Steps for NetZen pipeline analysis from fastq raw data

#1)cd into working directory

#cd /orange/dtran/Collaborator/DJ/TTF\_resistance\_2

# 2) Create file list

touch sample\_fastq\_table.csv

echo -e sample$'\t'fastq\_file > sample\_fastq\_table.csv

ls --color=none ./data/\*.gz >> sample\_fastq\_table.csv # should include the –color=none option to avoid writing out the color escape sequences.

## 3) Manual Edit sample\_fastq\_table.csv so that it contains two columns sample, fastq\_file

MAKE SURE THAT SAMPLE NAMES DO NOT CONTAIN SPACE OR : CHARACTERS. The sample names will be used downstream in creating contrast name, etc. , making directory and Linux OS will have hard time to use folder name or file name with space character.



## 4) Create the sample\_group\_table.csv file (tab delimited). The Sample Name in this table should match the name in sample\_fastq\_table.csv file

Note: Sample Group table can include several factors, including description. But if use heterogeneous public datasets from different lab with batch effect removal process, then description column should be excluded.

# 5) Create baseline table (baseline\_table.csv) with two columns: SampleID and Baseline\_samples. SampleID and Baseline\_samples columns should be separated by tab, not by comma. Baseline\_samples: comma separated list of control to compare with.



# 6) Create design table file (generated using R and sample\_group\_table using model.matrix function).

The table should have row names (sampleIDs), no column for sample IDs. Example of design table in Examples/design.csv. Complicated design would need interactions. Design interaction by formula. For more help, search help for model.matrix function in Rstudio. Examle of script to generate design table in Examples/get\_design\_matrix\_one\_factor.R for one factor or Examples/get\_design\_matrix\_multi\_factors.R. If there are multiple factors, then adjust the orders of levels of factors to the desired order (Control is in the first place) by using command: sample$factor1 = factor(sample$factor1, levels=c(“level1”, “level2”…)) where “level1” usually is a control. For example:

sample\_groups$Treatment = factor(sample\_groups$Treatment, levels = c("SC", "M", "N","P", "MN", "MP", "MNP"))

#module load R

#Rscript get\_design\_matrix.R

If not provided (design=Null) then autogenerate design table based on sample\_factor\_info, using simple linear model where each factor is independent, no interactions. design table :rows are sample IDs, columns are factors of experiments. This autogenerated option is recommended for batch removal process in heterogeneous public datasets. Then the pipeline should be stop at the get\_fit\_data\_from\_count\_step to look at autogenerated design table for creation of contrast\_network\_table\_with\_baseline\_table.csv based on autogenerated design.csv table.

After that, the pipeline should be resume from the next step ( get\_network\_layout\_from\_fit\_data)



This is similar to the Examples/get\_design\_matrix\_multi\_factors.R. If not for batch effect removal (when you analyze dataset from one batch), than it’s better to manualy generate design table using Examples/get\_design\_matrix\_multi\_factors.R rather than autogenerate as then you don’t need to stop the pipeline.

Check the design table for full rank. Sometimes the experiment does not have enough datapoints to solve the linear equations. Therefore need to check design table to see if this is the case using script in Example/check\_full\_rank.R

## 7)Manually Create contrast\_network\_table\_with\_baseline\_table.csv based on design.csv table



Each row is a contrast to compare.

Name: Name of comparison

Contrast: the presence of each factor in comparison, the order should be exactly the same as the order of factor columns in design table. Contrasts are separated by comm “,” without any space

Network: Name of network

baseline\_table: Name of baseline table

Most important here is the contrast column.

#8) Copy network file as there are many versions of network file without header row in the format: gene1,gene2,MI

(from file consolidated\_tpm\_table\_translated.filtered.cyto.csv, rename and remove header)

mv consolidated\_tpm\_table\_translated.cyto.csv network.csv

sed -i '1d' network.csv

# 9) Create configuration file

# Need to make run\_pipe available first through source ~/.bash\_profile (check bash\_bash\_profile first)

run\_pipe -n <pipeline\_name (NetZen\_from\_fastq for example)>

10) Run pipeline

Copy run.sh in Examples folder

Edit run.sh script to followed:

7) run script

submit to cluster:

sbatch run.sh (in the Examples folder)

#!/bin/bash

#SBATCH --job-name=run\_pipe # Job name

#SBATCH --mail-type=END,FAIL # Mail events (NONE, BEGIN, END, FAIL, ALL)

#SBATCH --ntasks=1 # Run on a single CPU

#SBATCH --mem=18gb # Job memory request

#SBATCH --time=48:00:00 # Time limit hrs:min:sec

#SBATCH --output=run\_pipe%j.log # Standard output and error log

pwd; hostname; date

module purge

module load gcc/8.3.0 python/3.9.2

echo "Python version"

python --version

python $SOURCE/run\_pipe -r NetZen\_from\_fastq\_run\_config.conf

Note: if using heterogeneous public samples, need to run two time. The first run to autogenerate design table, the step config should stop at get\_fit\_data\_from\_count\_step. For example:

steps = get\_fit\_data, get\_fit\_data\_from\_count\_step, -get\_comparisons, -get\_network\_layout\_from\_fit\_data, -get\_subnets\_from\_network\_layout, -get\_subnet\_image\_jobs, -get\_network\_data\_step, -save\_network\_image

After manually creating contrast\_network\_table\_with\_baseline\_table.csv , rerun with config:

steps = -get\_fit\_data, -get\_fit\_data\_from\_count\_step, get\_comparisons, get\_network\_layout\_from\_fit\_data, get\_subnets\_from\_network\_layout, get\_subnet\_image\_jobs, get\_network\_data\_step, save\_network\_image

***List of files as inputs into the pipeline:***

1) sample\_fastq\_table.csv

2) sample\_group\_table.csv

3) baseline\_table.csv

4) design.csv

5) contrast\_network\_table\_with\_baseline\_table.csv

6) network.csv : network file

7)\*.conf : pipeline configuration file

8) run.sh

8) Creating Web App:

submit to cluster the script

$NETZEN/make\_Web\_app <dataset\_name> <data\_folder> <outdir>

where dataset\_name is the name of dataset/experiment

data\_folder: folder of NETZEN analysis, parent of COUNTS folder.

Outdir: output directory to put the NETZEN WebApp

**How to combine many Netzen webapp into one integrated webapp:**

1) Copy the build:

$NETZEN: global enviromental variable

$outdir: output directory

outdir = NETZEN\_WebApp

rsync -azhuP $NETZEN/../build/\* ${outdir}

Copy or make simbolic link of data folder of each dataset into the data folder of integrated webapp

Do this for each dataset that you want to be included in the integrated webapp:

ln -s <NETZEN\_WebApp/data/${dataset}> ${outdir}/data/

2) Change the datasets.csv in the current folder to point to the datasets.

3) Export the datassets to html format:

FILE=${outdir}/datasets.csv

$NETZEN/json\_export ${FILE} ${outdir}/datasets.html

Example for combine datasets script is: /home/son.le/SOURCE\_CODE/source/UserGuide/Examples/make\_datasets.sh

**# To run any pipeline:**

./run.sh

copy , edit and execute run.sh script in Example folder. The conf file examples are in

$SOURCE/pipelines/config\_templates

#!/bin/sh

module purge

module load python

run\_pipe -r [name\_of\_config\_file]

Example:

#!/bin/sh

module purge

module load python

run\_pipe -r consolidated\_count\_from\_sra\_run\_config.conf

To get consolidated\_count: template: fastq\_to\_consolidated\_counts