uORF-Tools Documentation

Release 2.0.0

Rick Gelhausen

Jul 30, 2019

Documentation

1 uORF-Tools **1**
- 1.1 Introduction . . . 1
- 1.2 Program flowchart . . . 1
- 1.3 Directory table . . . 1
- 1.4 Input/Output Files . . . 3
- 1.5 Tool Parameters . . . 4
- 1.6 Requirements . . . 4
- 1.7 Tools . . . 4
- 1.8 Input files . . . 5
- 1.9 Example-workflow . . . 7
- 1.10 Preprocessing-workflow . . . 7
- 1.11 References . . . 7

2 Example workflow **9**
- 2.1 Setup . . . 9
- 2.2 Retrieve and prepare input files . . . 9
- 2.3 Running the workflow . . . 12
- 2.4 References . . . 13

3 Preprocessing workflow **15**
- 3.1 Setup . . . 15
- 3.2 Retrieve and prepare input files . . . 15
- 3.3 Running the workflow . . . 17

Bibliography **21**

CHAPTER 1

uORF-Tools

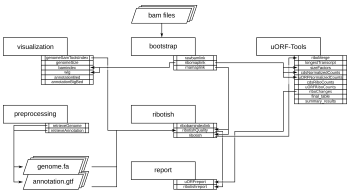
1.1 Introduction

uORF-Tools is a workflow and a collection of tools for the analysis of **Upstream Open Reading Frames** (short uORFs). The workflow is based on the workflow management system **snakemake** and handles installation of all dependencies via bioconda [GruningDSjodin+17], as well as all processings steps. The source code of uORF-Tools is open source and available under the License **GNU General Public License 3**. Installation and basic usage is described below.

Note: For a detailed step by step tutorial of our workflow on a sample dataset, please refer to our *example-workflow*.

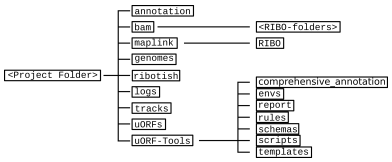
1.2 Program flowchart

The following flowchart describes the processing steps of the workflow and how they are connected:



1.3 Directory table

The output is written to a directory structure that corresponds to the workflow steps, you can decide at the beginning of the workflow whether you want to keep the intermediary files (default) or only the final result.



- **annotation:** contains the user-provided annotation file with genomic features.
  Contents: *annotation.gtf*
- **bam:** contains the input *.bam* files.
  Contents: *<method>-<conditon>-<replicate>.bam*
- **genomes:** contains the genome file, as well as an according index and sizes file.
  Contents: *genome.fa*, *genome.fa.fai*, *sizes.genome*
- **logs:** contains log files for each step of the workflow.
  Contents: *<rule>.o<jobID>*, *<methods>.log*
- **maplink:** contains soft links to the *.bam* files and an according index.
  - **RIBO:** contains soft links to the *.bam* and *.bam.bai* files for RIBO and corresponding parameter files (*.para.py*).

  Contents: *<method-condition-replicate>.bam.bai*, *RIBO/<condition-replicate>.bam.para.py*

- **ribotish:** contains the result files of ribotish.
  Contents: *<condition-replicate>-newORFs.tsv*, *<condition-replicate>-newORFs.tsv_all.txt*, *<condition-replicate>-qual.txt*, *<condition-replicate>-qual.pdf*
- **tracks:** contains *BED (.bed)*, *wig (.wig)* and *bigWig (.bw)* files for visualizing tracks in a genome browser.
  Contents: *annotation.bb*, *annotation.bed*, *annotation.bed6*, *annotation-woGenes.gtf*, *<method-condition-replicate>.bw*, *<method-condition-replicate>.wig*
- **uORFs:** contains the main output of the workflow.
  - **uORFs_regulation.tsv:** table summarizing the predicted uORFs with their regulation on the main ORF.
  - **merged_uORFs.bed:** genome browser track with predicted uORFs.

  Contents: *longest_protein_coding_transcripts.gtf*, *merged_uORFs.bed*, *merged_uORFs.csv*, *ribo_norm_CDS_reads.csv*, *ribo_norm_uORFs_reads.csv*, *ribo_raw_CDS_reads.csv*, *ribo_raw_uORFs_reads.csv*, *sfactors_lprot.csv*, *uORFs_regulation.tsv*

- **uORF-Tools:** contains the workflow tools.
  - **comprehensive_annotation:** an example annotation.
  - **envs:** conda environment files (.yaml).
  - **report:** restructuredText files for the report (.rst).
  - **rules:** the snakemake rules.
  - **schemas:** validation templates for input files
  - **scripts:** scripts used by the snakemake workflow.
  - **templates:** templates for the *config.yaml* and the *samples.tsv*.

1.4 Input/Output Files

This table contains explanations for each of the input/output files.

File name	Description
annotation.gtf	user-provided annotation file with genomic features
genome.fa	user-provided genome file containing the genome sequence
genome.fa.fai	index file of the genome file
sizes.genome	file containing the sizes of each genome sequence in the genome file
<method>-<conditon>-<replicate>.bam	user-provided alignment files (or created using the preprocessing workflow)
<method-condition-replicate>.bam.bai	index file of the alignment files
<condition-replicate>.bam.para.py	parameter file generated by RiboTISH
<methods>.log	files containing the process log for each method
<condition-replicate>-newORFs.tsv	RiboTISH output file containing newly discovered ORFs (significant only)
<condition-replicate>-newORFs.tsv_all.txt	RiboTISH output file containing newly discovered ORFs (all)
<condition-replicate>-qual.txt	RiboTISH quality control report text file
<condition-replicate>-qual.pdf	RiboTISH quality control report file with illustrations
annotation.bb	input annotation in bigbed format for genome browser visualization
annotation.bed	input annotation in bed format for genome browser visualization
annotation.bed6	input annotation in bed6 format for genome browser visualization
annotation-woGenes.gtf	input annotation filtered exclusively for gene features
<method-condition-replicate>.bw	BigWig files for visualizing data in a genome Browser
<method-condition-replicate>.wig	wig files for visualizing data in a genome Browser
uORFs_regulation.tsv	final output table including all uORFs and their mORF
merged_uORFs.bed	bed file containing potential ORFs for genome browser visualization
merged_uORFs.csv	list of potential ORFs with coordinates and mORF annotation
longest_protein_coding_transcripts.gtf	input annotation filtered for longest splice variant for each locus
ribo_raw_CDS_reads.csv	read counts for annotated ORFs
ribo_raw_uORFs_reads.csv	read counts for potential ORFs
ribo_norm_CDS_reads.csv	deseq2 normalized read counts for annotated ORFs
ribo_norm_uORFs_reads.csv	deseq2 normalized read counts for potential ORFs
sfactors_lprot.csv	deseq2 size factors for protein coding transcripts

1.4.1 uORFs_regulation.tsv

Description for the columns present in the final output file:

- transcript_id: transcript id of the main open reading frame (mORF)
- uORF_id: id of the potential Upstream open reading frame (uORF), derived from mORF id
- Ratio: list of columns, one for each sample, with the ratio of read counts for the mORF and the uORF
- Standard deviation of changes of the ratio of the relative uORF activities of treatment vs control
- binary logarithm fold change of the ratio of the relative uORF activities of treatment vs control

1.5 Tool Parameters

Special characters and versions used for the most important tools. Standard input/output parameters were omitted.

Tool	Version	Special parameters used
ribo-TISH	0.2.1	–longest (-v -p -b -g -f)
trim-galore	0.5.0	–phred33 -q 20 –length 15 –trim-n –suppress_warn –clip_R1 1 –dont_gzip
star	2.6.1b	–genomeDir genomeStarIndex –outSAMtype BAM SortedByCoordinate –outSAMattributes All –outFilterMultimapNmax 1 –alignEndsType Extend5pOfRead1
sort-merna	2.1b	-m 4096 -a –ref <dbstring> –reads –num_alignments 1 –fastx –aligned –other
fastqc	0.11.8	
im-agemag-ick	7.0.8_15	-density 150 -trim -quality 100 -flatten -sharpen 0x1.0

1.6 Requirements

In the following, we describe all the required files and tools needed to run our workflow.

1.7 Tools

1.7.1 miniconda3

As this workflow is based on the workflow management system snakemake [KosterR18], **Snakemake** will download all necessary dependencies via conda.

We strongly recommend installing miniconda3 with **python3.7**.

After downloading the **miniconda3** version suiting your linux system, execute the downloaded bash file and follow the instructions given.

1.7.2 snakemake

Note: The uORF-Tools require snakemake (version==5.4.5)

The newest version of snakemake can be downloaded via conda using the following command:

```
$ conda create -c conda-forge -c bioconda -n snakemake snakemake==5.4.5
```

This creates a new conda environment called **snakemake** and installs **snakemake** into the environment. The environment can be activated using:

```
$ conda activate snakemake
```

and deactivated using:

```
$ conda deactivate
```

1.7.3 uORF-Tools

Using the workflow requires the **uORF-Tools**. The latest version is available on our GitHub page.

In order to run the workflow, we suggest that you download the **uORF-Tools** into your project directory. The following command creates an example directory and changes into it:

```
$ mkdir project
$ cd project
```

Now, download and unpack the latest version of the **uORF-Tools** by entering the following commands:

```
$ wget https://github.com/Biochemistry1-FFM/uORF-Tools/archive/3.2.1.tar.gz
$ tar -xzf 3.2.1.tar.gz; mv uORF-Tools-3.2.1 uORF-Tools; rm 3.2.1.tar.gz;
```

The **uORF-Tools** are now a subdirectory of your project directory.

1.8 Input files

Several input files are required in order to run our workflow, a genome sequence (.fa), an annotation file (.gtf) and the bam files (.bam).

1.8.1 genome.fa and annotation.gtf

We recommend retrieving both the genome and the annotation files for mouse and human from GENCODE [HFG+12] and for other species from Ensembl Genomes [ZAA+18].

Note: For detailed information about downloading and unpacking these files, please refer to our *example-workflow*.

1.8.2 input .bam files

These are the input files provided by you (the user).

"For best performance, reads should be trimmed (to ~ 29 nt RPF length) and aligned to genome using end-to-end mode (no soft-clip). Intron splicing is supported. Some attributes are needed such as NM, NH and MD. For STAR, *–outSAMattributes All* should be set. bam file should be sorted and indexed by samtools." (RiboTISH requirements, see https://github.com/zhpn1024/ribotish).

Please ensure that you move all input *.bam* files into a folder called **bam** (Located in your project folder):

```
$ mkdir bam
$ cp *.bam bam/
```

1.8.3 Sample sheet and configuration file

In order to run the **uORF-Tools**, you have to provide a sample sheet and a configuration file. There are templates for both files available in the **uORF-Tools** folder.

Copy the templates of the sample sheet and the configuration file into the **uORF-Tools** folder:

```
$ cp uORF-Tools/templates/samples.tsv uORF-Tools/
$ cp uORF-Tools/templates/config.yaml uORF-Tools/
```

Customize the **config.yaml** using your preferred editor. It contains the following variables:

- **taxonomy** Specify the taxonomic group of the used organism in order to ensure the correct removal of reads mapping to ribosomal genes (Eukarya, Bacteria, Archea). (Option for the preprocessing workflow)
- **adapter** Specify the adapter sequence to be used. If not set, *Trim galore* will try to determine it automatically. (Option for the preprocessing workflow)
- **samples** The location of the samples sheet created in the previous step.
- **genomeindexpath** If the STAR genome index was already precomputed, you can specify the path to the files here, in order to avoid recomputation. (Option for the preprocessing workflow)
- **uorfannotationpath** If a uORF-annotation file was already pre-computed, you can specify the path to the file here. Please make sure, that the file has the same format as the uORF_annotation_hg38.csv file provided in the git repo (i.e. same number of columns, same column names)
- **alternativestartcodons** Specify a comma separated list of alternative start codons.

Edit the sample sheet corresponding to your project. It contains the following variables:

- **method** Indicates the method used for this project, here RIBO for ribosome profiling.
- **condition** Indicates the applied condition (e.g. A, B, . . .).
- **replicate** ID used to distinguish between the different replicates (e.g. 1,2, . . .)
- **inputFile** Indicates the according bam file for a given sample.

As seen in the *bam-samples.tsv* template:

method	condition	replicate	inputFile
RIBO	A	1	bam/RIBO-A-1.bam
RIBO	A	2	bam/RIBO-A-2.bam
RIBO	A	3	bam/RIBO-A-3.bam
RIBO	A	4	bam/RIBO-A-4.bam
RIBO	B	1	bam/RIBO-B-1.bam
RIBO	B	2	bam/RIBO-B-2.bam
RIBO	B	3	bam/RIBO-B-3.bam
RIBO	B	4	bam/RIBO-B-4.bam

Warning: Please make sure that you have at-least two replicates for each condition!

Warning: Please ensure that you put the treatment before the control alphabetically (e.g. A: Treatment B: Control)

1.8.4 cluster.yaml

In the **template** folder, we provide two cluster.yaml files needed by snakemake in order to run on a cluster system:

- **sge-cluster.yaml** - for grid based queuing systems
- **torque-cluster.yaml** - for torque based queuing systems

1.9 Example-workflow

A detailed step by step tutorial is available at: *example-workflow*.

1.10 Preprocessing-workflow

We also provide an preprocessing workflow containing a preprocessing step, starting with fastq files. A detailed step by step tutorial is available at: *preprocessing-workflow*.

1.11 References

CHAPTER 2

Example workflow

The retrieval of input files and running the workflow locally and on a server cluster via a queuing system is demonstrated using an example with data available from our FTP-Server. The dataset is available under the GEO accession number *GSE103719*.

Note: Ensure that you have **miniconda3** installed and a conda environment set-up. Please refer to the *overview* for details on the installation.

2.1 Setup

First of all, we start by creating the project directory and changing to it.

```
$ mkdir project
$ cd project
```

We then download the latest version of the **uORF-Tools** into the newly created project folder and unpack it.

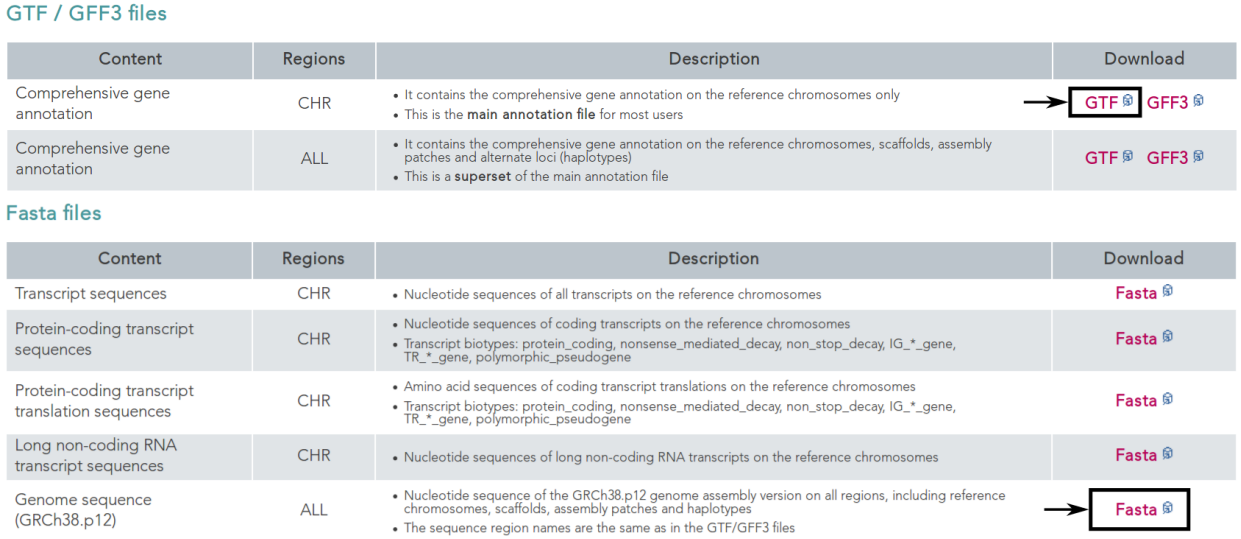
```
$ wget https://github.com/Biochemistry1-FFM/uORF-Tools/archive/3.2.1.tar.gz
$ tar -xzf 3.2.1.tar.gz; mv uORF-Tools-3.2.1 uORF-Tools; rm 3.2.1.tar.gz;
```

2.2 Retrieve and prepare input files

Before starting the workflow, we have to acquire and prepare several input files. These files are the annotation file, the genome file, the bam files, the configuration file and the sample sheet.

2.2.1 Annotation and genome files

First, we want to retrieve the annotation file and the genome file. In this case, we can find both on the GENCODE [HFG+12] webpage for the human genome.



GTF / GFF3 files

Content	Regions	Description	Download
Comprehensive gene annotation	CHR	• It contains the comprehensive gene annotation on the reference chromosomes only • This is the **main annotation file** for most users	GTF GFF3
Comprehensive gene annotation	ALL	• It contains the comprehensive gene annotation on the reference chromosomes, scaffolds, assembly patches and alternate loci (haplotypes) • This is a **superset** of the main annotation file	GTF GFF3

Fasta files

Content	Regions	Description	Download
Transcript sequences	CHR	• Nucleotide sequences of all transcripts on the reference chromosomes	Fasta
Protein-coding transcript sequences	CHR	• Nucleotide sequences of coding transcripts on the reference chromosomes • Transcript biotypes: protein_coding, nonsense_mediated_decay, non_stop_decay, IG_*_gene, TR_*_gene, polymorphic_pseudogene	Fasta
Protein-coding transcript translation sequences	CHR	• Amino acid sequences of coding transcript translations on the reference chromosomes • Transcript biotypes: protein_coding, nonsense_mediated_decay, non_stop_decay, IG_*_gene, TR_*_gene, polymorphic_pseudogene	Fasta
Long non-coding RNA transcript sequences	CHR	• Nucleotide sequences of long non-coding RNA transcripts on the reference chromosomes	Fasta
Genome sequence (GRCh38.p12)	ALL	• Nucleotide sequence of the GRCh38.p12 genome assembly version on all regions, including reference chromosomes, scaffolds, assembly patches and haplotypes • The sequence region names are the same as in the GTF/GFF3 files	Fasta

On this page, we can directly retrieve both files by clicking on the according download links next to the file descriptions. Alternatively, you can directly download them using the following commands:

```
$ wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_28/gencode.v28.
↪annotation.gtf.gz && wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/
↪release_28/GRCh38.p12.genome.fa.gz
```

Then, we are going to unpack both files.

```
$ gunzip gencode.v28.annotation.gtf.gz && gunzip GRCh38.p12.genome.fa.gz
```

Finally, we will rename these files to *annotation.gtf* and *genome.fa*.

```
$ mv gencode.v28.annotation.gtf annotation.gtf && mv GRCh38.p12.genome.fa genome.fa
```

Another webpage that provides these files is Ensembl Genomes [ZAA+18]. This usually requires searching their file system in order to find the wanted files. For this tutorial, we recommend to stick to GenCode instead.

2.2.2 .bam files

Next, we want to acquire the bam files. The bam files for the tutorial dataset can be downloaded from our FTP-Server:

Note: We provide both a .zip and a .tar.gz file. We recommend the .tar.gz file as most linux systems can decompress them via commandline by default.

```
$ wget ftp://biftp.informatik.uni-freiburg.de/pub/uORF-Tools/bam.tar.gz; tar -zxvf␣
↪bam.tar.gz; rm bam.tar.gz;
```

This will create a bam folder containing all the files necessary to run the workflow. If you prefer using your own .bam files, create a bam folder and copy the files into it. Make sure that your reads were trimmed (to ~29bp length) and

aligned to the genome using end-to-end alignment. The bam files need to include all SAM attributes and should be sorted using samtools.

```
$ mkdir bam
$ cp *.bam bam/
```

2.2.3 Sample sheet and configuration file

Finally, we will prepare the configuration file (*config.yaml*) and the sample sheet (*samples.tsv*). We start by copying templates for both files from the *uORF-Tools/templates/* into the *uORF-Tools/* folder.

```
$ cp uORF-Tools/templates/bam-samples.tsv uORF-Tools/
$ mv uORF-Tools/bam-samples.tsv uORF-Tools/samples.tsv
```

The sample file looks as follows:

method	condition	replicate	inputFile
RIBO	A	1	bam/RIBO-A-1.bam
RIBO	A	2	bam/RIBO-A-2.bam
RIBO	A	3	bam/RIBO-A-3.bam
RIBO	A	4	bam/RIBO-A-4.bam
RIBO	B	1	bam/RIBO-B-1.bam
RIBO	B	2	bam/RIBO-B-2.bam
RIBO	B	3	bam/RIBO-B-3.bam
RIBO	B	4	bam/RIBO-B-4.bam

Note: When using your own data, use any editor (vi(m), gedit, nano, atom, . . .) to customize the sample sheet.

Warning: Please ensure not to replace any tabulator symbols with spaces while changing this file.

Next, we are going to set up the *config.yaml*.

```
$ cp uORF-Tools/templates/config.yaml uORF-Tools/
$ vi uORF-Tools/config.yaml
```

This file contains the following variables:

- **taxonomy** Specify the taxonomic group of the used organism in order to ensure the correct removal of reads mapping to ribosomal genes (Eukarya, Bacteria, Archea). (Option for the preprocessing workflow)
- **adapter** Specify the adapter sequence to be used. If not set, *Trim galore* will try to determine it automatically. (Option for the preprocessing workflow)
- **samples** The location of the samples sheet created in the previous step.
- **genomeindexpath** If the STAR genome index was already precomputed, you can specify the path to the files here, in order to avoid recomputation. (Option for the preprocessing workflow)
- **uorfannotationpath** If a uORF-annotation file was already pre-computed, you can specify the path to the file here. Please make sure, that the file has the same format as the uORF_annotation_hg38.csv file provided in the git repo (i.e. same number of columns, same column names)
- **alternativestartcodons** Specify a comma separated list of alternative start codons.

```
#Taxonomy of the samples to be processed, possible are Eukarya, Bacteria, Archea
taxonomy: "Eukarya"
#Adapter sequence used
adapter: ""
samples: "uORF-Tools/samples.tsv"
genomeindexpath: ""
uorfannotationpath: ""
alternativestartcodons: ""
```

For this tutorial, we can keep the default values for the *config.yaml*. The organism analyzed in this tutorial is *homo sapiens*, therefore we keep the taxonomy at *Eukarya*. The path to *samples.tsv* is set correctly.

2.3 Running the workflow

Now that we have all the required files, we can start running the workflow, either locally or in a cluster environment.

2.3.1 Run the workflow locally

Use the following steps when you plan to execute the workflow on a single server or workstation. Please be aware that some steps of the workflow might require a lot of memory, specifically for eukaryotic species.

Navigate to the project folder containing the bam/ folder, the annotation.gtf and the genome.fa files and the uORF-Tools folder. Start the workflow locally from this folder by running:

```
$ snakemake --use-conda -s uORF-Tools/Snakefile --configfile uORF-Tools/config.yaml --
↪directory ${PWD} -j 20 --latency-wait 60
```

2.3.2 Run Snakemake in a cluster environment

Use the following steps if you are executing the workflow via a queuing system. Edit the configuration file *cluster.yaml* according to your queuing system setup and cluster hardware.

Navigate to the project folder on your cluster system. Start the workflow from this folder by running (The following system call shows the usage with Grid Engine.):

```
$ snakemake --use-conda -s uORF-Tools/Snakefile --configfile uORF-Tools/config.yaml --
↪directory ${PWD} -j 20 --cluster-config uORF-Tools/templates/sge-cluster.yaml
```

2.3.3 Example: Run Snakemake in a cluster environment

Warning: Be advised that this is a specific example, the required options may change depending on your system.

We ran the tutorial workflow in a cluster environment, specifically a TORQUE cluster environment. Therefore, we created a bash script *torque.sh* in our project folder.

```
$ vi torque.sh
```

Note: Please note that all arguments enclosed in <> have to be customized. This script will only work if your cluster uses the TORQUE queuing system.

We proceeded by writing the queuing script:

```
#!/bin/bash
#PBS -N <ProjectName>
#PBS -S /bin/bash
#PBS -q "long"
#PBS -d <PATH/ProjectFolder>
#PBS -l nodes=1:ppn=1
#PBS -o <PATH/ProjectFolder>
#PBS -j oe
cd <PATH/ProjectFolder>
source activate uORF-Tools
snakemake --latency-wait 600 --use-conda -s uORF-Tools/Snakefile --configfile uORF-
↪Tools/config.yaml --directory ${PWD} -j 20 --cluster-config uORF-Tools/templates/
↪torque-cluster.yaml --cluster "qsub -N {cluster.jobname} -S /bin/bash -q {cluster.
↪qname} -d <PATH/ProjectFolder> -l {cluster.resources} -o {cluster.logoutputdir} -j␣
↪oe"
```

We then simply submitted this job to the cluster:

```
$ qsub torque.sh
```

Using any of the presented methods, this will run the workflow on our dataset and create the desired output files.

2.3.4 Report

Once the workflow has finished, we can request an automatically generated *report.html* file using the following command:

```
$ snakemake --use-conda -s uORF-Tools/Snakefile --configfile uORF-Tools/config.yaml --
↪report report.html
```

2.4 References

CHAPTER 3

Preprocessing workflow

The retrieval of input files and running the workflow locally and on a server cluster via a queuing system is demonstrated using an example with data available from SRA via NCBI. The dataset is available under the GEO accession number *GSE103719*. The retrieval of the data is described in this tutorial.

Note: Ensure that you have **miniconda3** installed and a conda environment set-up. Please refer to the *overview* for details on the installation.

3.1 Setup

First of all, we start by creating the project directory and changing to it.

```
$ mkdir preprocessing_project
$ cd preprocessing_project
```

We then download the latest version of the uORF-Tools into the newly created project folder and unpack it.

```
$ wget https://github.com/Biochemistry1-FFM/uORF-Tools/archive/3.2.1.tar.gz
$ tar -xzf 3.2.1.tar.gz; mv uORF-Tools-3.2.1 uORF-Tools; rm 3.2.1.tar.gz;
```

3.2 Retrieve and prepare input files

Before starting the workflow, we have to acquire and prepare several input files. These files are the annotation file, the genome file, the fastq files, the configuration file and the sample sheet.

3.2.1 Annotation and genome files

```
$ wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_28/gencode.v28.
↪annotation.gtf.gz && wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/
↪release_28/GRCh38.p12.genome.fa.gz
```

Then, we are going to unpack both files:

```
$ gunzip gencode.v28.annotation.gtf.gz && gunzip GRCh38.p12.genome.fa.gz
```

Finally, we will rename these files to *annotation.gtf* and *genome.fa*.

```
$ mv gencode.v28.annotation.gtf annotation.gtf && mv GRCh38.p12.genome.fa genome.fa
```

Another webpage that provides these files is Ensembl Genomes [ZAA+18]. This usually requires searching their file system in order to find the wanted files. For this tutorial, we recommend to stick to GenCode instead.

3.2.2 Fastq files

In this example, we will use both *RNA-seq* and *RIBO-seq* data. In order to fasten up the tutorial, we download only 2 of the 4 replicates available for each Condition.

Note: Please note that you should always use all available replicates, when analyzing your data.

European Nucleotide Archive (ENA)

For many datasets, the easiest way to retrieve the fastq files is using the European Nucleotide Archive [SAA+17] as it provides direct download links when searching for a dataset. Use the interface on ENA or type the follwing commands:

```
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/005/SRR6026765/SRR6026765.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/006/SRR6026766/SRR6026766.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/009/SRR6026769/SRR6026769.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/000/SRR6026770/SRR6026770.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/003/SRR6026773/SRR6026773.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/004/SRR6026774/SRR6026774.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/007/SRR6026777/SRR6026777.fastq.gz;
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR602/008/SRR6026778/SRR6026778.fastq.gz;
```

Then, we create a fastq folder and move all the *.fastq.gz* files into this folder.

```
$ mkdir fastq; mv *.fastq.gz fastq/;
```

3.2.3 Sample sheet and configuration file

Finally, we will prepare the configuration file (*config.yaml*) and the sample sheet (*samples.tsv*). We start by copying templates for both files from the *uORF-Tools/templates/* into the *uORF-Tools/* folder.

```
$ cp uORF-Tools/templates/fastq-samples.tsv uORF-Tools/
```

The template looks as follows:

method	condition	replicate	inputFile
RNA	A	1	fastq/SRR6026769.fastq.gz
RNA	A	2	fastq/SRR6026770.fastq.gz
RNA	B	1	fastq/SRR6026765.fastq.gz
RNA	B	2	fastq/SRR6026766.fastq.gz
RIBO	A	1	fastq/SRR6026777.fastq.gz
RIBO	A	2	fastq/SRR6026778.fastq.gz
RIBO	B	1	fastq/SRR6026773.fastq.gz
RIBO	B	2	fastq/SRR6026774.fastq.gz

Warning: Please ensure that you do not replace any tabulator symbols with spaces while changing this file.

Next, we are going to set up the *config.yaml*.

```
$ cp uORF-Tools/templates/config.yaml uORF-Tools
$ vi uORF-Tools/config.yaml
```

This file contains the following variables:

- **taxonomy** Specify the taxonomic group of the used organism in order to ensure the correct removal of reads mapping to ribosomal genes (Eukarya, Bacteria, Archea). (Option for the preprocessing workflow)
- **adapter** Specify the adapter sequence to be used. If not set, *Trim galore* will try to determine it automatically. (Option for the preprocessing workflow)
- **samples** The location of the samples sheet created in the previous step.
- **genomeindexpath** If the STAR genome index was already precomputed, you can specify the path to the files here, in order to avoid recomputation. (Option for the preprocessing workflow)
- **uorfannotationpath** If a uORF-annotation file was already pre-computed, you can specify the path to the file here. Please make sure, that the file has the same format as the uORF_annotation_hg38.csv file provided in the git repo (i.e. same number of columns, same column names)
- **alternativestartcodons** Specify a comma separated list of alternative start codons.

Change the config file as follows:

```
#Taxonomy of the samples to be processed, possible are Eukarya, Bacteria, Archea
taxonomy: "Eukarya"
#Adapter sequence used
adapter: ""
samples: "uORF-Tools/fastq-samples.tsv"
genomeindexpath: ""
uorfannotationpath: ""
alternativestartcodons: ""
```

3.3 Running the workflow

Now that we have all the required files, we can start running the workflow, either locally or in a cluster environment.

3.3.1 Information about processing parameters

In this pipeline we use trim_galore for adapter and quality trimming (Parameters: –phred33 -q 20 –length 15 –trim-n –suppress_warn –clip_R1 1), sortmerna for rRNA removal (rRNA fasta files are obtained from https://github.com/biocore/sortmerna/tree/master/rRNA_databases, according to the specified Taxon (i.e. Eukarya, Bacteria, Archea)) and STAR for read alignment (Parameters: –outSAMtype BAM SortedByCoordinate –outSAMattributes All –outFilterMultimapNmax 1 –alignEndsType Extend5pOfRead1).

3.3.2 Run the workflow locally

Use the following steps when you plan to execute the workflow on a single server or workstation. Please be aware that some steps of the workflow require a lot of memory, specifically for eukaryotic species.

```
$ snakemake --use-conda -s uORF-Tools/Preprocessing_Snakefile --configfile uORF-Tools/
↪config.yaml --directory ${PWD} -j 20 --latency-wait 60
```

3.3.3 Run Snakemake in a cluster environment

Use the following steps if you are executing the workflow via a queuing system. Edit the configuration file *cluster.yaml* according to your queuing system setup and cluster hardware. The following system call shows the usage with Grid Engine:

```
$ snakemake --use-conda -s uORF-Tools/Preprocessing_Snakefile --configfile uORF-Tools/
↪config.yaml --directory ${PWD} -j 20 --cluster-config uORF-Tools/cluster.yaml
```

3.3.4 Example: Run Snakemake in a cluster environment

Warning: Be advised that this is a specific example, the required options may change depending on your system.

We ran the tutorial workflow in a cluster environment, specifically a TORQUE cluster environment. Therefore, we created a bash script *torque.sh* in our project folder.

```
$ vi torque.sh
```

We proceeded by writing the queueing script:

```
#!/bin/bash
#PBS -N <ProjectName>
#PBS -S /bin/bash
#PBS -q "long"
#PBS -d <PATH/ProjectFolder>
#PBS -l nodes=1:ppn=1
#PBS -o <PATH/ProjectFolder>
#PBS -j oe
cd <PATH/ProjectFolder>
source activate uORF-Tools
snakemake --latency-wait 600 --use-conda -s uORF-Tools/Preprocessing_Snakefile --
↪configfile uORF-Tools/config.yaml --directory ${PWD} -j 20 --cluster-config uORF-
↪Tools/templates/torque-cluster.yaml --cluster "qsub -N {cluster.jobname} -S /bin/
↪bash -q {cluster.qname} -d <PATH/ProjectFolder> -l {cluster.resources} -o {cluster.
↪logoutputdir} -j oe"
```

(continues on next page)

(continued from previous page)

We then simply submitted this job to the cluster:

```
$ qsub torque.sh
```

Using any of the presented methods, this will run the workflow on our dataset and create the desired output files.

Bibliography

[GruningDSjodin+17] Björn Grüning, Ryan Dale, Andreas Sjödin, Jillian Rowe, Brad A. Chapman, Christopher H. Tomkins-Tinch, Renan Valieris, and Johannes Köster. Bioconda: a sustainable and comprehensive software distribution for the life sciences. *bioRxiv*, 2017. URL: https://www.biorxiv.org/content/early/2017/10/27/207092, arXiv:https://www.biorxiv.org/content/early/2017/10/27/207092.full.pdf, doi:10.1101/207092.

[HFG+12] J. Harrow, A. Frankish, J. M. Gonzalez, E. Tapanari, M. Diekhans, F. Kokocinski, B. L. Aken, D. Barrell, A. Zadissa, S. Searle, I. Barnes, A. Bignell, V. Boychenko, T. Hunt, M. Kay, G. Mukherjee, J. Rajan, G. Despacio-Reyes, G. Saunders, C. Steward, R. Harte, M. Lin, C. Howald, A. Tanzer, T. Derrien, J. Chrast, N. Walters, S. Balasubramanian, B. Pei, M. Tress, J. M. Rodriguez, I. Ezkurdia, J. van Baren, M. Brent, D. Haussler, M. Kellis, A. Valencia, A. Reymond, M. Gerstein, R. Guigo, and T. J. Hubbard. GENCODE: the reference human genome annotation for The ENCODE Project. *Genome Res.*, 22(9):1760–1774, Sep 2012.

[KosterR18] Johannes Köster and Sven Rahmann. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, ():bty350, 2018. URL: http://dx.doi.org/10.1093/bioinformatics/bty350, arXiv:/oup/backfile/content_public/journal/bioinformatics/pap/10.1093_bioinformatics_bty350/2/bty350.pdf, doi:10.1093/bioinformatics/bty350.

[ZAA+18] Daniel R Zerbino, Premanand Achuthan, Wasiu Akanni, M Ridwan Amode, Daniel Barrell, Jyothish Bhai, Konstantinos Billis, Carla Cummins, Astrid Gall, Carlos García Girón, Laurent Gil, Leo Gordon, Leanne Haggerty, Erin Haskell, Thibaut Hourlier, Osagie G Izuogu, Sophie H Janacek, Thomas Juettemann, Jimmy Kiang To, Matthew R Laird, Ilias Lavidas, Zhicheng Liu, Jane E Loveland, Thomas Maurel, William McLaren, Benjamin Moore, Jonathan Mudge, Daniel N Murphy, Victoria Newman, Michael Nuhn, Denye Ogeh, Chuang Kee Ong, Anne Parker, Mateus Patricio, Harpreet Singh Riat, Helen Schuilenburg, Dan Sheppard, Helen Sparrow, Kieron Taylor, Anja Thormann, Alessandro Vullo, Brandon Walts, Amonida Zadissa, Adam Frankish, Sarah E Hunt, Myrto Kostadima, Nicholas Langridge, Fergal J Martin, Matthieu Muffato, Emily Perry, Magali Ruffier, Dan M Staines, Stephen J Trevanion, Bronwen L Aken, Fiona Cunningham, Andrew Yates, and Paul Flicek. Ensembl 2018. *Nucleic Acids Research*, 46(D1):D754–D761, 2018. URL: http://dx.doi.org/10.1093/nar/gkx1098, arXiv:/oup/backfile/content_public/journal/nar/46/d1/10.1093_nar_gkx1098/2/gkx1098.pdf, doi:10.1093/nar/gkx1098.

[HFG+12] J. Harrow, A. Frankish, J. M. Gonzalez, E. Tapanari, M. Diekhans, F. Kokocinski, B. L. Aken, D. Barrell, A. Zadissa, S. Searle, I. Barnes, A. Bignell, V. Boychenko, T. Hunt, M. Kay, G. Mukherjee, J. Rajan, G. Despacio-Reyes, G. Saunders, C. Steward, R. Harte, M. Lin, C. Howald, A. Tanzer, T. Derrien, J. Chrast, N. Walters, S. Balasubramanian, B. Pei, M. Tress, J. M. Rodriguez, I. Ezkurdia, J. van Baren,

M. Brent, D. Haussler, M. Kellis, A. Valencia, A. Reymond, M. Gerstein, R. Guigo, and T. J. Hubbard. GENCODE: the reference human genome annotation for The ENCODE Project. *Genome Res.*, 22(9):1760–1774, Sep 2012.

[ZAA+18] Daniel R Zerbino, Premanand Achuthan, Wasiu Akanni, M Ridwan Amode, Daniel Barrell, Jyothish Bhai, Konstantinos Billis, Carla Cummins, Astrid Gall, Carlos García Girón, Laurent Gil, Leo Gordon, Leanne Haggerty, Erin Haskell, Thibaut Hourlier, Osagie G Izuogu, Sophie H Janacek, Thomas Juettemann, Jimmy Kiang To, Matthew R Laird, Ilias Lavidas, Zhicheng Liu, Jane E Loveland, Thomas Maurel, William McLaren, Benjamin Moore, Jonathan Mudge, Daniel N Murphy, Victoria Newman, Michael Nuhn, Denye Ogeh, Chuang Kee Ong, Anne Parker, Mateus Patricio, Harpreet Singh Riat, Helen Schuilenburg, Dan Sheppard, Helen Sparrow, Kieron Taylor, Anja Thormann, Alessandro Vullo, Brandon Walts, Amonida Zadissa, Adam Frankish, Sarah E Hunt, Myrto Kostadima, Nicholas Langridge, Fergal J Martin, Matthieu Muffato, Emily Perry, Magali Ruffier, Dan M Staines, Stephen J Trevanion, Bronwen L Aken, Fiona Cunningham, Andrew Yates, and Paul Flicek. Ensembl 2018. *Nucleic Acids Research*, 46(D1):D754–D761, 2018. URL: http://dx.doi.org/10.1093/nar/gkx1098, arXiv:/oup/backfile/content_public/journal/nar/46/d1/10.1093_nar_gkx1098/2/gkx1098.pdf, doi:10.1093/nar/gkx1098.

[GruningDSjodin+17] Björn Grüning, Ryan Dale, Andreas Sjödin, Jillian Rowe, Brad A. Chapman, Christopher H. Tomkins-Tinch, Renan Valieris, and Johannes Köster. Bioconda: a sustainable and comprehensive software distribution for the life sciences. *bioRxiv*, 2017. URL: https://www.biorxiv.org/content/early/2017/10/27/207092, arXiv:https://www.biorxiv.org/content/early/2017/10/27/207092.full.pdf, doi:10.1101/207092.

[HFG+12] J. Harrow, A. Frankish, J. M. Gonzalez, E. Tapanari, M. Diekhans, F. Kokocinski, B. L. Aken, D. Barrell, A. Zadissa, S. Searle, I. Barnes, A. Bignell, V. Boychenko, T. Hunt, M. Kay, G. Mukherjee, J. Rajan, G. Despacio-Reyes, G. Saunders, C. Steward, R. Harte, M. Lin, C. Howald, A. Tanzer, T. Derrien, J. Chrast, N. Walters, S. Balasubramanian, B. Pei, M. Tress, J. M. Rodriguez, I. Ezkurdia, J. van Baren, M. Brent, D. Haussler, M. Kellis, A. Valencia, A. Reymond, M. Gerstein, R. Guigo, and T. J. Hubbard. GENCODE: the reference human genome annotation for The ENCODE Project. *Genome Res.*, 22(9):1760–1774, Sep 2012.

[KosterR18] Johannes Köster and Sven Rahmann. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, ():bty350, 2018. URL: http://dx.doi.org/10.1093/bioinformatics/bty350, arXiv:/oup/backfile/content_public/journal/bioinformatics/pap/10.1093_bioinformatics_bty350/2/bty350.pdf, doi:10.1093/bioinformatics/bty350.

[SAA+17] Nicole Silvester, Blaise Alako, Clara Amid, Ana Cerdeño-Tarrága, Laura Clarke, Iain Cleland, Peter W Harrison, Suran Jayathilaka, Simon Kay, Thomas Keane, and others. The european nucleotide archive in 2017. *Nucleic acids research*, 46(D1):D36–D40, 2017.

[ZAA+18] Daniel R Zerbino, Premanand Achuthan, Wasiu Akanni, M Ridwan Amode, Daniel Barrell, Jyothish Bhai, Konstantinos Billis, Carla Cummins, Astrid Gall, Carlos García Girón, Laurent Gil, Leo Gordon, Leanne Haggerty, Erin Haskell, Thibaut Hourlier, Osagie G Izuogu, Sophie H Janacek, Thomas Juettemann, Jimmy Kiang To, Matthew R Laird, Ilias Lavidas, Zhicheng Liu, Jane E Loveland, Thomas Maurel, William McLaren, Benjamin Moore, Jonathan Mudge, Daniel N Murphy, Victoria Newman, Michael Nuhn, Denye Ogeh, Chuang Kee Ong, Anne Parker, Mateus Patricio, Harpreet Singh Riat, Helen Schuilenburg, Dan Sheppard, Helen Sparrow, Kieron Taylor, Anja Thormann, Alessandro Vullo, Brandon Walts, Amonida Zadissa, Adam Frankish, Sarah E Hunt, Myrto Kostadima, Nicholas Langridge, Fergal J Martin, Matthieu Muffato, Emily Perry, Magali Ruffier, Dan M Staines, Stephen J Trevanion, Bronwen L Aken, Fiona Cunningham, Andrew Yates, and Paul Flicek. Ensembl 2018. *Nucleic Acids Research*, 46(D1):D754–D761, 2018. URL: http://dx.doi.org/10.1093/nar/gkx1098, arXiv:/oup/backfile/content_public/journal/nar/46/d1/10.1093_nar_gkx1098/2/gkx1098.pdf, doi:10.1093/nar/gkx1098.