UTAP Documentation

Release 1.0.0

Refael Kohen

Apr 14, 2020

Contents

1 UTAP demo website **3**

2 Analysis pipeline steps and reports **5**
2.1 Analysis pipeline steps 5
2.2 Pipeline report 6
2.3 Annotation file 7
2.4 Examples of reports 7

3 Installation **9**
3.1 Requirements 9
3.2 Create a directory for UTAP software and its output 9
3.3 Create genomes 10
3.4 Run UTAP 12

4 User guide **15**
4.1 Registration to the system 15
4.2 User datasets 16
4.3 Import Input data 16
4.4 Run analysis 18
4.5 Customization 24

5 Releases **25**

6 Source code **27**
6.1 UTAP source code 27
6.2 Dependencies 27

7 License **29**

8 Author **31**

9 Acknowlegments **33**

RNA-Seq technology is routinely used to characterize the transcriptome and detect gene expression differences among cell types, genotypes and conditions. Advances in short-read sequencing instruments such as Illumina Next-Seq, have yielded easy-to-operate machines, with higher throughput, at a lower price per base. However, processing this data requires bioinformatics expertise to tailor and execute specific solutions for each type of library preparation.

In order to enable fast and user-friendly data analysis, we developed an intuitive and scalable transcriptome pipeline that executes the full process, starting from sequences (RNA-Seq and bulk MARS-Seq), and ending with sets of differentially expressed genes. Output files are placed in a structured folder system, and summarization of the results is displayed in a rich and comprehensive report containing dozens of plots, tables and links.

CHAPTER 1

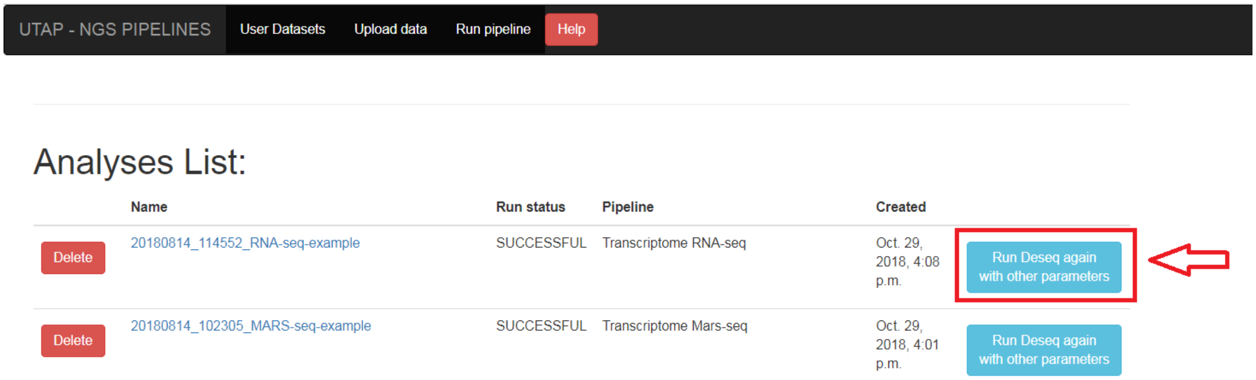
UTAP demo website

Navigate to http://utap-demo.weizmann.ac.il, and login using:

username: testuser

password: utap1234

Note: This is a demonstration site for UTAP, showcasing the user interface and examples of run results. It provides partial functionality, including the capability to rerun the DESeq step on existing analyses, but does not support running new analyses.



Analyses List:

	Name	Run status	Pipeline	Created	
Delete	20180814_114552_RNA-seq-example	SUCCESSFUL	Transcriptome RNA-seq	Oct. 29, 2018, 4:08 p.m.	Run Deseq again with other parameters
Delete	20180814_102305_MARS-seq-example	SUCCESSFUL	Transcriptome Mars-seq	Oct. 29, 2018, 4:01 p.m.	Run Deseq again with other parameters

CHAPTER 2

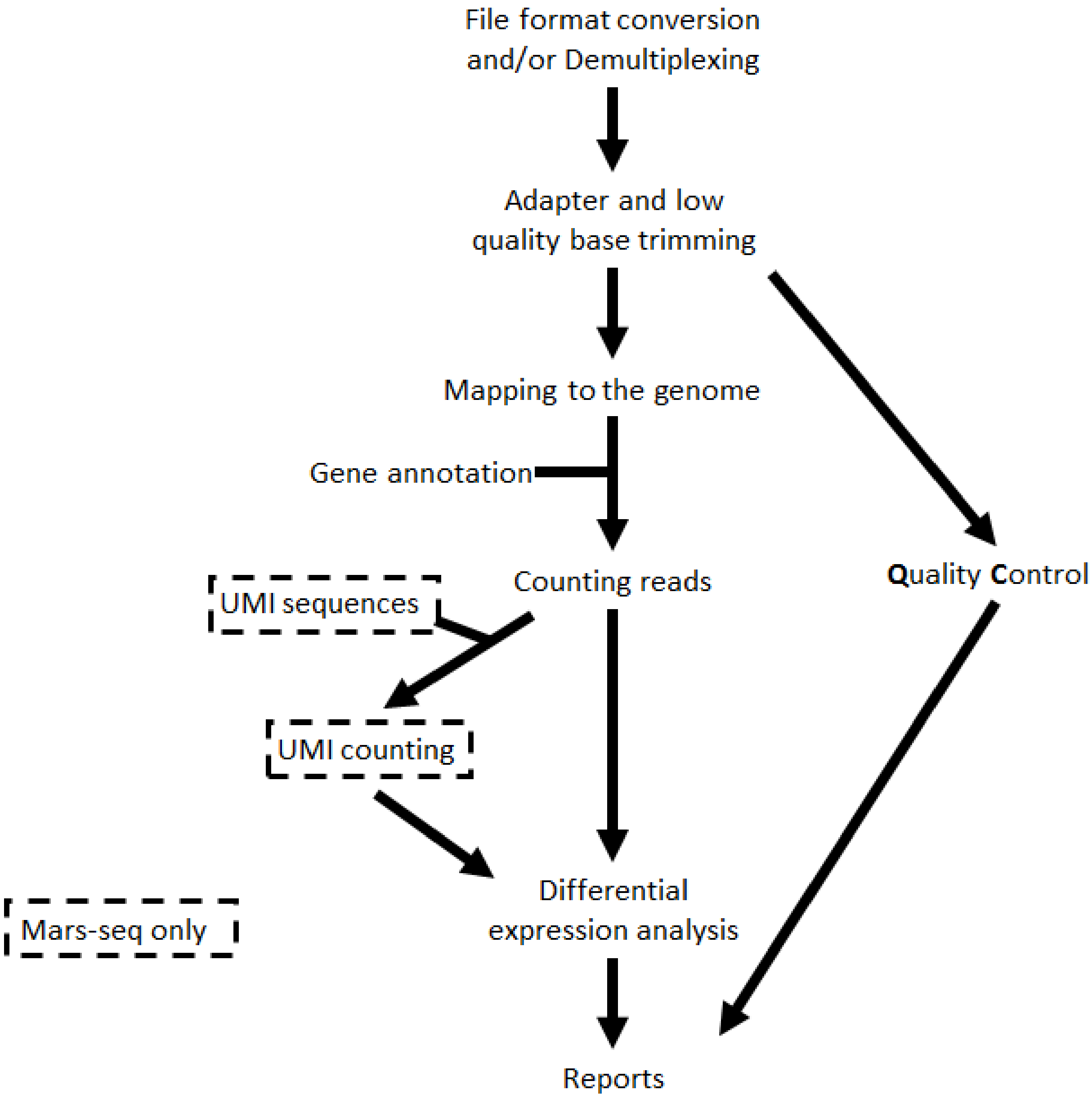
Analysis pipeline steps and reports

2.1 Analysis pipeline steps

1. Trims adapter sequences
2. Runs FastQC on the trimmed sequences for quality control of the samples, in parallel with the steps that follow
3. Maps reads to the selected reference genome
4. Adds UMI and gene information to the reads
5. Quantifies gene expression by counting reads
6. Counts UMI's for cases of PCR bias
7. Detects Differentially Expressed (DE) genes for a model with a single factor

Steps 3 and 5 are performed only for Mars-Seq

Step 6 is performed only if DESeq2 is selected


File format conversion and/or Demultiplexing
Adapter and low quality base trimming
Mapping to the genome
Gene annotation
Counting reads
Quality Control
UMI sequences
UMI counting
Differential expression analysis
Mars-seq only
Reports

2.2 Pipeline report

Upon completion of the analysis, you will be sent an email with links to the results report.

The report includes several sections:

1. Sequencing and Mapping QC
    a. **Figure 1** - Plots the average quality of each base across all reads. Qualities of 30 (predicted error rate 1:1000) and above are good
    b. **Figure 2** - Histogram showing the number of reads for each sample in the raw data

   c. **Figure 3** - Histogram showing the percentage of reads discarded after trimming the adapters (after removing adapters, short, polyA/T and low quality reads are discarded by the pipeline)
   d. **Figure 4** - Histogram with the number of reads for each sample in each step of the pipeline
   e. **Figure 5** - Plots sequence coverage on and near gene regions (Not available in current version)
   f. **Figure 6 -**
      i. Histogram showing the percentage of mapped reads (both uniquely and not uniquely) per sample
      ii. Histogram showing the percentage of the uniquely mapped reads that mapped to genes (included genes must have at least 5 reads)

2. **Exploratory Analysis**
   a. **Figure 7** - Heatmap plotting the highly-expressed genes (above 5% of total expression). For example, the expression of gene RN45S in sample SRR3112243 constitutes 15% of the total expression
   b. **Figure 8** - Heatmap of Pearson correlation between samples according to gene expression values
   c. **Figure 9** - Clustering dendogram of the samples according to gene expression
   d. **Figure 10 - PCA analysis**
      i. Histogram of % explained variability for each PC component
      ii. PCA plot of PC1 vs PC2
      iii. PCA plot of PC1 vs PC3
3. Differential Expression Analysis (this section exists only if you run the DESeq2 analysis) - a table with the number of differentially expressed genes (DE) in each category (up/down) for the different contrasts. In addition, links for p-value distribution, volcano plots and heatmaps, as well as a table of the DE genes with dot plots of their expression values are also provided
4. Bioinformatics Pipeline Methods - description of pipeline methods
5. Links to additional results - links for downloading tables with raw, normalized counts, log normalized values (rld), and statistical data of contrasts. In cases of models with batches, “combat” values were calculated (instead of rld) using the “sva” package, providing batch corrected normalized log2 count values.

2.3 Annotation file

For counts of the reads per gene, we use annotation files (gtf format) from “Ensembl” or “GENCODE”. In MARS-seq analysis, we extend the 3’ UTR exon away from the transcript on the DNA and extend or cut the 3” UTR exon towards the 5’ direction on the mRNA.

2.4 Examples of reports

RNA-Seq example

Mars-seq example

Note: This example analysis demonstrates a good starting point, and not necessarily an end result.

CHAPTER 3

Installation

Support: refael.kohen@weizmann.ac.il

3.1 Requirements

The application can be installed on a Linux server that supports cluster commands like qsub (pbs cluster), or bsub (lsf cluster).

The host server and/or each compute node in the relevent queue(s) requires ~40G of RAM memory.

The server requires the following:

- docker version >= 17
- miniconda version 2

You must set up a user (not root, and referred to below as "USER") with write permission in the miniconda environment and in the HOST_MOUNT folder (see below), that can run cluster commands and docker.

Its user id must be above 103; USER can belong to effective group with group id above 108.

USER requires a .ssh folder in its home directory. Create it with the command: ssh-keygen -t rsa

The "USER" should then do the following:

3.2 Create a directory for UTAP software and its output

Note: Since user output data will be written in this folder, please verify that you have sufficient disk space - approximately 400G per analysis.

```
HOST_MOUNT=<the relevant path>
mkdir $HOST_MOUNT
cd $HOST_MOUNT
```

3.2.1 Download the metadata

The meta-data folder contains the genomes and annotation files. You can download it using your browser from our google-drive or via ftp as noted below, and then unpack it in the $HOST_MOUNT folder.

```
#Download the zipped folder into $HOST_MOUNT folder:
wget http://utap-demo.weizmann.ac.il/reports/utap-meta-data-v1.0.7.tar.gz -P $HOST_
↪MOUNT

cd $HOST_MOUNT
tar -xzvf utap-meta-data-v1.0.7.tar.gz
```

3.2.2 Create conda environments

If you install the system on local server and don't use a compute cluster for running, skip this section. The environment requires ~29G of disk.

```
conda create -y --name utap r-essentials r-base=3.3.2
conda activate utap
# Install packages in the utap environment:
# Run the script in this file in your shell:
export CONDA_DIR=YOUR_CONDA_DIR # For example: CONDA_DIR=/home/user/miniconda2
$HOST_MOUNT/utap-meta-data/installation/install-conda-packages-transcriptome.sh >&
↪$HOST_MOUNT/utap-meta-data/installation/conda-install-transcriptome.stdout
conda deactivate

conda create -y --name utap-chromatin
conda activate utap-chromatin
export CONDA_DIR=YOUR_CONDA_DIR # For example: CONDA_DIR=/home/user/miniconda2
$HOST_MOUNT/utap-meta-data/installation/install-conda-packages-chromatin.sh >& $HOST_
↪MOUNT/utap-meta-data/installation/conda-install-chromatin.stdout
conda deactivate

conda create -y -n utap-py35 python=3.5 anaconda
conda activate utap-py35
conda install -y -c bioconda snakemake==3.13.3
conda deactivate
```

3.3 Create genomes

Extract the genomes to fasta format and create a Star index of the genomes (requires ~200GB of disk during the building process, reduced to ~135GB once the build completes and temporary files are removed):

3.3.1 Extract genome files

```
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/hg19.2bit $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/hg19/gemone_
↪hg19.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/hg38.2bit $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/hg38/gemone_
↪hg38.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/mm10.2bit $HOST_MOUNT/utap-meta-data/genomes/Mus_musculus/UCSC/mm10/gemone_
↪mm10.fa
```

(continues on next page)

(continued from previous page)

```
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/danRer10.2bit $HOST_MOUNT/utap-meta-data/genomes/Danio_rerio/UCSC/danRer10/
↪gemone_danRer10.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/tair11-araport.2bit $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/
↪ARAPORT/tair11/gemone_tair11-araport.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/tair10.2bit $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/NCBI/
↪tair10/gemone_tair10.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/twoBitToFa $HOST_MOUNT/utap-meta-data/2bit_
↪genomes/sl3.2bit $HOST_MOUNT/utap-meta-data/genomes/Solanum_lycopersicum/SGN/sl3/
↪gemone_sl3.fa
```

3.3.2 Build the STAR index

The commands listed below take ~1 hour per genome. They run on 30 threads (you can change that number by modifying the –runTreadN parameter), and consume RAM memory as follows:

- hg19: 29,918 MB
- hg38: 30,574 MB
- mm10: 27,532 MB
- danRer10: 23,523 MB
- tair11: 4,301 MB
- tair10: 4,282 MB
- sl3: 17,663 MB

```
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/
↪hg19/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/
↪UCSC/hg19/gemone_hg19.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/
↪hg38/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/
↪UCSC/hg38/gemone_hg38.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Mus_musculus/UCSC/
↪mm10/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/Mus_musculus/
↪UCSC/mm10/gemone_mm10.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Danio_rerio/UCSC/
↪danRer10/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/Danio_
↪rerio/UCSC/danRer10/gemone_danRer10.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/
↪ARAPORT/tair11/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/
↪Arabidopsis_thaliana/ARAPORT/tair11/gemone_tair11-araport.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/
↪NCBI/tair10/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/
↪Arabidopsis_thaliana/NCBI/tair10/gemone_tair10.fa
$HOST_MOUNT/utap-meta-data/softwares/bin/STAR --runThreadN 30 --runMode␣
↪genomeGenerate --genomeDir $HOST_MOUNT/utap-meta-data/genomes/Solanum_lycopersicum/
↪SGN/sl3/STAR_index/ --genomeFastaFiles $HOST_MOUNT/utap-meta-data/genomes/Solanum_
↪lycopersicum/SGN/sl3/gemone_sl3.fa
```

(continues on next page)

(continued from previous page)

3.3.3 Build the BOWTIE2 index

The commands listed below take ~1 hour per genome and consume ~15g RAM memory.

```
CONDA=/your/path/to/miniconda2/folder
$CONDA/envs/utap-chromatin/bin/bowtie2-build $HOST_MOUNT/utap-meta-data/genomes/Homo_
↪sapiens/UCSC/hg19/gemone_hg19.fa $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/
↪UCSC/hg19/BOWTIE2_index/hg19
$CONDA/envs/utap-chromatin/bin/bowtie2-build $HOST_MOUNT/utap-meta-data/genomes/Homo_
↪sapiens/UCSC/hg38/gemone_hg38.fa $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/
↪UCSC/hg38/BOWTIE2_index/hg38
$CONDA/envs/utap-chromatin/bin/bowtie2-build $HOST_MOUNT/utap-meta-data/genomes/Mus_
↪musculus/UCSC/mm10/gemone_mm10.fa $HOST_MOUNT/utap-meta-data/genomes/Mus_musculus/
↪UCSC/mm10/BOWTIE2_index/mm10
```

After extracting the fasta files and building the index, you can delete the fasta and .2bit files:

```
rm $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/hg19/gemone_hg19.fa
rm $HOST_MOUNT/utap-meta-data/genomes/Homo_sapiens/UCSC/hg38/gemone_hg38.fa
rm $HOST_MOUNT/utap-meta-data/genomes/Mus_musculus/UCSC/mm10/gemone_mm10.fa
rm $HOST_MOUNT/utap-meta-data/genomes/Danio_rerio/UCSC/danRer10/gemone_danRer10.fa
rm $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/ARAPORT/tair11/gemone_
↪tair11-araport.fa
rm $HOST_MOUNT/utap-meta-data/genomes/Arabidopsis_thaliana/NCBI/tair10/gemone_tair10.
↪fa
rm $HOST_MOUNT/utap-meta-data/genomes/Solanum_lycopersicum/SGN/sl3/gemone_sl3.fa
rm $HOST_MOUNT/utap-meta-data/2bit_genomes/*
```

3.4 Run UTAP

3.4.1 Pull UTAP image from the public repository

```
docker pull refaelkohen/utap
```

For running UTAP on a local server, execute the following command (all parameters all mandatory), which will create a Docker container called "utap".

```
$HOST_MOUNT/utap-meta-data/installation/utap-install.sh -a DNS_HOST -b HOST_MOUNT -c␣
↪REPLY_EMAIL -d MAIL_SERVER -e HOST_APACHE_PORT -g ADMIN_PASS -h USER -i INSTITUTE_
↪NAME -j DB_PATH -k MAX_UPLOAD_SIZE -m MAX_CORES -n local
```

For running UTAP on a compute cluster run the command:

```
$HOST_MOUNT/utap-meta-data/installation/utap-install.sh -a DNS_HOST -b HOST_MOUNT -c␣
↪REPLY_EMAIL -d MAIL_SERVER -e HOST_APACHE_PORT -g ADMIN_PASS -h USER -i INSTITUTE_
↪NAME -j DB_PATH -k MAX_UPLOAD_SIZE -m MAX_CORES -n CLUSTER_TYPE -o CLUSTER_QUEUE -p␣
↪RESOURCES_PARAMS -q CONDA
```

After the run, you can access the application using the address: http://DNS_HOST:HOST_APACHE_PORT (according to your choices for values of these parameters)

You can run the command in the background and close the terminal.

3.4.2 Parameters

-a DNS_HOST	DNS address of the host server. **For example:** http://servername.ac.il or servername.ac.il
-b HOST_MOUNT	Mount point from the docker on the host (full path of the folder). This is the folder that contains the utap-meta-data folder. All input and output data for all of the users will be written into this folder.
-c REPLY_EMAIL	Support email for users. Users can reply to this email.
-d MAIL_SERVER	Domain name of the mail server **For example:** mg.weizmann.ac.il
-e HOST_APACHE_PORT	Any available port on the host server for the Docker Apache. **For example:** 8081
-f HOST_SSH_PORT	(Optional) Any available port on the host server for the Docker ssh server. **For example:** 2222
-g ADMIN_PASS	Password of an admin in the djnago database (the string can contain only A-Za-z0-9 characters without whitespaces).
-h USER	user in host server that has permission to run cluster commands and write into the $HOST_MOUNT folder (cannot be root).
-i INSTITUTE_NAME	Your institute name or lab (the string can contain only A-Za-z0-9 characters without whitespaces).
-j DB_PATH	Full path to the folder where the DB will be located. $USER needs to have write permission for this folder. The "DB_PATH" should not be under a mounted folder. The DB is very small, so it is will not create disk space problems. **For example:** mkdir /utap-db; chown -R $USER/utap-db;
-k MAX_UPLOAD_SIZE	Maximum file/folder size that a user can upload at once (Megabytes). **For example:** 314572800 (i.e. 300*1024*1024 = 314572800Mb = 300Gb)
-l PROXY_RUL	(Optional) url of utap if you using with proxy. default: DNS_HOST:HOST_APACHE_PORT
-m MAX_CORES	Maximum cores in the host computer or in each node of the cluster
-n CLUSTER_TYPE	"local". The commands of the UTAP application will be run on the local server; there is no need to supply the parameters: CLUSTER_QUEUE, CONDA, or AUTH_KEYS_FILE.

3.4.3 Additional parameters for installing on a cluster

-n CLUSTER_TYPE Type of the cluster.

For example: lsf or pbs.

The commands will be sent to the cluster. Currently, UTAP supports LSF or PBS cluters.

-o CLUSTER_QUEUE Queue name in the cluster. $USER must have permissions to run on this queue.

-p RESOURCES_PARAMS Parameters for your cluster resources.

If the memory is per cpu, set mem=resources.mem_mb_per_thread,

If the memory is for all cpus of the job together, set mem=resources.mem_mb_total

Examples (be sure to use quotes exactly as shown below):

'-l select=1:ncpus={threads}:mem={resources.mem_mb_total}mb'

'-l mem={resources.mem_mb_total}mb,nodes=1:ppn={threads}'

'-n {threads} -R "rusage[mem={resources.mem_mb_per_thread}]" -R "span[hosts=1]"'

-q CONDA Full path to root folder of miniconda.

For example: /miniconda2

Important:

A file called db.sqlite3 will be created within $DB_PATH folder.

The db.sqlite3 file is the database of the application; it contains user details, and links to results in the $HOST_MOUNT folder.

The $HOST_MOUNT folder contains all of the data for all of the users (input and output files).

The db.sqlite3 database and $HOST_MOUNT folder are located on the disk of the host server (out of the docker container).

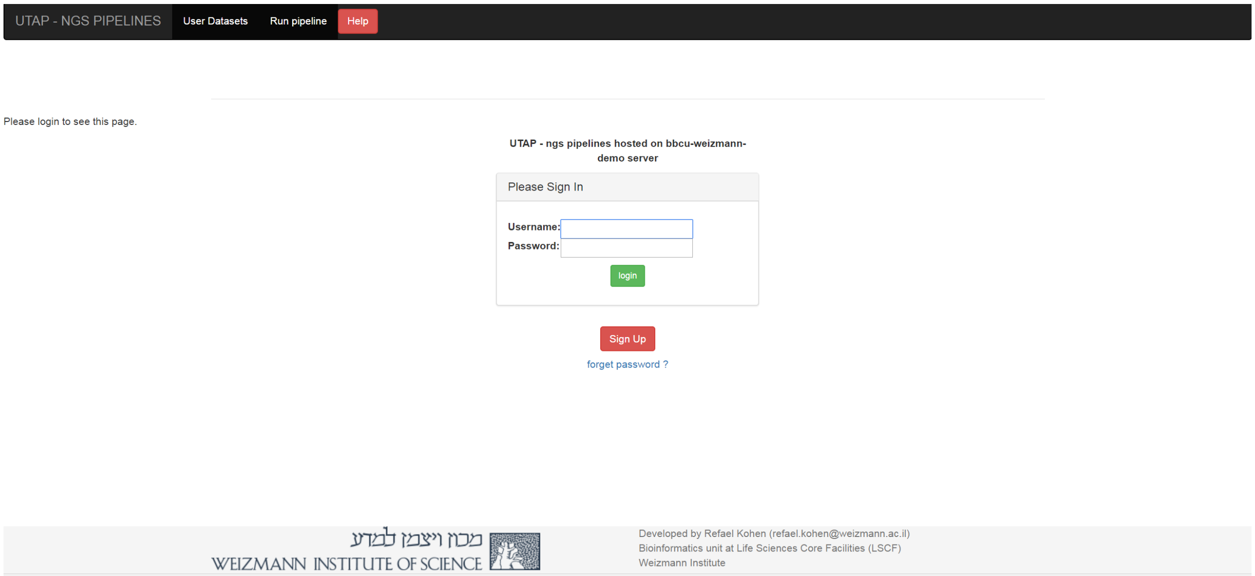
When you stop/delete the "utap" container, the database and $HOST_MOUNT folder are not deleted.

If there is a need to temporarily delete the docker, keep the database ("db.sqlite3") and the same $HOST_MOUNT folder. When you rerun the docker via the utap-install.sh script, you can use the existing database ("db.sqlite3") and $HOST_MOUNT folder.

CHAPTER 4

User guide

4.1 Registration to the system

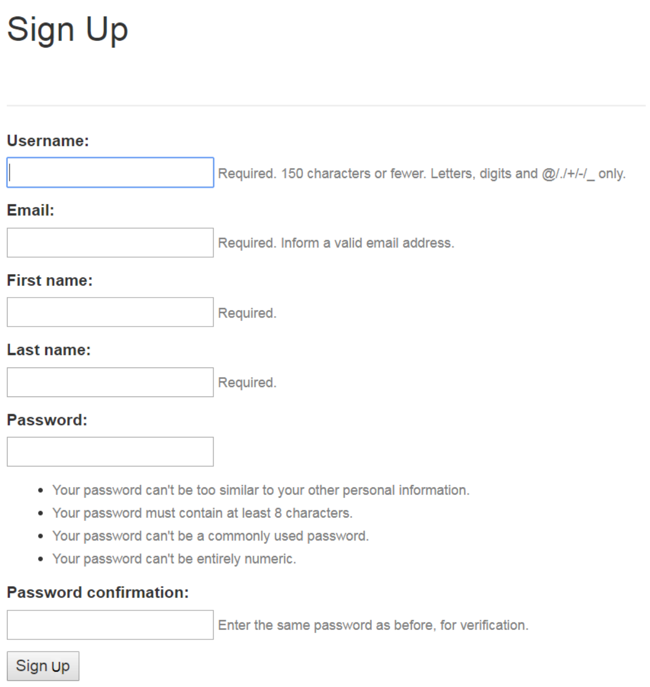

UTAP - NGS PIPELINES
User Datasets
Run pipeline
Help
Please Sign In
Username:
Password:
login
Sign Up
forget password ?
WEIZMANN INSTITUTE OF SCIENCE
מכון ויצמן למדע

Please login to see this page.

UTAP - ngs pipelines hosted on bbcu-weizmann-demo server

Developed by Refael Kohen (refael.kohen@weizmann.ac.il)
Bioinformatics unit at Life Sciences Core Facilities (LSCF)
Weizmann Institute

Click on the signup button and fill out the form:



Sign Up

Username:
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email:
Required. Inform a valid email address.

First name:
Required.

Last name:
Required.

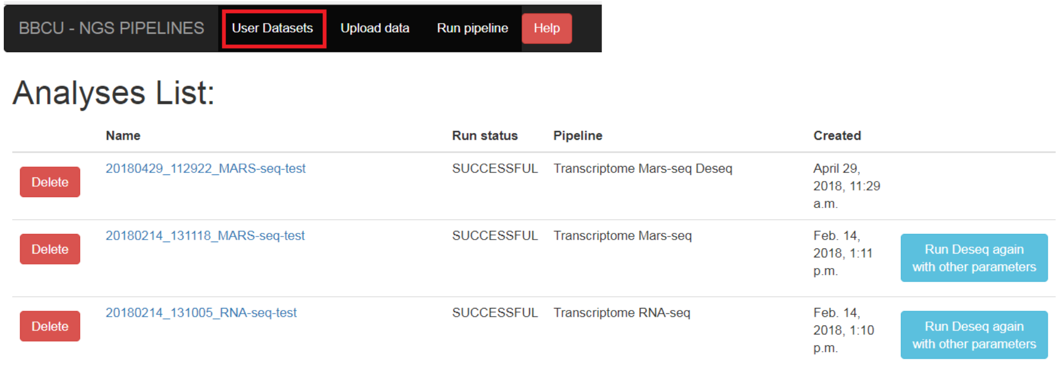
Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:
Enter the same password as before, for verification.

4.2 User datasets

The "User datasets" screen contains the list of the user's analyses. You can see the status of the run (RUNNING/SUCCESSFOL/FAILS). You need to refresh the page to see if the status has changed (you also will get email in the end of the run).



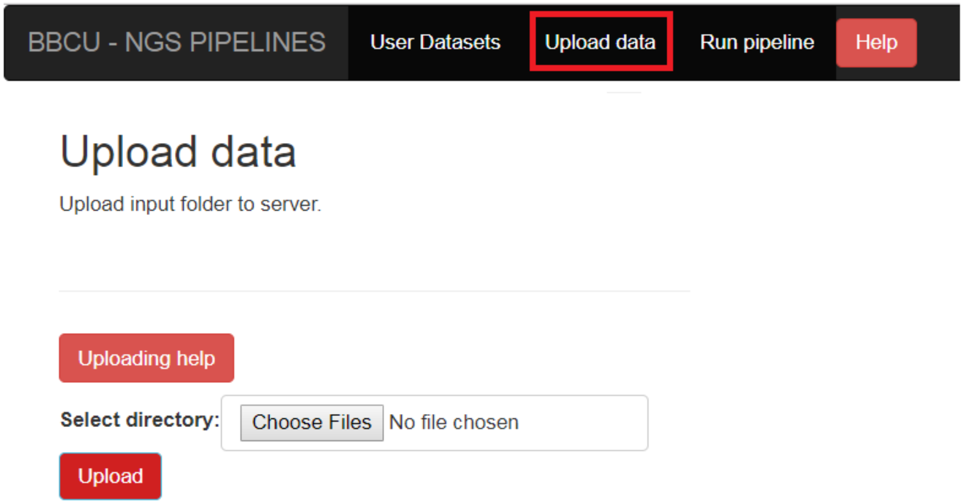
Analyses List:

	Name	Run status	Pipeline	Created	
Delete	20180429_112922_MARS-seq-test	SUCCESSFUL	Transcriptome Mars-seq Deseq	April 29, 2018, 11:29 a.m.	
Delete	20180214_131118_MARS-seq-test	SUCCESSFUL	Transcriptome Mars-seq	Feb. 14, 2018, 1:11 p.m.	Run Deseq again with other parameters
Delete	20180214_131005_RNA-seq-test	SUCCESSFUL	Transcriptome RNA-seq	Feb. 14, 2018, 1:10 p.m.	Run Deseq again with other parameters

4.3 Import Input data

In order to run the transcriptome analysis pipeline, fastq sequence files need to be located on the server.

Click on the "Upload data" button on the navigation bar, and select the folder of fastq files.



Upload data

Upload input folder to server.

Select directory: No file chosen

4.3.1 Instructions for uploading data to the server

Before running the pipeline, first upload the input file to the server as follows:

Press on the "choose files" button, and select the root folder of the files

Uploading folder of fastq files

- Fastq files must be organized, within the selected folder (root folder), into subfolders as shown below. Subfolders names are derived from sample names.
- Each subfolder contains the relevant fastq file, which can be compressed into the "gz" format.
- Fastq file names must end with "_R1.fastq(.gz)" or "_R1.fq(.gz)" for single-read data. The "R" prefix donates the read number.
- In the case of paired-end data (required for Mars-Seq), corresponding files must exist, with suffix "_R2.fastq(.gz)" instead of "_R1.fastq(.gz)".

For example:

- **root folder**
  - **sample1**
    - sample1_R1.fastq
    - sample1_R2.fastq (must exist in Mars-seq and in paired-end)
  - **sample2**
    - sample2_R1.fastq
    - sample2_R2.fastq (must exist in Mars-seq and in paired-end)

The pipeline also supports the old convention of the fastq file names *_L00*_R1_0*.fastq. The letter "L" denotes the lane number, "R" denotes the read number, and the numbers 001,002 etc denote serial number of the file for each lane and read

For example:

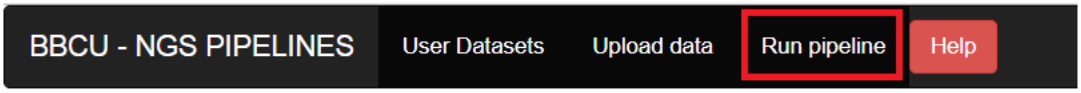
- **root folder**
  - **sample1**
    - sample1_S0_L001_R1_001.fastq
    - sample1_S0_L001_R1_002.fastq
    - sample1_S0_L002_R1_001.fastq
    - sample1_S0_L002_R1_002.fastq
    - sample1_S0_L001_R2_001.fastq
    - sample1_S0_L001_R2_002.fastq
    - sample1_S0_L002_R2_001.fastq
    - sample1_S0_L002_R2_002.fastq
  - **sample2**
    - …

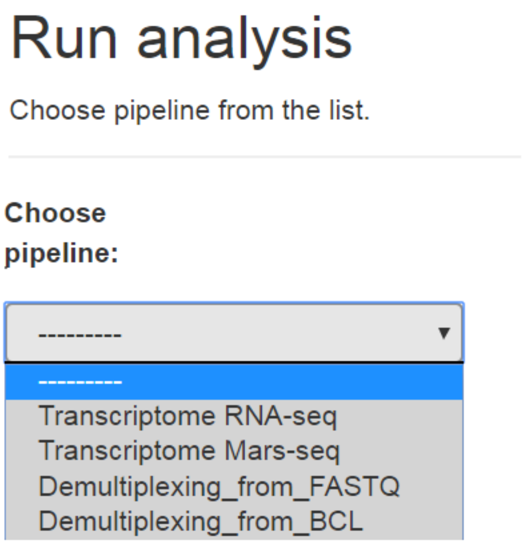
Uploading input for the demultiplexing pipeline

- For the pipeline of demultiplexing from BCL files: upload the original bcl folder. The original folder name should adhere to the template: <date>_<field2>.<field3>_field4>, e.g. 180514_NB551168_0123_AHTHHKBGX5.
- For the pipeline of demultiplexing from FASTQ files: upload the fastq files. Note: the pipeline gets one file per read as input (i.e. one file for *R1*, *R2*, *I1* etc.).

4.4 Run analysis

After importing you data (or if you have old data on the server that was imported in the past), you can run the pipeline by selecting the "Run pipeline" option


BBCU - NGS PIPELINES
User Datasets
Upload data
Run pipeline
Help


Run analysis

Choose pipeline from the list.

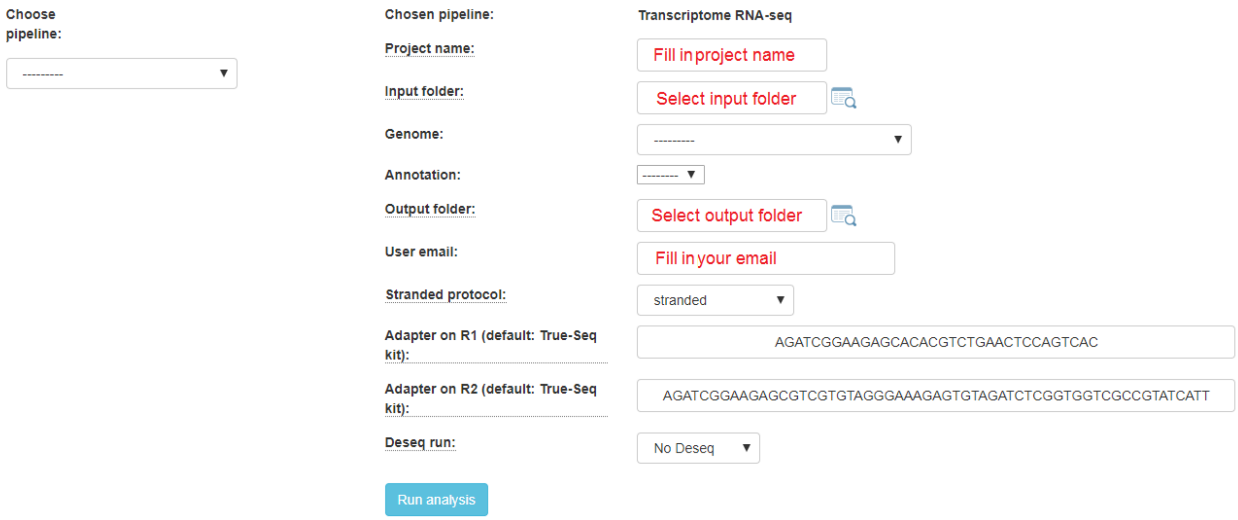
Choose pipeline:

Transcriptome RNA-seq
Transcriptome Mars-seq
Demultiplexing_from_FASTQ
Demultiplexing_from_BCL

4.4.1 Run RNA-seq or MARS-seq pipeline

RNA-seq Analysis Setup

If your protocol is RNA-seq, you will get this screen:


Choose pipeline: ---------

Chosen pipeline: Transcriptome RNA-seq

Project name: Fill in project name

Input folder: Select input folder

Genome: ---------

Annotation: --------

Output folder: Select output folder

User email: Fill in your email

Stranded protocol: stranded

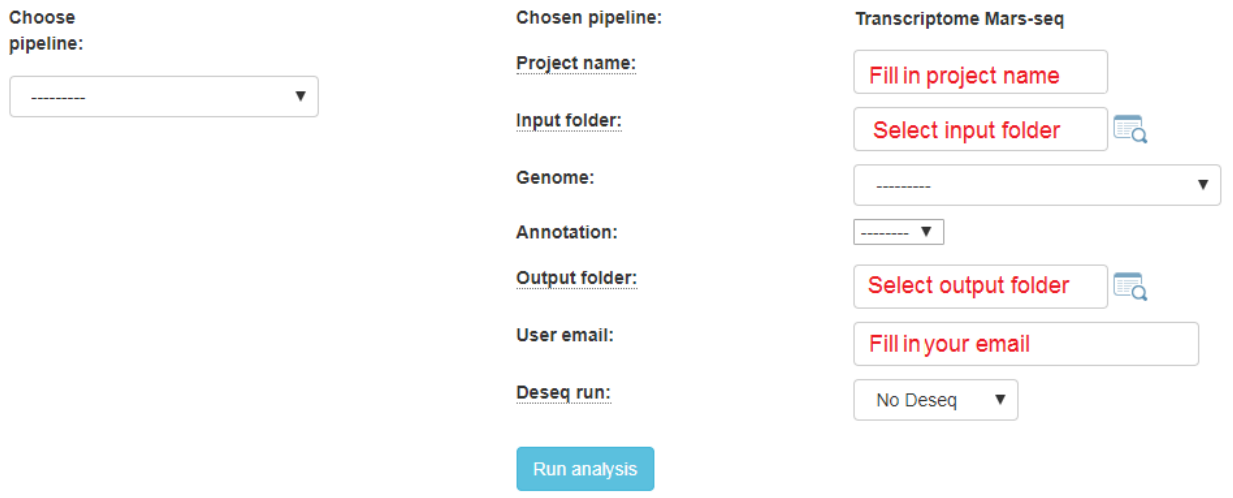
Adapter on R1 (default: True-Seq kit): AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC

Adapter on R2 (default: True-Seq kit): AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT

Deseq run: No Deseq

Run analysis

If your protocol is MARS-seq, you will get this screen:



Choose pipeline: ---------

Chosen pipeline: Transcriptome Mars-seq

Project name: Fill in project name

Input folder: Select input folder

Genome: ---------

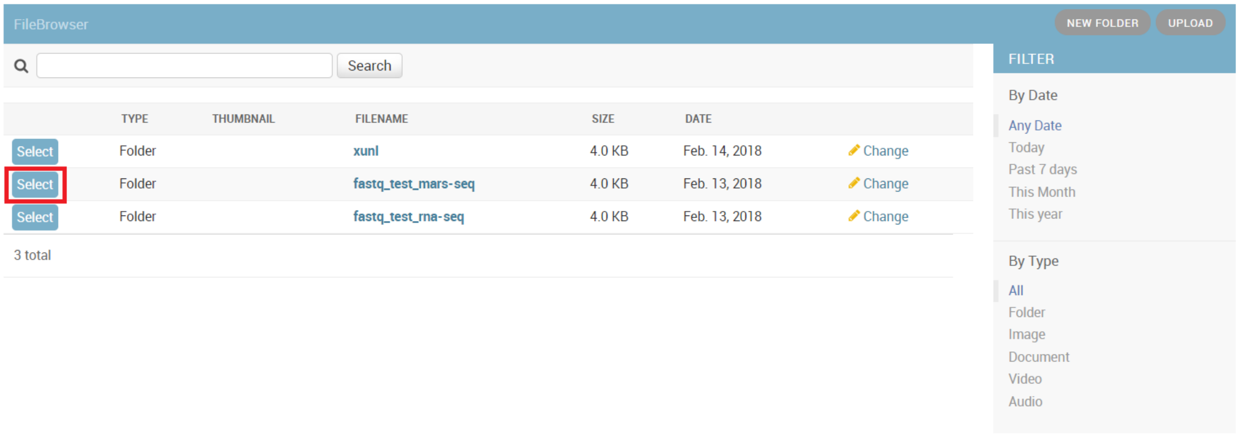
Annotation: --------

Output folder: Select output folder

User email: Fill in your email

Deseq run: No Deseq

In the input folder field, Browse within your directory structure and Select the **root folder** for analysis. Note that if you wish to go up one level (or more) click on the desired folder level using the path at the top of the window.



	TYPE	THUMBNAIL	FILENAME	SIZE	DATE	
Select	Folder		xunl	4.0 KB	Feb. 14, 2018	Change
Select	Folder		fastq_test_mars-seq	4.0 KB	Feb. 13, 2018	Change
Select	Folder		fastq_test_rna-seq	4.0 KB	Feb. 13, 2018	Change

3 total

Input folder names must conform to the correct format as previously described. If there is a problem with the folder you selected, first resolve the error and then retry, selecting the updated folder.

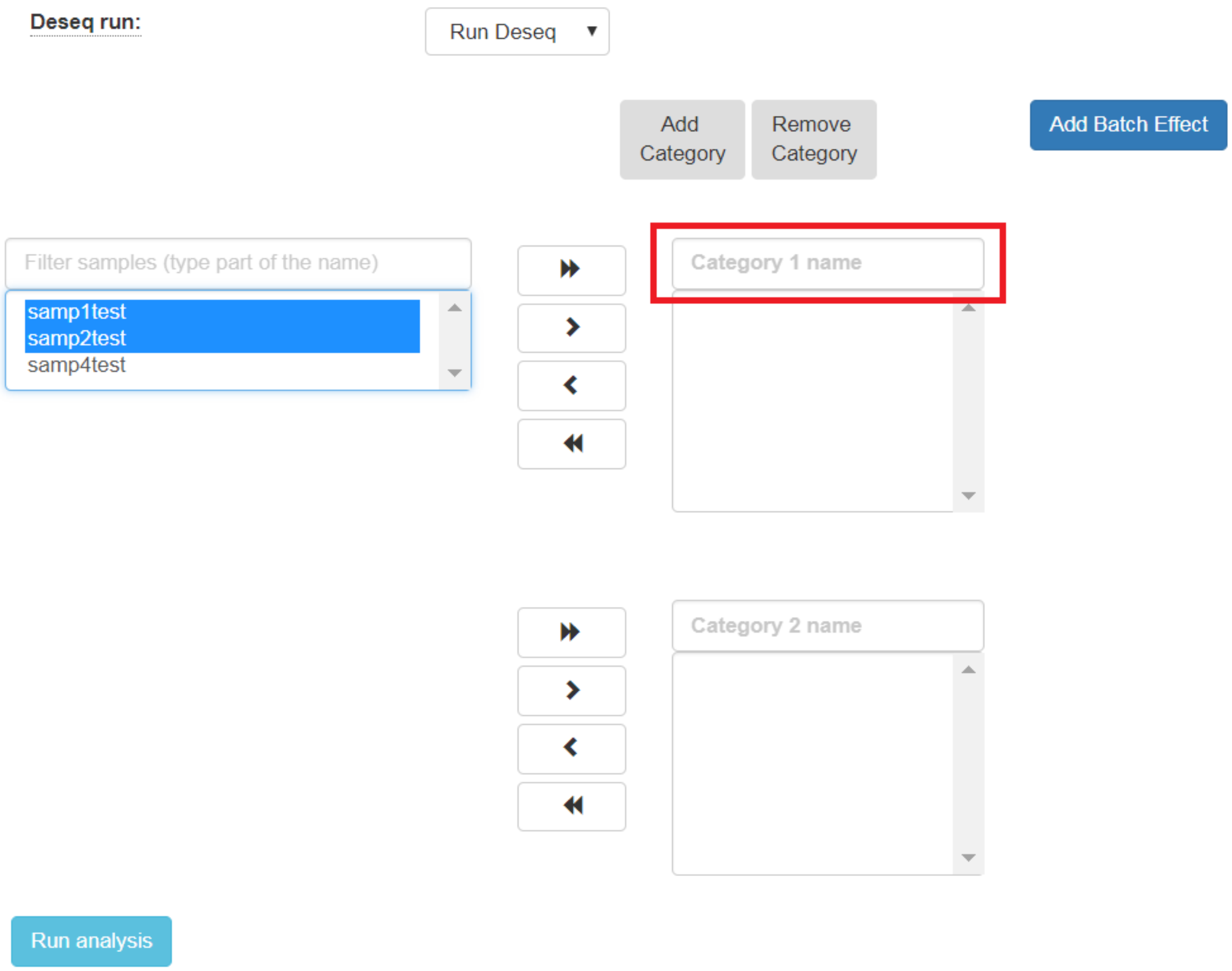
If you wish the output folder to be different from the one automatically filled in (based on the selected input folder), just select the desired output folder.

Fill in the project name, then select the genome and annotation.

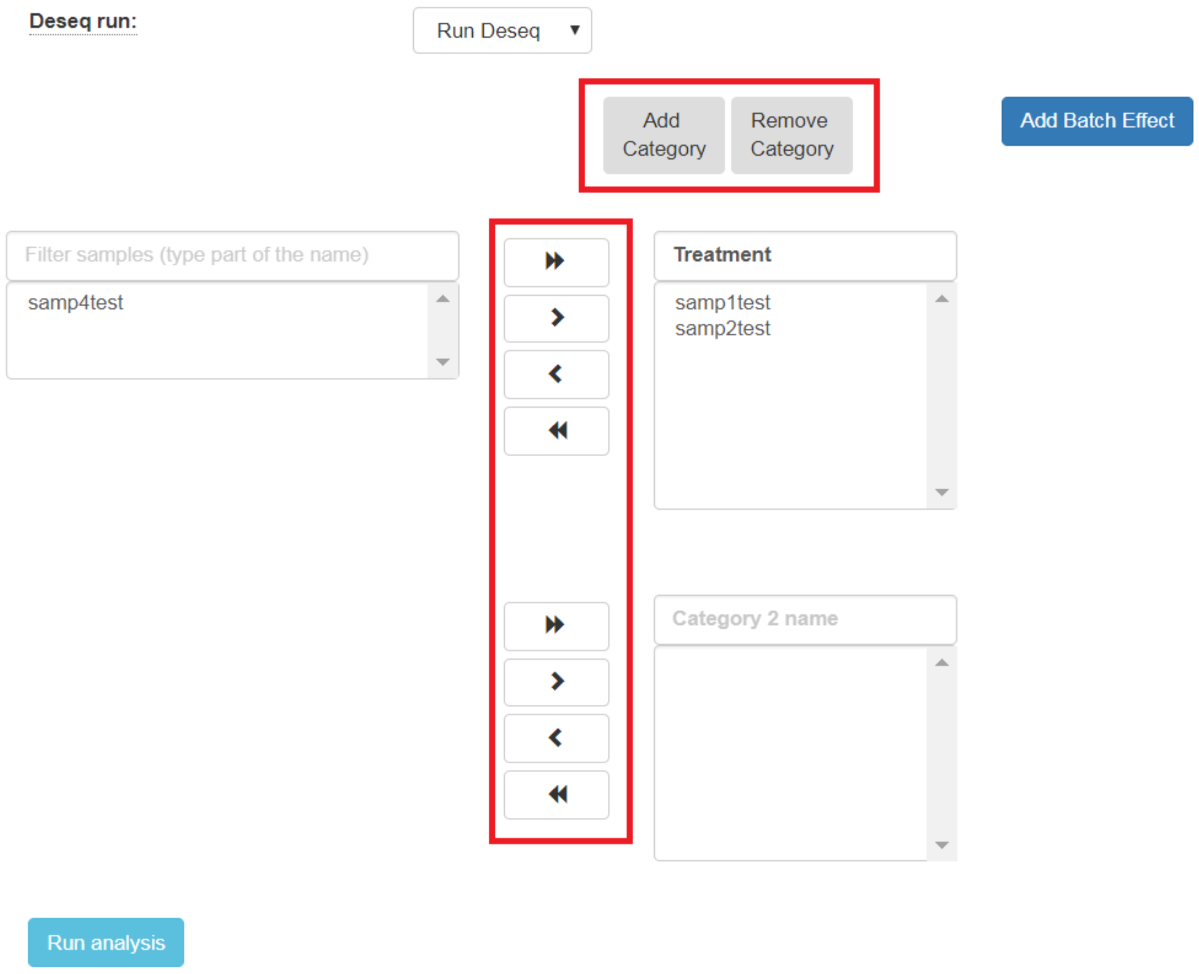
For RNA-seq protocols, choose whether your protocol is stranded (sequenced reads save the original strand of RNA fragments) or non-stranded.

Define the type of your adapters for each read (R1 and R2). These adapters will be removed from the reads by the pipeline. You can leave the default adapters if you use True-seq protocol P5 and P7 adapters.

To identify what's differentially expressed by using the DESeq2 package, select the Run Deseq option. By default, two categories must be created. Fill in the category names for each of the 2 categories shown. To define more categories, click on the Add Categories button to enable entering their details.

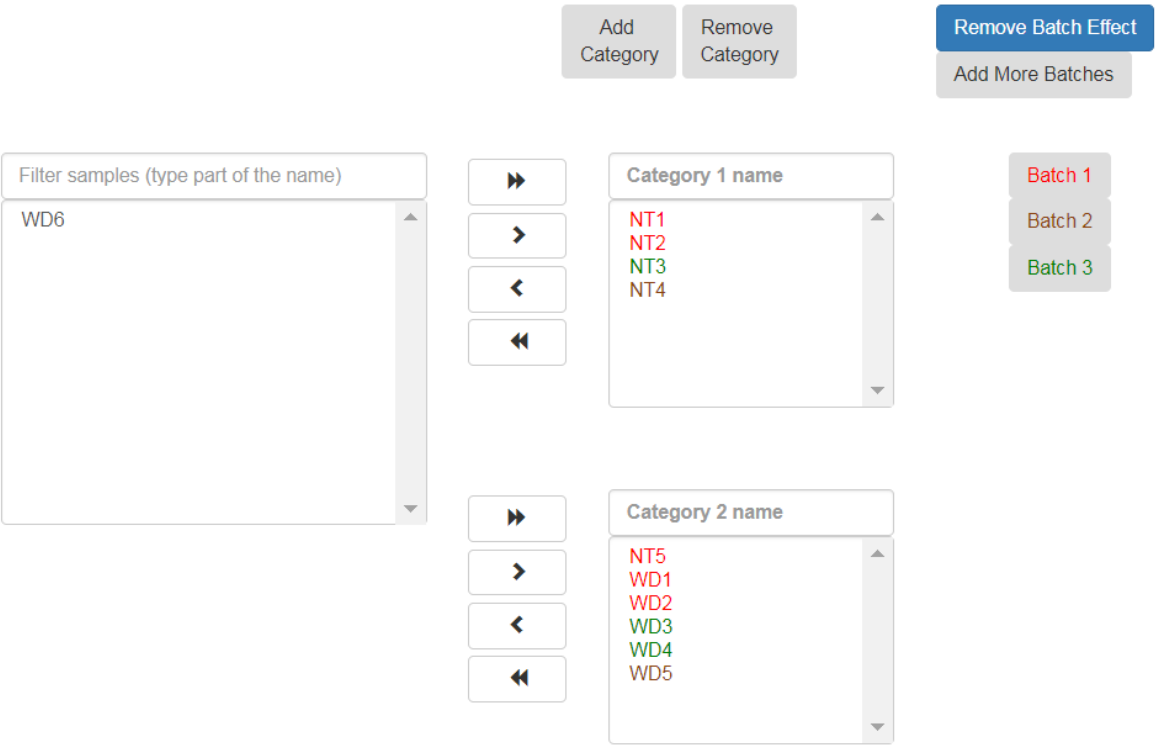

Deseq run:
Run Deseq
Add Category
Remove Category
Add Batch Effect
Filter samples (type part of the name)
samp1test
samp2test
samp4test
Category 1 name
Category 2 name
Run analysis

Choose the samples by first selecting them, and then using the arrows to move them to the appropriate categories. You may also add additional categories.


Deseq run:
Run Deseq
Add Category
Remove Category
Add Batch Effect
Filter samples (type part of the name)
samp4test
Treatment
samp1test
samp2test
Category 2 name
Run analysis

The order of what's being compared will be determined by the specification order of the categories. For example, DESeq2's output will show a "Treatment" vs "Control" comparison when "Treatment" is defined to be the first category, and "Control" the second.

If the samples were prepared in different batches, one can annotate them as follows: After moving the samples into category boxes, click on the "Add Batch Effect" button, select the samples from the category boxes that belong to a particular batch, and click on the "Batch 1" button. Repeat the operation for the other batches. Be sure that the batch effect is designed correctly - seeDESeq2 documentation here.


Add Category
Remove Category
Remove Batch Effect
Add More Batches
Filter samples (type part of the name)
WD6
Category 1 name
NT1
NT2
NT3
NT4
Batch 1
Batch 2
Batch 3
Category 2 name
NT5
WD1
WD2
WD3
WD4
WD5

All of the steps of the pipeline (mapping, counts etc.) will be run on all of the samples, with the exception of Deseq which will be run only on samples with categories.

Finally, click on the "Run analysis" button.

At the end of the run, an email will be sent reporting analysis completion.

Using the pipeline efficiently

If you want re-run only the Deseq step several times on the same input folder (with other comparisons/batches), after completion of the initial analysis you will see (on the "user dataset" screen) a new button called "run again with other parameters". Clicking on this button will re-run only the Deseq step.

Thus, the analysis will re-run only the short Dseq step (which takes a few minutes) and not re-run all of the time-consuming steps of the complete pipeline.

4.4.2 Run Demultiplexing pipeline

There are 2 pipelines for demultiplexing; the first accepts BCL files as input, the second fastq files.

Demultiplexing from BCL files

Upload BCL files to the server according to the following instructions:

All original BCL file folders must be built according to Next-seq (or Hi-seq) machine requirements. Folder names should adhere to the template <date>_<machine name>_<run number>_<flowcell id>, e.g. "170802_NB501465_0140_AH3W3KBGX3".

The pipeline converts bcl files to fastq files, and demultiplexes fastq files according to MAR-seq or True-seq (or semi-True-seq) protocols.

Demultiplexing from fastq files

The pipeline demultiplexes fastq files according to MAR-seq or True-seq (or semi- True-seq) protocols.

Upload fastq files to the server according to the following instructions:

Note that the pipeline get as input one file per read (i.e. one file for each of *R1*, *R2*, *I1* etc.). Choose the root folder of the fastq files from the list.

If the sequencing is single read, choose the file with *R1* in its file name for read 1, the file with either *R2* or *I1* for index read, and leave the read 2 field empty.

If the sequencing is paired end, choose the file with *R1* in its file name for read 1, the file with *R2* for read 2, and the file with *I1* for index read. If no file name includes *I1*, choose one with *R2* for index read, and one with *R3* for read 2.

4.5 Customization

We chose the various pipeline parameters based on our rich experience in transcriptome analysis. This works very well for users who are not deeply familiar with bioinformatics software, and who prefer to quickly benefit from these choices without having to delve into the pipeline's architecture. On the other hand, many research groups have their own particular preferences, and can achieve flexibility by making some minor adjustments to the code as follows:

1. **System-wide changes:**

   The Snakefiles of the pipelines are built using the Snakemake workflow management system (https://snakemake.readthedocs.io) , and are located at:

   $CONDA/envs/utap/lib/python2.7/site-packages/ngs-snakemake/snakefile-marseq.txt
   $CONDA/envs/utap/lib/python2.7/site-packages/ngs-snakemake/snakefile-rnaseq.txt

   (where $CONDA is the location of the miniconda - see https://utap.readthedocs.io/en/latest/rst/installation.html).

   Users can modify the above scripts by adding new rules, commands and changing parameters.

   In addition, one can customize the following R script's DESeq2 analyses and report generation:

   $CONDA/envs/utap/lib/python2.7/site-packages/ngs-snakemake/reports.Rmd

   After the code is changed, subsequent analyses using the web application will reflect these changes.

2. **Ad-hoc changes:**

   Another option is to change parameters only for a particular run.

   After running the analysis in the usual way, one can navigate to the output folder, which contains a copy of the snakefile and Rmd script. One can then change and re-run the analysis from the linux terminal by executing:

   `./snakemake_cmd_RUN_ID` (where an example of a RUN_ID is a timestamp like "20171205_145424").

CHAPTER 5

Releases

- 1.0.7 - Added installation parameter to determine maximum cores that UTAP can use.
- 1.0.6 – Added installation parameter to support a variety Portable Batch System (PBS) clusters
- 1.0.5 – Support for the ATAC-seq pipeline

CHAPTER 6

Source code

6.1 UTAP source code

https://bitbucket.org/bbcu/utap

6.2 Dependencies

Softwares

Genomes

CHAPTER 7

License

UTAP is licensed under GNU General Public License version 3. License needed for commercial use.

CHAPTER 8

Author

Refael Kohen,

refael.kohen@weizmann.ac.il

Bioinformatics unit at Life Sciences Core Facilities (LSCF)

Weizmann Institute of Science, Rehovot 76100, Israel.

CHAPTER 9

Acknowlegments

Please cite: Kohen R, Barlev J, Hornung G, Stelzer G, Feldmesser E, Kogan K, Safran M, Leshkowitz D: UTAP: User-friendly Transcriptome Analysis Pipeline. BMC Bioinformatics 2019, 20(1):154.