Running SeqWellTCRAnalysis in the cloud

Purpose: The easiest way right now to analyze TCR sequencing results from Tu, et al. (NI 2019) is to run the analysis on cloud via docker (or another equivalently powerful server/computer, also running docker). While it might be possible to run the source code directly (via Matlab console), we do not recommend it, as it may take a long time.

Before running the analysis, the raw sequencing file has to be first mapped (by BWA) to TCR references. Please see “fastqprocessing” on github for more details.

It is also recommended to supply a list of cell barcodes from RNAseq analysis. This will shorten the analysis by limiting it only to cell barcodes that are in the supplied list. The list should be in single column tsv format with no header.

All input files should be uploaded onto a gcloud bucket. This include the sorted Bam file, the kickoff file (see example on github), and the cell barcode list (if using).

We are currently working on migrating the workflow onto Broad Terra, and into a more efficient combination of WDL and Python. Stay tune to the Love Lab github for updates.

Spin up instance on Google cloud – console.cloud.google.com: Once an appropriate account with billing (or free trial credit) has been set up with gcloud, navigate to the console page.

1. Under Compute Engine, select VM instances
2. Select ‘Create VM instance’
3. In creation menu change:
    a. Machine type – Recommend at least 32 core -120GB RAM standard machine
    b. Check box for Deploy a container image to this VM instance
    c. Increase Boot Disk size to 100 GB
    d. Select Allow full access to all Cloud APIs
    e. Add the docker image address (mitlovelab/seqwelltcranalysis) to the image creation path
4. Hit ‘Create’
5. Once created, hit ssh box next to instance name to connect to the machine
6. At commandline type: docker run –it mitlovelab /seqwelltcranalysis bash
    a. This starts the container, with a bash interface. Type ‘ls’ to see the folders available in the default container
    b. We recommend making a data folder (mkdir data) to store all input files that will be copied over in the next step
7. Once inside the container, use gsutil cp command to download all the data files and the input file onto the machine (from your glcoud bucket)
    a. **Important** – Make sure the place you put the datafiles is the same exact path you put in the input file for the code.
8. Kickoff analysis. Type: –

    SeqWellTCRAnalysis [full path name of kickoff file] [#of cores to use (depend on the instance)]
9. Once done, use gsutil cp command to move results back to cloud storage.

    a. The resulting data will be stored in the same folder where the kickoff file is stored in the container. There will be a rather large number of intermediate files.
    b. The files needed for integration with Seurat object is the ‘MappingSummary’ file.
    c. SummaryPlots can also be downloaded to see preliminary qc of the data, though not necessary.
    d. Info.mat file contains some intermediate info (number of cores used, number of barcodes detected from the supplied barcode list, etc.). Can also be downloaded for trouble-shooting.

10. **Important** – After the analysis is finished, shut down VM in google cloud console. This will never shut down on its own so you will be paying by the hour for it.
11. **During the analysis, you can quit the ssh session**, then later, when ssh back into the session, use docker ps –s to see the running session, and then use docker attach to attach back into the session.
    a. CPU usage of the instance can be monitored from the VM Instance page. Expect CPU usage to increase during analysis, then fall down to near 0 after the analysis is finished.

Known quarks and issues: This version of code has several known issues and quarks, most of which can be dealt with by modifying the instance parameters.

1. **Random numbers being outputted to console during analysis.** Currently the code outputs number of reads found for each of the barcodes it has processed. This is for troubleshooting purposes, and in updated version will be corrected.
2. **Code complaining about not able to write to disk.** We have noticed that the docker container often needs a lot more allocated storage space than anticipated. Increase the disk space allocation to solve the issue (as disk space is relatively cheap, we often allocate 1-2TB of disk space to the instance, just to avoid the issue).
3. **Code complaining not finding a certain distributed job summary.** In instances where not a lot of cell barcodes are being analyzed, we noticed that sometimes the code would not distribute the jobs correctly, resulting in some cores not getting a job allocated. Check the intermediate ‘core’ files (denoted by ‘core’ followed by a number from 1- the number of allocated cores) to see how many jobs were successfully distributed, then change the number of allocated cores (during kickoff) to that number. We expect this issue to be fixed once moved to Terra.