Velopipe

Velopipe is a RNA Vecocity pipeline for SEQC.

Understanding the Overall Workflow

1. The BAM file from SEQC needs to be reformatted (to comply with the GA4GH standard) so that each read in the BAM file has Chromium cellular and molecular barcode information attached. Due to the way SEQC handles reads, the previous versions of SEQC (<0.2.9) is not able to do this properly, thus the very first step is reprocessing the sample using the latest version of SEQC (0.2.9+). The new versions of SEQC will generate a file having pre-/post-correction CB/UMI information.
2. The pre-/post-correction CB/UMI information will be attached to the BAM file (aka. tagging BAM or attaching barcodes to BAM).
3. Velocyto will be run with the barcode attached BAM file as input. This will generate a loom file with spliced/unspliced reads flagged.

Notes

The new SEQC has some bug fixes. One of the biggest change was, the data type used to handle UMI was incorrect in the previous versions of SEQC. Sometime you would see negative numbers for DNA3Bit encoded UMI. Of course, in this case you won't be able to decode it back to the ACGT form of UMI sequence - DNA3Bit would throw an error. Due to this issue, UMI collapsing was sometimes incorrectly done, causing some count changes in sparse/dense matrix as well as in ReadArray.

Since we're rerunning the latest SEQC here, you might observe some increase/decrease in the number of cells/molecules/and etc., sometimes a negligible change, sometimes up to hundreds +/- changes, all depends on the samples being processed.

Preparing Job File

You need to provide two JSON files that describes your job:

${sample-name}.inputs.json

```
{
    "Velopipe2.sampleName": "RU263",
    "Velopipe2.filteredBarcodes": "s3://dp-lab-
data/collaborators/pi/project/sample/..._dense.csv",
    "Velopipe2.bam": "s3://dp-lab-
data/collaborators/pi/project/sample/..._Aligned.out.sorted.bam",
    "Velopipe2.numOfChunks": 100,
    "Velopipe2.gtf": "s3://seqc-public/genomes/mm38_long_polya/annotations.gtf",
    "Velopipe2.fullBarcodeWhitelist": "s3://seqc-public/barcodes/ten_x_v3/flat/3M-
february-2018.txt",
    "Velopipe2.cbCorrection": "s3://dp-lab-test/seqc/joe/Ru263/seqc-results-
v2/941_Ru263_IGO_09507_10_cb-correction.csv.gz",
    "Velopipe2.umiCorrection": "s3://dp-lab-test/seqc/joe/Ru263/seqc-results-
v2/941_Ru263_IGO_09507_10_umi-correction.csv.gz"
}
```

- `Velopipe2.filteredBarcodes`: a file from which filtered barcodes will be extracted:
  - SEQC dense (filtered) count matrix (e.g. `*_dense.csv`) or
  - SEQC sparse (raw) count matrix (e.g. `*_sparse_counts_barcodes.csv`)
  - Cell Ranger barcodes (e.g. `barcodes.tsv.gz`)
  - Custom filtered nucleotide barcode list (`custom-filtered-barcodes.acgt.txt`)
- `Velopipe2.bam`: BAM file generated by SEQC
- `Velopipe2.numOfChunks`: BAM file will be split into multiple chunks to paralleize.
- `Velopipe2.gtf`: GTF file:
  - Human: `s3://seqc-public/genomes/hg38_long_polya/annotations.gtf`
  - Mouse: `s3://seqc-public/genomes/mm38_long_polya/annotations.gtf`
- `Velopipe2.fullBarcodeWhitelist`: 10x official barcode whitelist:
  - 10x v2: `s3://seqc-public/barcodes/ten_x_v2/flat/737K-august-2016.txt`
  - 10x v3: `s3://seqc-public/barcodes/ten_x_v3/flat/3M-february-2018.txt`
- `Velopipe2.cbCorrection`: CB correction file generated by SEQC
- `Velopipe2.umiCorrection`: UMI correction file generated by SEQC

(*) Note that SEQC produces a read-name sorted BAM file (this is different from position sorted).

(*) For Velocyto, do not use the GTF file specifically created for the single-nuclei RNA-seq assay (e.g. `s3://seqc-public/genomes/hg38_long_polya_snRNAseq/annotations.gtf`). This file is modified such a way that SEQC or Cell Ranger can count intronic reads, but it is not compatible with Velocyto. You will get an error such as:

```
The entry exon_number was not present in the gtf file.
```

${sample-name}.labels.json

```
{
    "pipelineType": "Velopipe2",
    "project": "Project 193",
    "sample": "RU263",
    "owner": "chunj",
    "destination": "s3://dp-lab-test/seqc/joe/Ru263/velopipe",
    "transfer": "-",
    "comment": ""
}
```

- `project`: project ID retrieved from SCRI database
- `sample`: sample name
- `destination`: AWS S3 location where the final output files (e.g. loom) should be saved