# Adapter trimming in Ubuntu using Cutadapt

## Notes

* The QIAseq miRNA 5' UDI adapter is CTACACGACGCTCTTCCGATCT. This was trimmed on BaseSpace during the RNA-seq run so it is not present in the data.
* The QIAseq miRNA 3’ NGS adapter is made up of three parts: SEQUENCE1SEQUENCE2SEQUENCE3.
* SEQUENCE 1 is always AACTGTAGGCACCATCAAT. This is the sequence provided by Qiagen as the sequence of the QIAseq miRNA 3’ NGS adapter (but in reality it has other things appended to it as shown below).
* SEQUENCE 2 is a random UMI of 12 nucleotides (e.g., AGTCTAAGCGGT).
* SEQUENCE 3 is always the Illumina TruSeq adapter: AGATCGGAAGAGCACACGTCTGAACTCCAGTCA.
* Therefore, a full QIAseq miRNA 3’ NGS adapter could be AACTGTAGGCACCATCAATCACCCGTCTGAAAGATCGGAAGAGCACACGTCTGAACTCCAGTCA, and another one could be AACTGTAGGCACCATCAATAGTCTAAGCGGTAGATCGGAAGAGCACACGTCTGAACTCCAGTCA, etc.

## Cutadapt installation instructions for Ubuntu

1. Open an Ubuntu terminal.
2. Type the following command to update your package list:

sudo apt update

* 1. If it asks you a [Y/n] question, type **Y** and press enter.

1. Ensure you have virtualenv installed with this command:

sudo apt install python3-virtualenv

1. Create a new virtual environment and install Cutadapt into it with these commands:

virtualenv cutadapt-venv

cutadapt-venv/bin/pip --upgrade pip

cutadapt-venv/bin/pip install cutadapt

1. Activate the virtual environment, which allows you to just type cutadapt without the full path, with this command:

source cutadapt-venv/bin/activate

1. Confirm that Cutadapt installed successfully using this command:

cutadapt –version

1. The full Cutadapt guide is at <https://cutadapt.readthedocs.io/en/stable/index.html>.
2. **Every time a new terminal is opened, the command at step 5 needs to be run again before using cutadapt.**
3. This tool will first trim the 3’ adapter based on the first part of its sequence (SEQUENCE1 as shown above). Any sequence **after** SEQUENCE1 will also be trimmed together with SEQUENCE 1.
4. If the adapter is missing the first few nucleotides of SEQUENCE1, this tool does not recognise it as an adapter and it thinks that the read does not contain any adapters. This only happens when the adapter starts in the first few nucleotides of the read, and this indicates that the entire read is made up of an adapter sequence. This read remains untrimmed. The tool is told to discard all untrimmed reads so these are also removed.
5. Next, after all trimming has been completed, and the untrimmed reads have been discarded, the tool will also discard any reads that are less than 14 nucleotides long. This is because the ncRNA sequences expected to exist in the data (from this project) will never be shorter than 14 nucleotides long, and therefore anything shorter is not a real ncRNA sequence (it is probably fragmented RNA) and is filtered out as it cannot be analysed further.

## Automatically process all files with Cutadapt using a shell script

1. Open an Ubuntu terminal.
2. Activate the virtual environment with this command:

source cutadapt-venv/bin/activate

1. Navigate to the folder containing the .fastq.gz files to be processed by Cutadapt, with this command:

cd “/mnt/path/to/Concatenated\_fastq\_files”

1. Place the **run\_cutadapt.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_cutadapt.sh

1. Run the script with this command:

./run\_cutadapt.sh

1. This script will create two output files per input .fastq.gz file and save them in a folder called **Cutadapt output**. This folder will be next to the **Concatenated\_fastq\_files** folder.
   1. The first output file is a .fastq.gz file containing the trimmed reads for that sample. It will have the same filename as the input .fastq.gz file with “\_trimmed” appended to it.
   2. The second output file is a .txt file containing the summary statistics for that sample. It will have the same filename as the input .fastq.gz file with “\_trim\_report” appended to it.
2. Open MATLAB.
3. Make sure the **saveCutadaptStats.m** file exists. It can be stored anywhere on the computer.
4. Add this .m file to the MATLAB path (you can open the file in MATLAB, then right-click on the file tab and select Add to path).
5. Run this MATLAB function with this command:

saveCutadaptStats("path/to/Cutadapt output")

1. You can use the data in the Excel file saved by the above function to produce useful figures summarising the Cutadapt results.

# Assess quality of sequencing reads with FastQC

## FastQC installation instructions for Ubuntu

1. Open an Ubuntu terminal.
2. Type the following command to update your package list:

sudo apt update

* 1. If it asks you a [Y/n] question, type **Y** and press enter.

1. Install FastQC with this command:

sudo apt -y install fastqc

1. Verify successful installation with this command:

fastqc –-version

1. Install unzip (needed by a shell script later) with this command:

sudo apt install unzip

## Quality control analysis with FastQC in Ubuntu

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the .fastq.gz files to be processed by FastQC, using this command:

cd "/mnt/path/to/Cutadapt output"

1. Place the **run\_FastQC.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_FastQC.sh

1. Run the script with this command:

./run\_FastQC.sh

1. This script will create one output file per input file and save it in a folder called **FastQC output**. This folder will be next to the **Cutadapt output** folder. The output file is a .txt file containing the FastQC output data for that sample. It will have the same filename as the input .fastq.gz file with “\_fastQC” appended to it.
2. Open MATLAB.
3. Make sure the **extract\_fastqc\_data\_to\_excel.m** file exists. It can be stored anywhere on the computer.
4. Add this .m file to the MATLAB path (you can open the file in MATLAB, then right-click on the file tab and select **Add to path**).
5. Run this MATLAB function with this command:

exctract\_fastqc\_data\_to\_excel

* 1. Five Excel files will be saved in the current directory.

1. Run the MATLAB function **append\_info\_to\_fastqc\_excel.m** with this command (the current directory should be the **FastQC output** folder):

append\_info\_to\_fatqc\_excel

* 1. Data will be added to column A of the five Excel files created by the **extract\_fastqc\_data\_to\_excel.m** function.

1. You can use the data from these five Excel files to create useful figures summarising the results of FastQC.

# Map reads to miRNAs, tRFs, and piRNAs using multi-step alignment

## Notes

* According to the library preparation kit, only RNA molecules that are ≤50 nt long and have a 3’ OH group and a 5’ PO4 group are included in the RNA-seq library. Therefore, these are the RNAs expected to be present in the reads:

|  |  |  |
| --- | --- | --- |
| **RNA type** | **Length** | **Likely to be found in human plasma?** |
| miRNA | 18 – 22 nt | Yes |
| piRNA | 24 – 32 nt | Yes |
| tRF-1, tRF-3, tRF-5 | 14 – 30 nt | Yes |
| siRNA | 21 – 23 nt | Yes |
| Fragments of other small and long ncRNAs | Various | Yes |

* When using a multi-step alignment approach to quantify different small ncRNAs, we start with the most abundant and highly annotated RNA classes first, and progressively filter out the remaining unmapped reads. The order prioritises classes with higher abundance, better annotation, and stricter sequence constraints to prevent ambiguous or false positive alignments.
* The order used in this project is miRNAs > tRFs > piRNAs > other ncRNAs.
  + miRNAs are well-annotated, highly specific, and abundant in plasma, so capturing them first avoids ambiguity with other classes.
  + tRFs are highly abundant in cell-free RNA and they are distinct enough from miRNAs to come next, avoiding confusion with piRNAs.
  + piRNAs are less abundant than tRFs or miRNAs and their longer length helps distinguish them from other small RNAs.

## Prepare a miRNA reference to map to

1. The miRBase version used for this analysis is 22.1.
2. Go to [www.mirbase.org](http://www.mirbase.org), select **Downloads**, then download the **mature.fa** file. This is a FASTA file containing all mature miRNA sequences of ALL species available on the database (not just human).
3. Save this file inside a new folder called **Bowtie index of miRNA reference**.
4. Rename the file to **mature\_miRNAs.fa**.
5. Open an Ubuntu terminal.
6. Navigate to the folder containing the downloaded file, using this command:
7. cd "/mnt/path/to/Bowtie index of miRNA reference"
8. Extract only the human miRNA sequences into a new file called **mature\_hsa\_miRNAs.fa**, using this command:

grep -A 1 "^>hsa" mature\_miRNAs.fa > mature\_hsa\_miRNAs.fa

1. Convert the RNA letters to DNA letters (i.e., replace U with T), using this command (because the fastq.gz files that we have contain Ts instead of Us):

sed '/^[^>]/s/U/T/g' mature\_hsa\_miRNAs.fa > mature\_hsa\_miRNAs\_DNA.fa

## Prepare a tRF reference to map to

1. A FASTA file with all known human tRFs does not currently exist, so it will be created manually.
2. The database (tRFdb) used for this analysis does not have a version.
3. Go to <http://genome.bioch.virginia.edu/trfdb/>, select **SEARCH TRFDB**, then download each relevant tRF type (i.e., tRF-1, tRF-3, and tRF-5) list as .csv files for human only.
4. Rename the tRF-1 file to **tRF-1.csv**, the tRF-3 file to **tRF-3.csv**, and the tRF-5 file to **tRF-5.csv**.
5. Save these files inside a new folder called **Bowtie index of tRF reference**. This folder should be next to the **Cutadapt output** folder.
6. Create a copy of the **tRF-1.csv** file in the same folder and name it **tRF-1\_IDs.csv**.
7. Open the **tRF-1\_IDs.csv** file and delete all columns except for “tRF ID” and “tRF Sequence”. Also delete the first row (headers).
8. Remove duplicates from column A (expand the selection) so that only unique tRF IDs remain.
9. Repeat steps 6-8 with the **tRF-3.csv file** and then the **tRF-5.csv file**.
   1. **NB:** No duplicates were found at step 8 for tRF-1, but there were duplicates for tRF-3 and tRF-5.
10. Open an Ubuntu terminal.
11. Navigate to the folder containing the downloaded files, using this command:

cd "/mnt/path/to/Bowtie index of tRF reference"

1. Place the shell script **create\_tRF\_FASTA.sh** in this folder.
2. Make sure the script is executable with this command:

chmod +x create\_tRF\_FASTA.sh

1. Run the script with this command:

./create\_tRF\_FASTA.sh

1. This script will create a FASTA file containing the tRF IDs and sequences that will be used as a reference to map to tRF-1, tRF-3, and tRF-5 sequences.
2. **NB:** There is no need to convert the Us to Ts in this file (like what is done for the miRNA reference) because this file already contains Ts instead of Us.

## Prepare a piRNA reference to map to

1. The piRBase version used for this analysis is v3.0.
2. Go to <http://bigdata.ibp.ac.cn/piRBase/>, select **Downloads**, then download the **gold standard set** for human (hsa). This is a compressed FASTA file called **hsa.gold.fa.gz**.
3. Open an Ubuntu terminal.
4. Navigate to the folder containing the downloaded file, using this command:

cd "/mnt/path/to/folder"

1. Generate the MD5 checksum using this command:

md5sum hsa.gold.fa.gz

1. Compare the generated checksum with the MD5 value provided by piRBase. If the two values match, then the file is intact and not corrupted.
2. Unzip the file using this command:

gunzip hsa.gold.fa.gz

1. This will decompress the file to **hsa.gold.fa**. Rename the file to **piRNA\_gold\_standard\_set.fa**. Transfer this file inside a new folder called **Bowtie index of piRNA reference**. This folder should be next to the **Cutadapt output** folder.
2. **NB:** There is no need to convert the Us to Ts in this file (like what is done for the miRNA reference) because this file already contains Ts instead of Us.

## Prepare an “other ncRNA” reference to map to

1. Go to <https://www.ensembl.org/Homo_sapiens/Info/Index>, select **Download FASTA**, then **ncrna**, then download the human ncrna FASTA file (.fa.gz). E.g., the file called **Homo\_sapiens.GRCh38.ncrna.fa.gz**.
2. Open an Ubuntu terminal.
3. Navigate to the folder containing the downloaded file, using this command:

cd "/mnt/path/to/folder"

1. Unzip the file using this command:

gunzip Homo\_sapiens.GRCh38.ncrna.fa.gz

1. This will decompress the file to **Homo\_sapiens.GRCh38.ncrna.fa**. Rename it to **hsa\_ncRNAs.fa**.
2. Save the file in a new folder called **Bowtie index of other ncRNA reference**. This folder should be next to the **Cutadapt output** folder.
3. Navigate to the folder containing the **hsa\_ncRNAs.fa** file, using this command:

cd "/mnt/path/to/Bowtie index of other ncRNA reference"

1. Check what types of ncRNAs are present in this file, using this command:

grep "^>" hsa\_ncRNAs.fa | awk -F"transcript\_biotype:" '{print $2}' | awk '{print $1}' | sort | uniq

* 1. This command produced this output:  
     Mt\_rRNA

Mt\_tRNA

TEC

lncRNA

miRNA

misc\_RNA

rRNA

retained\_intron

ribozyme

sRNA

scRNA

scaRNA

snRNA

snoRNA

vault\_RNA

1. This shows that tRFs and piRNAs are not present in the **hsa\_ncRNAs.fa** file. However, miRNAs are present.
   1. Since a miRNA reference has already been obtained from miRBase, the miRNAs from the **hsa\_ncRNAs.fa** file will need to be removed.
   2. Other ncRNA types that are unlikely to be present in human plasma are **Mt\_rRNA**, **Mt\_tRNA**, **TEC**, **retained\_intron**, **ribozyme**, **scRNA**, and **scaRNA**, so these also need to be removed.
   3. The category **sRNA** (small RNA) is broad and not informative, so it should also be removed.
   4. The category **misc\_RNA** is also broad. Further inspection of all entries classified as **misc\_RNA** can be done using this command:

grep "^>" hsa\_other\_ncRNAs.fa | grep "misc\_RNA"

* + 1. This command showed that RNAs in this category have one of these descriptions: **Y\_RNA**, **7SK RNA**, **pseudogene**, **vertebrate telomerase RNA**, or **metazoan signal recognition particle RNA**.
       1. **Y RNA** fragments can be found in human plasma so this should remain in the reference file.
       2. **7SK RNA** is found in nuclei and not in human plasma so it should be removed from the reference file.
       3. **Pseudogenes** are copies of genes that have lost their coding potential due to mutations. They are unlikely to be found in human plasma, so they should be removed from the reference file.
       4. **Vertebrate telomerase RNA** and **Metazoan signal recognition particle (SRP) RNA** are unlikely to be found in human plasma so they should be removed from the reference file.
  1. Fragments of **rRNA** and **lncRNA** may be present in human plasma so these should remain in the reference file.

1. Use this command to create an **hsa\_other\_ncRNAs.fa** file, which contains only the relevant ncRNAs:

awk '/^>/{if($0 !~ /miRNA|Mt\_rRNA|Mt\_tRNA|TEC|retained\_intron|ribozyme|sRNA|scRNA|scaRNA|7SK RNA|pseudogene|Vertebrate telomerase RNA|Metazoan signal recognition particle RNA/) {print; getline; print} else {getline}}' hsa\_ncRNAs.fa > hsa\_other\_ncRNAs.fa

1. Confirm that the removal of the unwanted ncRNA types worked, using this command (it will show which types of ncRNAs have been retained):

grep "^>" hsa\_other\_ncRNAs.fa | awk -F"transcript\_biotype:" '{print $2}' | awk '{print $1}' | sort | uniq

* 1. This command should produce this output (note that **Y RNA** is within **misc\_RNA**):  
     lncRNA

misc\_RNA

rRNA

snRNA

snoRNA

vault\_RNA

## Bowtie installation instructions for Ubuntu

1. Open an Ubuntu terminal.
2. Type the following command to update your package list:

sudo apt-get update

* 1. If it asks you a [Y/n] question, type **Y** and press enter.

1. Install Bowtie with this command:

sudo apt-get install bowtie

## Build a Bowtie index of the reference files to map to

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the **mature\_hsa\_miRNAs\_DNA.fa** file that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie index of miRNA reference"

1. Index the reference sequences using Bowtie with this command:

bowtie-build mature\_hsa\_miRNAs.fa mature\_hsa\_miRNAs

* 1. This uses the above .fa file to create six index files prefixed with “mature\_miRNAs”, which will be used by Bowtie later. They are saved in the current directory.
     1. **mature\_miRNAs.1.ebwt** to **mature\_miRNAs.4.ebwt**: These files contain the forward index, which Bowtie uses to map reads against the reference sequences.
     2. **mature\_miRNAs.rev.1.ebwt** and **mature\_miRNAs.rev.2.ebwt**: These files contain the reverse index, which Bowtie uses to align reads to the reverse complement of the reference sequences.

1. Navigate to the folder containing the **tRFs.fa** file that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie index of tRF reference"

1. Index the reference sequences using Bowtie with this command:

bowtie-build tRFs.fa tRFs

1. Navigate to the folder containing the **piRNA\_gold\_standard\_set.fa** file that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie index of piRNA reference"

1. Index the reference sequences using Bowtie with this command:

bowtie-build piRNA\_gold\_standard\_set.fa piRNA\_gold\_standard\_set

1. Navigate to the folder containing the **hsa\_other\_ncRNAs.fa** file that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie index of other ncRNA reference"

1. Index the reference sequences using Bowtie with this command:

bowtie-build hsa\_other\_ncRNAs.fa hsa\_other\_ncRNAs

## Alignment 1: Align the reads to a miRNA reference using Bowtie

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the .fastq.gz files that will be used for mapping, using this command:

cd "/mnt/path/to/Cutadapt output"

1. Place the **run\_bowtie\_miRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_bowtie\_miRNA.sh

1. Run the script with this command:

./run\_bowtie\_miRNA.sh

1. This script will produce 3 output files per .fastq.gz file processed. All files will be inside a folder called **Bowtie miRNA alignment output**, which is found next to the **Cutadapt output** folder. All files will have the filename of the processed .fastq.gz file, followed by a new suffix as follows:
   1. The **miRNA\_unmapped.fastq.gz** file contains the reads that did not map to miRNAs.
   2. The **miRNA\_alignments.sam** file contains the alignments to miRNAs.
   3. The **miRNA\_bowtie.log** file contains information about the alignment to miRNAs.

## Alignment 2: Align the reads to a tRF reference using Bowtie

1. **NB:** This step must be done AFTER miRNA alignment has finished, since it will only use the reads that did not map to miRNAs, and not the original reads.
2. Open an Ubuntu terminal.
3. Navigate to the folder containing the .fastq.gz files that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie miRNA alignment output"

1. Place the **run\_bowtie\_tRF.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_bowtie\_tRF.sh

1. Run the script with this command:

./run\_bowtie\_tRF.sh

1. This script will produce 3 output files per .fastq.gz file processed. All files will be inside a folder called **Bowtie tRF alignment output**, which is found next to the **Cutadapt output** folder. All files will have the filename of the processed .fastq.gz file, followed by a new suffix as follows:
   1. The **tRF\_unmapped.fastq.gz** file contains the reads that did not map to miRNAs.
   2. The **tRF\_alignments.sam** file contains the alignments to miRNAs.
   3. The **tRF\_bowtie.log** file contains information about the alignment to miRNAs.

## Alignment 3: Align the reads to a piRNA reference using Bowtie

1. **NB:** This step must be done AFTER tRF alignment has finished, since it will only use the reads that did not map to miRNAs or tRFs, and not the original reads.
2. Open an Ubuntu terminal.
3. Navigate to the folder containing the .fastq.gz files that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie tRF alignment output"

1. Place the **run\_bowtie\_piRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_bowtie\_piRNA.sh

1. Run the script with this command:

./run\_bowtie\_piRNA.sh

1. This script will produce 3 output files per .fastq.gz file processed. All files will be inside a folder called **Bowtie piRNA alignment output**, which is found next to the **Cutadapt output** folder. All files will have the filename of the processed .fastq.gz file, followed by a new suffix as follows:
   1. The **piRNA\_unmapped.fastq.gz** file contains the reads that did not map to piRNAs.
   2. The **piRNA\_alignments.sam** file contains the alignments to piRNAs.
   3. The **piRNA\_bowtie.log** file contains information about the alignment to piRNAs.

## Alignment 4: Align the reads to an “other ncRNA” reference using Bowtie

1. **NB:** This step must be done AFTER piRNA alignment has finished, since it will only use the reads that did not map to miRNAs, tRFs, or piRNAs, and not the original reads.
2. Open an Ubuntu terminal.
3. Navigate to the folder containing the .fastq.gz files that will be used for mapping, using this command:

cd "/mnt/path/to/Bowtie piRNA alignment output"

1. Place the **run\_bowtie\_other\_ncRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_bowtie\_other\_ncRNA.sh

1. Run the script with this command:

./run\_bowtie\_other\_ncRNA.sh

1. This script will produce 3 output files per .fastq.gz file processed. All files will be inside a folder called **Bowtie other ncRNA alignment output**, which is found next to the **Cutadapt output** folder. All files will have the filename of the processed .fastq.gz file, followed by a new suffix as follows:
   1. The **other\_ncRNA\_unmapped.fastq.gz** file contains the reads that did not map to “other ncRNAs”.
   2. The **other\_ncRNA\_alignments.sam** file contains the alignments to “other ncRNAs”.
   3. The **other\_ncRNA\_bowtie.log** file contains information about the alignment to “other ncRNAs”.

## Summarise alignment statistics

1. Run the MATLAB function **extract\_bowtie\_alignment\_stats.m** with this command (the current directory should be the folder that contains the **Bowtie miRNA alignment output** folder):

exctract\_bowtie\_alignment\_stats

1. One Excel file called **Bowtie\_alignment\_stats.xlsx** will be saved in the current directory. You can use this file to create useful figures summarising the Bowtie results.
2. Open an Ubuntu terminal.
3. Navigate to the folder containing the **Bowtie miRNA alignment output** folder, using this command:

cd "/mnt/path/to/folder"

1. Place the **get\_percent\_identity.sh** and the **calculate\_percent\_identity\_mean.sh** shell scripts in this folder.
2. Make sure the first script is executable with this command:

chmod +x get\_percent\_identity.sh

1. Run the first script with this command:

./get\_percent\_identity.sh

1. This script will produce one output file per .sam file processed (4 .sam files per sample so 4 output files per sample). All files will be inside a folder called **Percent identity files**.
   1. The **[sampleID]\_miRNA\_percent\_identity.txt** file contains the read IDs and their corresponding percent identity when the reads were aligned to a miRNA reference. This information is taken from the .sam file in the **Bowtie miRNA alignment output** folder.
   2. The **[sampleID]\_tRF\_percent\_identity.txt** file contains the read IDs and their corresponding percent identity when the reads were aligned to a tRF reference. This information is taken from the .sam file in the **Bowtie tRF alignment output** folder.
   3. The **[sampleID]\_piRNA\_percent\_identity.txt** file contains the read IDs and their corresponding percent identity when the reads were aligned to a piRNA reference. This information is taken from the .sam file in the **Bowtie piRNA alignment output** folder.
   4. The **[sampleID]\_other\_ncRNA\_percent\_identity.txt** file contains the read IDs and their corresponding percent identity when the reads were aligned to an “other ncRNA” reference. This information is taken from the .sam file in the **Bowtie other ncRNA alignment output** folder.
2. Make sure the second script is executable with this command:

chmod +x calculate\_percent\_identity\_means.sh

1. Run the second script with this command:

./calculate\_percent\_identity\_means.sh

1. This script will save 4 .csv files inside the **Percent identity files** folder:
   1. The **miRNA\_percent\_identity\_means.csv** file contains the mean percent identity value from reads that aligned to miRNAs, for each sample.
   2. The **tRF\_percent\_identity\_means.csv** file contains the mean percent identity value from reads that aligned to tRFs, for each sample.
   3. The **piRNA\_percent\_identity\_means.csv** file contains the mean percent identity value from reads that aligned to piRNAs, for each sample.
   4. The **other\_ncRNA\_percent\_identity\_means.csv** file contains the mean percent identity value from reads that aligned to “other ncRNAs”, for each sample.
2. You can use the data in these .csv files to plot useful figures summarising the results.

# Quantify miRNAs, tRFs, piRNAs, and “other ncRNAs” using featureCounts

## Create a GTF annotation file for miRNAs

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the **mature\_hsa\_miRNAs\_DNA.fa** file, using this command:

cd "/mnt/path/to/Bowtie index of miRNA reference"

1. Place the **FASTA\_to\_GTF\_miRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x FASTA\_to\_GTF\_miRNA.sh

1. Run the script with this command:

./FASTA\_to\_GTF\_miRNA.sh

1. This script will produce a file called **mature\_hsa\_miRNAs.gtf** in the **Bowtie index of miRNA reference** folder. It will have 9 columns (fields), representing the standard GTF format:
   1. **seqname:** miRNA name
   2. **source:** miRBase
   3. **feature:** transcript
   4. **start:** 1
   5. **end:** end position of the feature (e.g., 22 if the miRNA is 22 nt long)
   6. **score:** . [stands for “not applicable”]
   7. **strand:** +
   8. **frame:** .
   9. **attribute:** transcript\_id “hsa-miR-16-5p” [as an example]

## Create a GTF annotation file for tRFs

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the **tRFs.fa** file, using this command:

cd "/mnt/path/to/Bowtie index of tRF reference"

1. Place the **FASTA\_to\_GTF\_tRF.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x FASTA\_to\_GTF\_tRF.sh

1. Run the script with this command:

./FASTA\_to\_GTF\_tRF.sh

1. This script will produce a file called **tRFs.gtf** in the **Bowtie index of tRF reference** folder. It will have 9 columns (fields), representing the standard GTF format:
   1. **seqname:** tRF name
   2. **source:** tRFdb
   3. **feature:** transcript
   4. **start:** 1
   5. **end:** end position of the feature (e.g., 18 if the tRF is 18 nt long)
   6. **score:** . [stands for “not applicable”]
   7. **strand:** +
   8. **frame:** .
   9. **attribute:** transcript\_id “5001a” [as an example]

## Create a GTF annotation file for piRNAs

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the **piRNA\_gold\_standard\_set.fa** file, using this command:

cd "/mnt/path/to/Bowtie index of piRNA reference"

1. Place the **FASTA\_to\_GTF\_piRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x FASTA\_to\_GTF\_piRNA.sh

1. Run the script with this command:

./FASTA\_to\_GTF\_piRNA.sh

1. This script will produce a file called **piRNA\_gold\_standard\_set.gtf** in the **Bowtie index of piRNA reference** folder. It will have 9 columns (fields), representing the standard GTF format:
   1. **seqname:** piRNA name
   2. **source:** piRBase
   3. **feature:** transcript
   4. **start:** 1
   5. **end:** end position of the feature (e.g., 25 if the piRNA is 25 nt long)
   6. **score:** . [stands for “not applicable”]
   7. **strand:** +
   8. **frame:** .
   9. **attribute:** transcript\_id “piR-hsa-9” [as an example]

## Create a GTF annotation file for “other ncRNAs”

1. Open an Ubuntu terminal.
2. Navigate to the folder containing the **has\_other\_ncRNAs.fa** file, using this command:

cd "/mnt/path/to/Bowtie index of other ncRNA reference"

1. Place the **FASTA\_to\_GTF\_other\_ncNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x FASTA\_to\_GTF\_other\_ncRNA.sh

1. Run the script with this command:

./FASTA\_to\_GTF\_other\_ncRNA.sh

1. This script will produce a file called **hsa\_other\_ncRNA.gtf** in the **Bowtie index of other ncRNA reference** folder. It will have 9 columns (fields), representing the standard GTF format:
   1. **seqname:** ncRNA name
   2. **source:** ensembl
   3. **feature:** transcript
   4. **start:** 1
   5. **end:** end position of the feature (e.g., 30 if the ncRNA is 30 nt long)
   6. **score:** . [stands for “not applicable”]
   7. **strand:** +
   8. **frame:** .
   9. **attribute:** transcript\_id “ABCD” [as an example]

## Install featureCounts

1. Create a new folder called **featureCounts**, next to the **Cutadapt output** folder.
2. Open an Ubuntu terminal.
3. Navigate to the new folder, using this command:

cd "/mnt/path/to/featureCounts"

1. Run these commands to install dependencies:

sudo apt update

sudo apt install build-essential

1. Run this command to download Subread (change the version numbers to the latest version, which can be found at <https://subread.sourceforge.net/>):

wget https://sourceforge.net/projects/subread/files/subread-2.0.7/subread-2.0.7-Linux-x86\_64.tar.gz

1. Extract the files with this command:

tar -xzvf subread-2.0.7-Linux-x86\_64.tar.gz

1. Move the **bin** directory (this is inside the subread folder that was just extracted) to a directory that is included in your path (to check which directories are included in your path, use the command echo $PATH), using this command:

sudo mv subread-2.0.7-Linux-x86\_64/bin/\* /usr/local/bin/

1. Verify installation with this command (it should print the version):

featureCounts -v

1. Delete the **featureCounts** folder.

## Quantify miRNAs using featureCounts

1. Open an Ubuntu terminal.
2. Navigate to the **Bowtie miRNA alignment output** folder containing the .sam files to be analysed, using this command:

cd "/mnt/path/to/Bowtie miRNA alignment output"

1. Place the **run\_featureCounts\_miRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_featureCounts\_miRNA.sh

1. Run the script with this command:

./run\_featureCounts\_miRNA.sh

1. This script will produce two output files per .sam file analysed, and they will be saved in a new folder called **featureCounts output miRNA**, which will be next to the **Bowtie miRNA alignment output** folder. The output files will have the same filename as the input .sam file, but with a suffix appended:
   1. A **miRNA\_counts.txt** file which contains the miRNA counts for each miRNA found in the GTF file.
   2. A **miRNA\_counts.summary** file which contains information about the number of reads that mapped to miRNAs and the number of reads that were unmapped (this information is taken from the .sam file). This should contain the same information that is found in the .log file created by the Bowtie alignment earlier.

## Quantify tRFs using featureCounts

1. Open an Ubuntu terminal.
2. Navigate to the **Bowtie tRF alignment output** folder containing the .sam files to be analysed, using this command:

cd "/mnt/path/to/Bowtie tRF alignment output"

1. Place the **run\_featureCounts\_tRF.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_featureCounts\_tRF.sh

1. Run the script with this command:

./run\_featureCounts\_tRF.sh

1. This script will produce two output files per .sam file analysed, and they will be saved in a new folder called **featureCounts output tRF**, which will be next to the **Bowtie tRF alignment output** folder. The output files will have the same filename as the input .sam file, but with a suffix appended:
   1. A **tRF\_counts.txt** file which contains the tRF counts for each tRF found in the GTF file.
   2. A **tRF\_counts.summary** file which contains information about the number of reads that mapped to tRFs and the number of reads that were unmapped (this information is taken from the .sam file). This should contain the same information that is found in the .log file created by the Bowtie alignment earlier.

## Quantify piRNAs using featureCounts

1. Open an Ubuntu terminal.
2. Navigate to the **Bowtie piRNA alignment output** folder containing the .sam files to be analysed, using this command:

cd "/mnt/path/to/Bowtie piRNA alignment output"

1. Place the **run\_featureCounts\_piRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_featureCounts\_piRNA.sh

1. Run the script with this command:

./run\_featureCounts\_piRNA.sh

1. This script will produce two output files per .sam file analysed, and they will be saved in a new folder called **featureCounts output piRNA**, which will be next to the **Bowtie piRNA alignment output** folder. The output files will have the same filename as the input .sam file, but with a suffix appended:
   1. A **piRNA\_counts.txt** file which contains the piRNA counts for each piRNA found in the GTF file.
   2. A **piRNA\_counts.summary** file which contains information about the number of reads that mapped to piRNA and the number of reads that were unmapped (this information is taken from the .sam file). This should contain the same information that is found in the .log file created by the Bowtie alignment earlier.

## Quantify “other ncRNAs” using featureCounts

1. Open an Ubuntu terminal.
2. Navigate to the **Bowtie other ncRNA alignment output** folder containing the .sam files to be analysed, using this command:

cd "/mnt/path/to/Bowtie other ncRNA alignment output"

1. Place the **run\_featureCounts\_other\_ncRNA.sh** shell script in this folder.
2. Make sure the script is executable with this command:

chmod +x run\_featureCounts\_other\_ncRNA.sh

1. Run the script with this command:

./run\_featureCounts\_other\_ncRNA.sh

1. This script will produce two output files per .sam file analysed, and they will be saved in a new folder called **featureCounts output other ncRNA**, which will be next to the **Bowtie other ncRNA alignment output** folder. The output files will have the same filename as the input .sam file, but with a suffix appended:
   1. An **other\_ncRNA\_counts.txt** file which contains the “other ncRNA” counts for each ncRNA found in the GTF file.
   2. An **other\_ncRNA\_counts.summary** file which contains information about the number of reads that mapped to “other ncRNA” and the number of reads that were unmapped (this information is taken from the .sam file). This should contain the same information that is found in the .log file created by the Bowtie alignment earlier.

## Consolidate featureCounts results

1. Run the MATLAB function **consolidate\_RNA\_counts.m** with this command (the current directory should be the folder that contains the **featureCounts output miRNA** folder):

Consolidate\_RNA\_counts

1. Four Excel files will be saved in the current directory:
   1. **miRNA\_counts\_all\_samples.xlsx** contains the miRNA counts.
   2. **tRF\_counts\_all\_samples.xlsx** contains the tRF counts.
   3. **piRNA\_counts\_all\_samples.xlsx** contains the piRNA counts.
   4. **other\_ncRNA\_counts\_all\_samples.xlsx** contains the “other ncRNA” counts.

# Filter and normalise miRNA, tRF, piRNA, and “other ncRNA” counts

Remove non-expressed miRNAs, tRFs, piRNAs, and other ncRNAs from count data

1. Open MATLAB.
2. Navigate to the folder containing the **miRNA\_counts\_all\_samples.xlsx** file.
3. Make sure the **remove\_non\_expressed\_RNAs.m** file exists. It can be stored anywhere on the computer, not necessarily in the same folder as the **miRNA\_counts\_all\_samples.xlsx** file.
4. Add this .m file to the MATLAB path (you can open the file in MATLAB, then right-click on the file tab and select **Add to path**).
5. Run this MATLAB function with this command:

remove\_non\_expressed\_RNAs

1. This function will create 8 files in the current directory:
   1. The (1) **Non-expressed\_miRNAs.xlsx**, (2) **Non-expressed\_tRFs.xlsx**, (3) **Non-expressed\_piRNAs.xlsx**, and (4) **Non-expressed\_other\_ncRNAs.xlsx** files list RNAs that had zero counts in all samples (i.e., these RNAs were not expressed).
      1. **NB:** If any of these files are empty, then it means that all RNAs (of that type) had a count >0 in AT LEAST ONE sample.
   2. The (5) **miRNA\_counts\_filtered.xlsx**, (6) **tRF\_counts\_filtered.xlsx**, (7) **piRNA\_counts\_filtered.xlsx**, and (8) **other\_ncRNA\_counts\_filtered.xlsx** files contain the counts for RNAs that passed the filtering criteria (i.e., have a count >0 in at least one sample).

## Notes

* The goal is to focus the rest of the analysis on miRNAs, tRFs, and piRNAs only. The “other ncRNAs” will not be analysed further, as they are not informative for the purposes of this study. The only reason for aligning to them was to check if other ncRNA fragments are present in the samples (as would be expected), and to remove them from the remaining reads in case the remaining reads will be used for novel ncRNA identification in the future.

Extra filtering of piRNA counts

Upon inspection of the current piRNA list, most piRNAs have 0 counts in most samples. This is causing issues later when normalising the piRNA counts, because it results in negative VST-normalised counts which do not make biological sense, because there cannot be negative expression of a piRNA. Therefore, a further filtering step is needed.

1. Open MATLAB.
2. Navigate to the folder containing the p**iRNA\_counts\_filtered.xlsx** file.
3. Make sure the **filter\_low\_piRNAs.m** file exists. It can be stored anywhere on the computer, not necessarily in the same folder as the **miRNA\_counts\_filtered.xlsx** file.
4. Add this .m file to the MATLAB path (you can open the file in MATLAB, then right-click on the file tab and select **Add to path**).
5. Run this MATLAB function with this command:

filter\_low\_piRNAs

1. This function will create 2 files in the current directory:
   1. The **piRNA\_counts\_filtered\_strict.xlsx** file contains piRNAs that meet the filtering criteria (i.e., robustly expressed in a sufficient fraction of samples).
   2. The **Lowly\_expressed\_piRNAs.xlsx** file lists piRNAs filtered out for having low or sporadic expression.

Normalise all expressed miRNAs, tRFS, and highly expressed piRNAs using the VST method of DESeq2 in R

1. Open RStudio.
2. Navigate to the folder containing the **miRNA\_counts\_filtered.xlsx** file outputted by the MATLAB function above, using this command:

setwd("path/to/folder")

1. Make sure the **normalise\_RNA\_counts\_DESeq2.R** file exists. It can be stored anywhere on the computer, not necessarily in the same folder as the **miRNA\_counts\_filtered.xlsx** file.
2. Run the script.
3. This script will save 3 files in the current folder, containing the RNA counts after VST normalisation by DESeq2:
   1. **VST\_normalised\_miRNA\_counts.csv**
   2. **VST\_normalised\_tRF\_counts.csv**
   3. **VST\_normalised\_piRNA\_counts.csv**

### Remove lowly-expressed miRNAs, tRFs, and piRNAs from the VST-normalised counts

1. Open MATLAB.
2. Navigate to the folder containing the **VST\_normalised\_miRNA\_counts.csv** file outputted by the MATLAB function above, using the address bar.
3. Make sure the **remove\_low\_RNAs\_VST.m** file exists. It can be stored anywhere on the computer, not necessarily in the same folder as the **VST\_normalised\_miRNA\_counts.csv** file.
4. Add this .m file to the MATLAB path (you can open the file in MATLAB, then right-click on the file tab and select **Add to path**).
5. Run this MATLAB function with this command:

remove\_low\_RNAs\_VST

1. This function will create 6 files in the current directory:
   1. The **Lowly-expressed\_VST\_miRNAs.xlsx**, **Lowly-expressed\_VST\_tRFs.xlsx**, and **Lowly-expressed\_VST\_piRNAs.xlsx** files, which contain miRNAs, tRFs, and piRNAs, below the lower 25% quantile of median expression.
      1. **NB:** If any of these files are empty then it means that there were no lowly-expressed ncRNAs of that ncRNA type, because they are all highly expressed.
   2. The **Highly-expressed\_VST\_miRNA\_counts.xlsx**, **Highly-expressed\_VST\_tRF\_counts.xlsx**, and **Highly-expressed\_VST\_piRNA\_counts.xlsx** files, which contain VST-normalised counts of miRNAs, tRFs, and piRNAs above the lower 25% quantile of median expression.