloCBwhitelist ...
  ```

  The genome index is the same as for normal STAR runs.
  The parameters required to run STARsolo on 10X Chromium data are described below:

- The STAR solo algorithm is turned on with:

  ```
  --soloType Droplet
  ```

  or, since 2.7.3a, with more descriptive:

  ```
  --soloType CB_UMI_Simple
  ```

- The CellBarcode whitelist has to be provided with:

  ```
  --soloCBwhitelist /path/to/cell/barcode/whitelist
  ```

  The 10X Chromium whitelist files can be found or inside the CellRanger distribution or on GitHub/10XGenomics. Please make sure that the whitelist is compatible with the specific version of the 10X chemistry: V2 or V3. For instance, in CellRanger 3.1.0, the *V2 whitelist* is:

  ```
  cellranger-cs/3.1.0/lib/python/cellranger/barcodes/737K-august-2016.txt

  https://github.com/10XGenomics/cellranger/raw/master/lib/python/cellranger/barcodes/737K-august-2016.txt
  ```

  and *V3 whitelist* (gunzip it for STARsolo):

  ```
  cellranger-cs/3.1.0/lib/python/cellranger/barcodes/3M-february-2018.txt.gz

  https://github.com/10XGenomics/cellranger/raw/master/lib/python/cellranger/barcodes/3M-february-2018.txt.gz
  ```

- The default barcode lengths (CB=16b, UMI=10b) work for 10X Chromium V2. For V3, specify:

  ```
  --soloUMIlen 12
  ```

- Importantly, in the --readFilesIn option, the 1st file has to be cDNA read, and the 2nd file has to be the barcode (cell+UMI) read, i.e.

  ```
  --readFilesIn cDNAfragmentSequence.fastq.gz CellBarcodeUMIsequence.fastq.gz
  ```

  For instance, standard 10X runs have cDNA as Read2 and barcode as Read1:

  ```
  --readFilesIn Read2.fastq.gz Read1.fastq.gz
  ```

  For multiple lanes, use commas separated lists for Read2 and Read1:

  ```
  --readFilesIn Read2_Lane1.fastq.gz,Read2_Lane2.fastq.gz,Read2_Lane3.fastq.gz
  ```

```
Read1_Lane1.fastq.gz,Read1_Lane2.fastq.gz,Read1_Lane3.fastq.gz
```

How to make STARsolo *raw* gene counts (almost) identical to CellRanger's

- CellRanger uses its own "filtered" version of annotations (GTF file) which is a subset of ENSEMBL annotations, with several gene biotypes removed (mostly small non-coding RNA). Annotations affect the counts, and to match CellRanger counts CellRanger annotations have to be used.
- 10X provides several versions of the CellRanger annotations: https://support.10xgenomics.com/single-cell-gene-expression/software/downloads/latest
  For the best match, the annotations in CellRanger run and STARsolo run should be exactly the same.
- The FASTA and GTF files, for one of the older releases:

  ```
  refdata-cellranger-GRCh38-3.0.0/genes/genes.gtf
  refdata-cellranger-GRCh38-3.0.0/fasta/genome.fa
  ```

  or, for the latest release:

  ```
  refdata-gex-GRCh38-2020-A/genes/genes.gtf
  refdata-gex-GRCh38-2020-A/fasta/genome.fa
  ```

  have to be used in STAR genome index generation step before mapping:

  ```
  STAR  --runMode genomeGenerate --runThreadN ... --genomeDir ./ --genomeFastaFiles
  /path/to/genome.fa  --sjdbGTFfile /path/to/genes.gtf
  ```

- If you want to use your own GTF (e.g. newer version of ENSEMBL or GENCODE), you can generate the "filtered" GTF file using 10X's mkref tool: https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/advanced/references
- To make the agreement between STARsolo and CellRanger even more perfect, you can add

  ```
  --genomeSAsparseD 3
  ```

  to the genome generation options, which is used by CellRanger to generate STAR genomes. It will generate sparse suffixs array which has an additional benefit of fitting into 16GB of RAM. However, it also results in 30-50% reduction of speed.
- The considerations above are for *raw* counts, i.e. when cell filtering (calling) is not performed. To get *filtered* cells, refer to Cell filtering (calling) section.

Matching CellRanger 3.x.x results

- By default, cell barcode and UMI collapsing parameters are designed to give the best agreement with CellRanger 2.x.x. CellRanger 3.x.x introduced some minor changes to this algorithm. To get the best agreement between STARsolo and CellRanger 3.x.x, add these parameters:

  ```
  --soloCBmatchWLtype 1MM_multi_Nbase_pseudocounts --soloUMIfiltering MultiGeneUMI_CR --
  soloUMIdedup 1MM_CR
  ```

Matching CellRanger 4.x.x and 5.x.x results

- Starting from CellRanger 4.0, the TSO adapter sequence is clipped from the 5' of the cDNA read, and polyA-tail is trimmed from the 3'. For the best mathch to CellRanger >= 4.0, use these parameters:

```
    --clipAdapterType CellRanger4 --outFilterScoreMin 30
    --soloCBmatchWLtype 1MM_multi_Nbase_pseudocounts --soloUMIfiltering MultiGeneUMI_CR --
    soloUMIdedup 1MM_CR
```

The adapter clipping utilizes vectorized Smith-Waterman algorithm from Opal package by Martin Šošić: https://github.com/Martinsos/opal

Barcode geometry

Simple barcodes

Simple barcode lengths and start positions on barcode reads are described with

```
--soloCBstart, --soloCBlen, --soloUMIstart, --soloUMIlen
```

which works with

```
--soloType CB_UMI_Simple (a.k.a Droplet)
```

Barcode and cDNA on the same mate

By default, it is assumed that the barcode is located on one of the mates of paired-end read, while cDNA is on the other mate. However, in some scRNA-seq protocols the barcode and cDNA sequences are located on the same mate. In this case, we can specify the mate on which the barcode is located (1 or 2) with `--soloBarcodeMate` . Also, the barcode/adapter sequences have to be clipped (leaving only cDNA) with `--clip5pNbases` or `--clip3pNbases` .

For instance, for the **10X 5' protocol**, the 1st mate contains the barcode at the 5', with 16b CB, 10b UMI and 13b adapter (39b total). If the 1st mate is sequenced longer than 39b, the remaining bases are cDNA that can be mapped together with the 2nd mate (which contains only cDNA):

```
--soloBarcodeMate 1   --clip5pNbases 39 0
--soloType CB_UMI_Simple   --soloCBstart 1   --soloCBlen 16   --soloUMIstart 17   --soloUMIlen
10
--readFilesIn read1.fq read2.fq
```

Note that the read files are input in the Read1 Read2 order in this case.

Complex barcodes

More complex barcodes are activated with `--soloType CB_UMI_Complex` and are described with the following parameters

```
soloCBposition              -
strings(s)              position of Cell Barcode(s) on the barcode read.
                        Presently only works with --soloType CB_UMI_Complex, and barcodes are
assumed to be on Read2.
                        Format for each barcode:
startAnchor_startDistance_endAnchor_endDistance
                        start(end)Anchor defines the anchor base for the CB: 0: read start; 1:
read end; 2: adapter start; 3: adapter end
                        start(end)Distance is the distance from the CB start(end) to the Anchor
base
                        String for different barcodes are separated by space.
                        Example: inDrop (Zilionis et al, Nat. Protocols, 2017):
                        --soloCBposition  0_0_2_-1  3_1_3_8

soloUMIposition             -
string                  position of the UMI on the barcode read, same as soloCBposition
```

```
                        Example: inDrop (Zilionis et al, Nat. Protocols, 2017):
                        --soloUMIposition  3_9_3_14

soloAdapterSequence          -
string:                 adapter sequence to anchor barcodes.

soloAdapterMismatchesNmax   1
int>0:                  maximum number of mismatches allowed in adapter sequence
```

Cell filtering (calling)

In addition to raw, unfiltered output of gene/cell counts, STARsolo performs cell filtering (a.k.a. cell calling), which aims to select a subset of cells that are likely to be "real" cells as opposed to empty droplets (containing ambient RNA). Two types of filtering are presently implemented: simple (knee-like) and advanced EmptyDrop-like. The selected filtering is also used to produce summary statistics for filtered cells in the Summary.csv file, which is similar to CellRanger's summary and is useful for Quality Control.

Knee filtering

Knee filtering is similar to the method used by CellRanger 2.2.x. This is turned on by default and is controlled by:

```
--soloCellFilter  CellRanger2.2
```

You can also add three numbers for this option (default values are given in parenthesis): the number of expected cells (3000), robust maximum percentile for UMI count (0.99), maximum to minimum ratio for UMI count (10).

EmptyDrop-like filtering

CellRanger 3.0.0 use advanced filtering based on the EmptyDrop algorithm developed by Lun et al. This algorithm calls extra cells compared to the knee filtering, allowing for cells that have relatively fewer UMIs but are transcriptionally different from the ambient RNA. In STARsolo, this filtering can be activated by:

```
--soloCellFilter  EmptyDrops_CR
```

It can be followed by 10 numeric parameters: nExpectedCells (3000), maxPercentile (0.99), maxMinRatio (10), indMin (45000), indMax (90000), umiMin (500), umiMinFracMedian (0.01), candMaxN (20000), FDR (0.01), simN (10000).

Cell filtering of previously generated raw matrix

It is possible to run only the filtering algorithm (without the need to re-map) inputting the previously generated **raw** matrix:

```
STAR --runMode soloCellFiltering  /path/to/count/dir/raw/   /path/to/output/prefix   --
soloCellFilter EmptyDrops_CR
```

The */path/to/count/dir/raw/* directory should contain the **"raw"** *barcodes.tsv*, *features.tsv*, and *matrix.mtx* files generated in a previos STARsolo run. The output will contain the filtered files.

Quantification of different transcriptomic features

- In addition to the gene counts (deafult), STARsolo can calculate counts for other transcriptomic features:
  - pre-mRNA counts, useful for single-nucleus RNA-seq. This counts all read that overlap gene loci, i.e. included both exonic and intronic reads:

    ```
    --soloFeatures GeneFull
    ```

  - Counts for annotated and novel splice junctions:

```
--soloFeatures SJ
```

- **Velocyto spliced/unspliced/ambiguous quantification**

  This option will calculate Spliced, Unspliced, and Ambiguous counts per cell per gene similar to the velocyto.py tool developed by LaManno et al. This option is under active development and the results may change in the future versions.

  ```
  --soloFeatures Gene Velocyto
  ```

  Note that Velocyto quantification requires Gene features
- All the features can be conveniently quantified in one run:

  ```
  --soloFeatures Gene GeneFull SJ Velocyto
  ```

Multi-gene reads

Multi-gene reads are concordant with (i.e. align equally well to) transcripts of two or more genes. One class of multi-gene read are those that map uniquely to a genomic region where two or more genes overlap. Another class are those reads that map to multiple loci in the genome, with each locus annotated to a different gene.

Including multi-gene reads allows for more accurate gene quantification and, more importantly, enables detection of gene expression from certain classes of genes that are supported only by multi-gene reads, such as overlapping genes and highly similar paralog families.

The multi-gene read recovery options are specified with `--soloMultiMappers`. Several algorithms are implemented:

```
--soloMultiMappers Uniform
```

uniformly distributes the multi-gene UMIs to all genes in its gene set. Each gene gets a fractional count of 1/N_genes, where N_genes is the number of genes in the set. This is the simplest possible option, and it offers higher sensitivity for gene detection at the expense of lower precision.

```
--soloMultiMappers PropUnique
```

distributes the multi-gene UMIs proportionally to the number of unique UMIs per gene. UMIs that map to genes that are not supported by unique UMIs are distributed uniformly.

```
--soloMultiMappers EM
```

uses Maximum Likelihood Estimation (MLE) to distribute multi-gene UMIs among their genes, taking into account other UMIs (both unique- and multi-gene) from the same cell (i.e. with the same CB). Expectation-Maximization (EM) algorithm is used to find the gene expression values that maximize the likelihood function. Recovering multi-gene reads via MLE-EM model was previously used to quantify transposable elements in bulk RNA-seq {TEtranscripts} and in scRNA-seq {Alevin; Kallisto-bustools}.

```
--soloMultiMappers Rescue
```

distributes multi-gene UMIs to their gene set proportionally to the sum of the number of unique-gene UMIs and uniformly distributed multi-gene UMIs in each gene Mortazavi et al. It can be thought of as the first step of the EM algorithm.

Any combination of these options can be specified and different multi-gene falvors will be output into different files. The unique-gene UMI counts are output into the *matrix.mtx* file in the *raw/Gene* directory, while the sum of unique+multi-gene UMI counts will be output into *UniqueAndMult-EM.mtx, UniqueAndMult-PropUnique.mtx, UniqueAndMult-Rescue.mtx, UniqueAndMult-Uniform.mtx* files.

BAM tags

- To output BAM tags into SAM/BAM file, add them to the list of standard tags in

  ```
  --outSAMattributes NH HI nM AS CR UR CB UB GX GN sS sQ sM
  ```

  Any combinations of tags can be used.
- CR/UR: **raw (uncorrected)** CellBarcode/UMI
- CY/UY: quality score for CellBarcode/UMI
- GX/GN: for gene ID/names
- sS/sQ: for sequence/quality combined CellBarcode and UMI; sM for barcode match status.
- CB/UB: **corrected** CellBarcode/UMI. Note, that these tags require sorted BAM output, i.e. we need to add the following option:

  ```
  --outSAMtype BAM SortedByCoordinate
  ```

Input reads from BAM files.

The read sequences and barcodes can be input from SAM (or BAM) files, both unmapped (uBAM) and previously mapped (e.g. Cellranger's BAM):

```
--readFilesIn input.bam --readFilesType SAM SE
```

In case of BAM files, use `samtools view` command to convert to BAM:

```
--readFilesCommand samtools view -F 0x100
```

The file should contain one line for each read. For previously mapped file it can be achieved by filtering out non-primary alignments as shown above. Note that unmapped reads have to be included in the file to be remapped. We need to specify which SAM attributes correspond to seqeunces/qualities of cell barcodes (CR/CY) and UMIs (UR/UY):

```
--soloInputSAMattrBarcodeSeq CR UR    --soloInputSAMattrBarcodeQual CY UY
```

If you wish to omit some, All or None of the SAM attributes in the output BAM file (if you requested one), use `--readFilesSAMattrKeep` option. For previously mapped files, `--readFilesSAMattrKeep None` is often the best option to avoid duplicated SAM attributes in the BAM output.

If you request coordinate-sorted BAM output, and use a coordinate-sorted mapped BAM input (such as CellRanger's possorted BAM), it may result in slow sorting and require large amountss of RAM. In this case, it is recommended to shuffle the alignments before mapping with `samtools bamshuf` command.

Different scRNA-seq technologies

Plate-based (Smart-seq) scRNA-seq

Plate-based (Smart-seq) scRNA-seq technologies produce separate FASTQ files for each cell. Cell barcodes are not incorporated in the read sequences, and there are no UMIs. Typical STAR command for mapping and quantification of these file will look like:

```
--soloType SmartSeq --readFilesManifest /path/to/manifest.tsv --soloUMIdedup Exact --soloStrand
Unstranded
```

- STARsolo `--soloType SmartSeq` option produces cell/gene (and other features) count matrices, using rules similar to the droplet-based technologies. The differnces are (i) individual cells correspond to different FASTQ files,there are no Cell Barcode sequences, and "Cell IDs" have to be provided as input (ii) there are no UMI sequences, but reads can be deduplicated if they have identical start/end coordinates.
- The convenient way to list all the FASTQ files and Cell IDs is to create a file manifest and supply it in `--readFilesManifest /path/to/manifest.tsv`. The manifest file should contain 3 tab-separated columns. For paired-end reads:

```
Read1-file-name \t Read2-file-name \t Cell-id
```

For single-end reads, the 2nd column should contain the dash - :

```
Read1-file-name \t - \t Cell-id
```

Cell-id can be any string without spaces. Cell-id will be added as ReadGroup tag (*RG:Z:*) for each read in the SAM/BAM output. If Cell-id starts with *ID:*, it can contain several fields separated by tab, and all the fields will be copied verbatim into SAM *@RG* header line.

- Deduplication based on read start/end coordinates can be done with `--soloUMIdedup Exact` option. To avoid deduplication (e.g. for single-end reads) use `--soloUMIdedup NoDedup`. Both deduplication options can be used together `--soloUMIdedup Exact NoDedup` and will produce two *.mtx* matrices.
- Common Smart-seq protocols are unstranded and thus will require `--soloStrand Unstranded` option. If your protocol is stranded, you can can choose the proper `--soloStrand Forward` (default) or `--soloStrand Reverse` options.

All parameters that control STARsolo output are listed again below with defaults and short descriptions:

```
soloType                    None
    string(s): type of single-cell RNA-seq
                            CB_UMI_Simple   ... (a.k.a. Droplet) one UMI and one Cell Barcode
of fixed length in read2, e.g. Drop-seq and 10X Chromium.
                            CB_UMI_Complex  ... one UMI of fixed length, but multiple Cell
Barcodes of varying length, as well as adapters sequences are allowed in read2 only, e.g.
inDrop.
                            CB_samTagOut    ... output Cell Barcode as CR and/or CB SAm tag. No
UMI counting. --readFilesIn cDNA_read1 [cDNA_read2 if paired-end] CellBarcode_read . Requires -
-outSAMtype BAM Unsorted [and/or SortedByCoordinate]
                            SmartSeq        ... Smart-seq: each cell in a separate FASTQ
(paired- or single-end), barcodes are corresponding read-groups, no UMI sequences, alignments
deduplicated according to alignment start and end (after extending soft-clipped bases)

soloCBwhitelist             -
    string(s): file(s) with whitelist(s) of cell barcodes. Only --soloType CB_UMI_Complex
allows more than one whitelist file.
                            None            ... no whitelist: all cell barcodes are allowed

soloCBstart                 1
```

```
    int>0: cell barcode start base

soloCBlen                   16
    int>0: cell barcode length

soloUMIstart                17
    int>0: UMI start base

soloUMIlen                  10
    int>0: UMI length

soloBarcodeReadLength       1
    int: length of the barcode read
                            1   ... equal to sum of soloCBlen+soloUMIlen
                            0   ... not defined, do not check

soloBarcodeMate             0
    int: identifies which read mate contains the barcode (CB+UMI) sequence
                            0   ... barcode sequence is on separate read, which should always
be the last file in the --readFilesIn listed
                            1   ... barcode sequence is a part of mate 1
                            2   ... barcode sequence is a part of mate 2

soloCBposition              -
    strings(s)              position of Cell Barcode(s) on the barcode read.
                            Presently only works with --soloType CB_UMI_Complex, and barcodes
are assumed to be on Read2.
                            Format for each barcode:
startAnchor_startPosition_endAnchor_endPosition
                            start(end)Anchor defines the Anchor Base for the CB: 0: read start;
1: read end; 2: adapter start; 3: adapter end
                            start(end)Position is the 0-based position with of the CB
start(end) with respect to the Anchor Base
                            String for different barcodes are separated by space.
                            Example: inDrop (Zilionis et al, Nat. Protocols, 2017):
                            --soloCBposition  0_0_2_-1  3_1_3_8

soloUMIposition             -
    string                  position of the UMI on the barcode read, same as soloCBposition
                            Example: inDrop (Zilionis et al, Nat. Protocols, 2017):
                            --soloCBposition  3_9_3_14

soloAdapterSequence         -
    string:                 adapter sequence to anchor barcodes.

soloAdapterMismatchesNmax   1
    int>0:                  maximum number of mismatches allowed in adapter sequence.

soloCBmatchWLtype           1MM_multi
    string:                 matching the Cell Barcodes to the WhiteList
                            Exact                           ... only exact matches allowed
                            1MM                             ... only one match in whitelist
with 1 mismatched base allowed. Allowed CBs have to have at least one read with exact match.
                            1MM_multi                       ... multiple matches in whitelist
with 1 mismatched base allowed, posterior probability calculation is used choose one of the
matches.
                                                                Allowed CBs have to have at
least one read with exact match. This option matches best with CellRanger 2.2.0
                            1MM_multi_pseudocounts          ... same as 1MM_Multi, but
pseudocounts of 1 are added to all whitelist barcodes.
                            1MM_multi_Nbase_pseudocounts    ... same as 1MM_multi_pseudocounts,
multimatching to WL is allowed for CBs with N-bases. This option matches best with CellRanger
>= 3.0.0


soloInputSAMattrBarcodeSeq  -
    string(s):              when inputting reads from a SAM file (--readsFileType SAM SE/PE),
these SAM attributes mark the barcode sequence (in proper order).
                            For instance, for 10X CellRanger or STARsolo BAMs, use --
soloInputSAMattrBarcodeSeq CR UR .
```

```
                            This parameter is required when running STARsolo with input from
SAM.

soloInputSAMattrBarcodeQual  -
    string(s):              when inputting reads from a SAM file (--readsFileType SAM SE/PE),
these SAM attributes mark the barcode qualities (in proper order).
                            For instance, for 10X CellRanger or STARsolo BAMs, use --
soloInputSAMattrBarcodeQual CY UY .
                            If this parameter is '-' (default), the quality 'H' will be
assigned to all bases.

soloStrand                  Forward
    string: strandedness of the solo libraries:
                            Unstranded  ... no strand information
                            Forward     ... read strand same as the original RNA molecule
                            Reverse     ... read strand opposite to the original RNA molecule

soloFeatures                Gene
    string(s):              genomic features for which the UMI counts per Cell Barcode are
collected
                            Gene            ... genes: reads match the gene transcript
                            SJ              ... splice junctions: reported in SJ.out.tab
                            GeneFull        ... full genes: count all reads overlapping genes'
exons and introns

soloUMIdedup                1MM_All
    string(s):              type of UMI deduplication (collapsing) algorithm
                            1MM_All                     ... all UMIs with 1 mismatch distance
to each other are collapsed (i.e. counted once).
                            1MM_Directional_UMItools    ... follows the "directional" method
from the UMI-tools by Smith, Heger and Sudbery (Genome Research 2017).
                            1MM_Directional             ... same as 1MM_Directional_UMItools,
but with more stringent criteria for duplicate UMIs
                            Exact                       ... only exactly matching UMIs are
collapsed.
                            NoDedup                     ... no deduplication of UMIs, count all
reads.
                            1MM_CR                      ... CellRanger2-4 algorithm for 1MM UMI
collapsing.

soloUMIfiltering            -
    string(s)               type of UMI filtering
                            -               ... basic filtering: remove UMIs with N and
homopolymers (similar to CellRanger 2.2.0).
                            MultiGeneUMI    ... basic + remove lower-count UMIs that map to
more than one gene.
                            MultiGeneUMI_CR ... basic + remove lower-count UMIs that map to
more than one gene, matching CellRanger > 3.0.0 .
                                                Only works with --soloUMIdedup 1MM_CR

soloOutFileNames            Solo.out/          features.tsv barcodes.tsv        matrix.mtx
    string(s)               file names for STARsolo output:
                            file_name_prefix   gene_names   barcode_sequences
cell_feature_count_matrix

soloCellFilter              CellRanger2.2 3000 0.99 10
    string(s):              cell filtering type and parameters
                            None            ... do not output filtered cells
                            TopCells        ... only report top cells by UMI count, followed by
the exact number of cells
                            CellRanger2.2   ... simple filtering of CellRanger 2.2.
                                                Can be followed by numbers: number of expected
cells, robust maximum percentile for UMI count, maximum to minimum ratio for UMI count
                                                The harcoded values are from CellRanger:
nExpectedCells=3000;  maxPercentile=0.99;  maxMinRatio=10
                            EmptyDrops_CR   ... EmptyDrops filtering in CellRanger flavor.
Please cite the original EmptyDrops paper: A.T.L Lun et al, Genome Biology, 20, 63 (2019):
https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1662-y
                                                Can be followed by 10 numeric parameters:
nExpectedCells   maxPercentile   maxMinRatio   indMin   indMax   umiMin   umiMinFracMedian
```

```
candMaxN   FDR   simN
                                                The harcoded values are from CellRanger:
3000            0.99            10    45000    90000      500               0.01      20000
0.01  10000

soloOutFormatFeaturesGeneField3 "Gene Expression"
        string(s):                              field 3 in the Gene features.tsv file. If "-",
then no 3rd field is output.
```