1. RNApipe
   - 1.1 Introduction
   - 1.2 Installation
     - 1.2.1 miniconda3
     - 1.2.2 RNApipe
   - 1.3 Input files
     - 1.3.1 Annotation.gff and genome.fa
     - 1.3.2 Input .fastq files
     - 1.3.3 Sample sheet and configuration file
     - 1.3.4 Download speices database in kobas
   - 1.4 Output files
   - 1.5 Tool and parameters
2. Workflow configuration
   - 2.1 Default analysis softwares:
   - 2.2 Default parameters
   - 2.3 Hardware configuration
   - 2.4 Reference
3. Example workflow
   - 3.1 Setup
   - 3.2 Retrieve and prepare input files
     - 3.2.1 Annotation and genome files
     - 3.2.2 .fastq files
     - 3.2.3 Sample sheet and configuration file
     - 3.2.4 Download speices database in koabs
   - 3.3 Running the workflow
   - 3.4 Results

1. RNApipe

1.1 Introduction

RNApipe is an automated analysis workflow for RNA-seq data from all reference species. It includes basic processing steps such as quality control and preprocessing of raw data, genome alignment, quantification, and differential expression analysis. In addition, RNApipe also supports functional annotation analysis of differential genes in most species. RNApipe is based on the Snakemake workflow management system and the Conda environment manager to complete the automatic installation of all dependencies and the automatic running of the program. The source code of RNApipe is open source and available on Github (https://github.com/ywu019/RNApipe). Installation and basic usage are described below.

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

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

1.2 Installation

1.2.1 miniconda3

Since RNApipe is based on the workflow management system snakemake, Snakemake can download all the necessary dependencies through conda. First of all, we strongly recommend installing miniconda3 with python3. After downloading the version of miniconda3 suitable for your linux system, execute the downloaded bash file and follow the instructions given.

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
chmod 777 Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh
```

Chinese users may need to change mirrors to ensure fast and smooth software installation. Here we add Tsinghua mirror source with the following commands.

```
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/bioconda/
```

1.2.2 RNApipe

RNApipe is required for using the workflow. The latest release is available on our GitHub page. To run the workflow, we recommend that you download RNApipe into your project directory. Then download and unzip the latest version of RNApipe by entering the following commands:

```
wget https://github.com/ywu019/RNApipe/archive/refs/heads/main.zip
unzip main.zip
mv RNApipe-main/ RNApipe; rm main.zip;cd RNApipe;
```

Note: RNApipe requires a good network connection to install. If the above code doesn't work, try downloading RNApipe's zip package directly from Github.

RNApipe now resides in a subdirectory of the project directory. Create and activate an RNApipe environment with the following commands:

```
conda env create -n RNApipe -f envs/envs.yaml
conda activate RNApipe
```

Now RNApipe needed tools have been installed in an environment called RNApipe .

Note: The environment must be installed using a server node with a network. When some computing nodes have no network, the smooth installation of the environment will be affected.

1.3 Input files

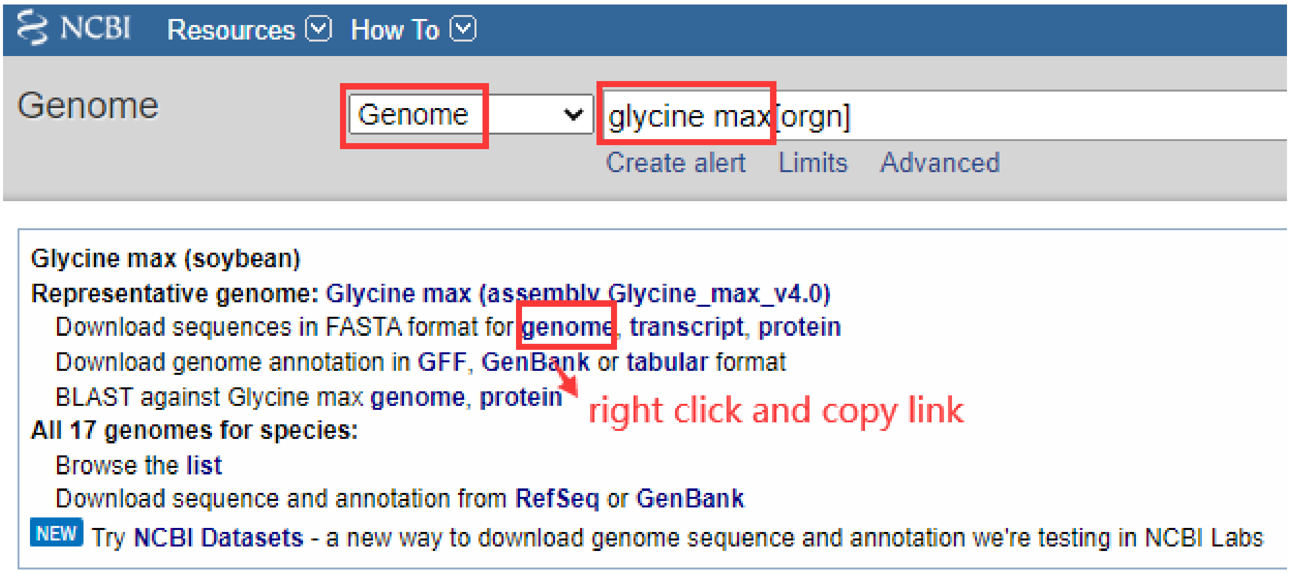
Several input files are required in order to run the workflow, a genome file (.fa), an annotation file (.gtf) and compressed sequencing files (.fastq.gz).

Input files name	Description
genome.fa	user-provided genome file containing the genome sequence
annotation.gtf	user-provided annotation file with genomic features
<condition>_<replicate>.fastq.gz	user-provided compressed sequencing files
samples.tsv	sample file describing the relation between the input fastq files
config.yaml	configuration file to customize the workflow

1.3.1 Annotation.gff and genome.fa

We recommend retrieving the genome and annotation files of the species from the National Center for Biotechnology Information (NCBI). Take the example of downloading soybean genome files and annotation files for demonstration, so that users can obtain genome files of other species.

Users can enter the corresponding species name in the GENOME search box of NCBI, right click the 'genome' and copy the link.



Glycine max (soybean)

Representative genome: Glycine max (assembly Glycine_max_v4.0)

Download sequences in FASTA format for **genome**, **transcript**, **protein**

Download genome annotation in GFF, **GenBank** or **tabular** format

BLAST against Glycine max **genome**, **protein**

right click and copy link

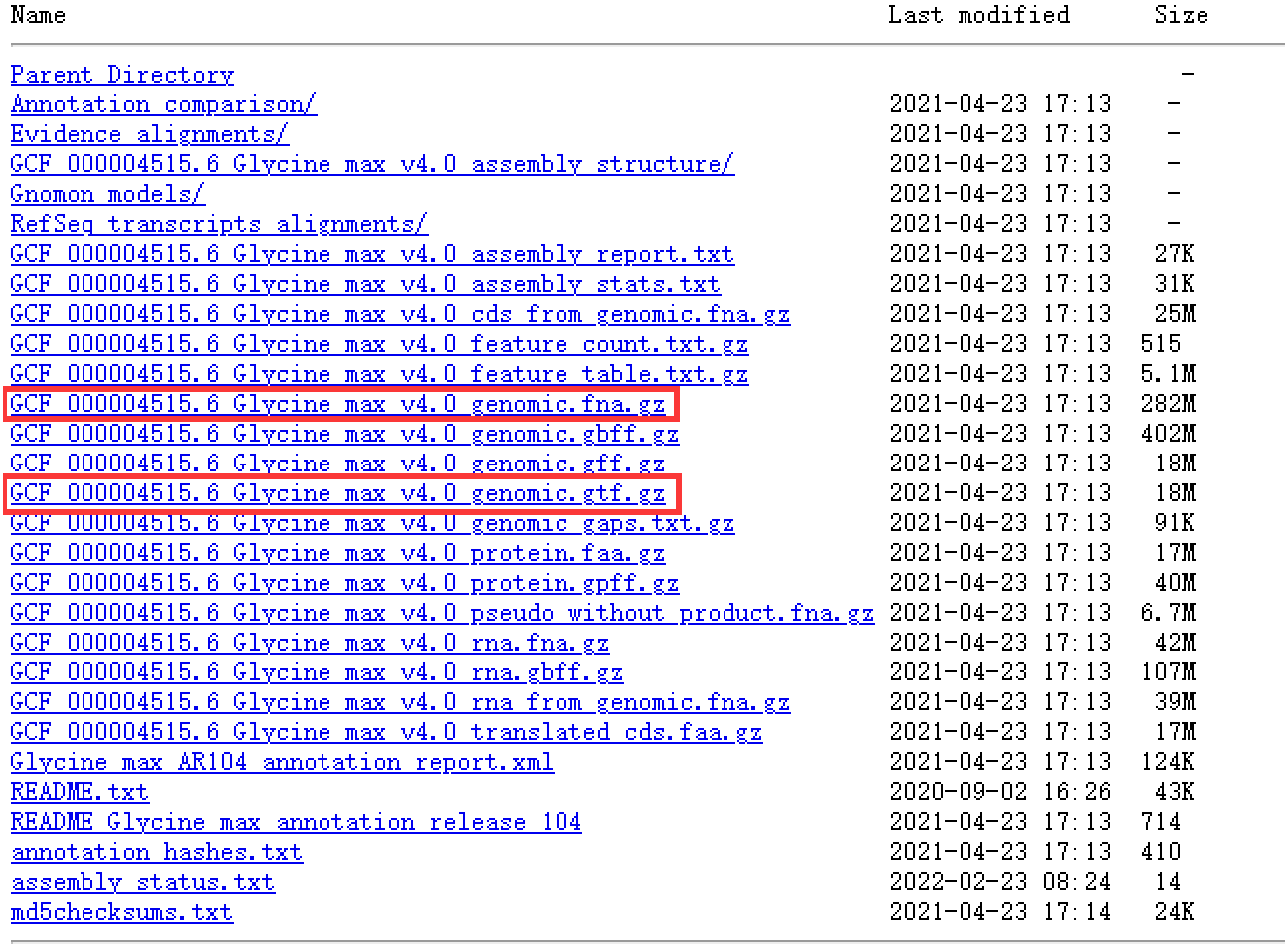
All 17 genomes for species:

Browse the **list**

Download sequence and annotation from **RefSeq** or **GenBank**

NEW Try **NCBI Datasets** - a new way to download genome sequence and annotation we're testing in NCBI Labs

Get the link https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/515/GCF_000004515.6_Glycine_max_v4.0/GCF_000004515.6_Glycine_max_v4.0_genomic.fna.gz ,Modify to the previous directory: https: //ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/515/GCF_000004515.6_Glycine_max_v4.0/, then open it in a new browser window and get the following interface.



Name	Last modified	Size
Parent Directory		-
Annotation_comparison/	2021-04-23 17:13	-
Evidence_alignments/	2021-04-23 17:13	-
GCF_000004515.6_Glycine_max_v4.0_assembly_structure/	2021-04-23 17:13	-
Gnomon_models/	2021-04-23 17:13	-
RefSeq_transcripts_alignments/	2021-04-23 17:13	-
GCF_000004515.6_Glycine_max_v4.0_assembly_report.txt	2021-04-23 17:13	27K
GCF_000004515.6_Glycine_max_v4.0_assembly_stats.txt	2021-04-23 17:13	31K
GCF_000004515.6_Glycine_max_v4.0_cds_from_genomic.fna.gz	2021-04-23 17:13	25M
GCF_000004515.6_Glycine_max_v4.0_feature_count.txt.gz	2021-04-23 17:13	515
GCF_000004515.6_Glycine_max_v4.0_feature_table.txt.gz	2021-04-23 17:13	5.1M
GCF_000004515.6_Glycine_max_v4.0_genomic.fna.gz	2021-04-23 17:13	282M
GCF_000004515.6_Glycine_max_v4.0_genomic.gbff.gz	2021-04-23 17:13	402M
GCF_000004515.6_Glycine_max_v4.0_genomic.gff.gz	2021-04-23 17:13	18M
GCF_000004515.6_Glycine_max_v4.0_genomic.gtf.gz	2021-04-23 17:13	18M
GCF_000004515.6_Glycine_max_v4.0_genomic_gaps.txt.gz	2021-04-23 17:13	91K
GCF_000004515.6_Glycine_max_v4.0_protein.faa.gz	2021-04-23 17:13	17M
GCF_000004515.6_Glycine_max_v4.0_protein.gpff.gz	2021-04-23 17:13	40M
GCF_000004515.6_Glycine_max_v4.0_pseudo_without_product.fna.gz	2021-04-23 17:13	6.7M
GCF_000004515.6_Glycine_max_v4.0_rna.fna.gz	2021-04-23 17:13	42M
GCF_000004515.6_Glycine_max_v4.0_rna.gbff.gz	2021-04-23 17:13	107M
GCF_000004515.6_Glycine_max_v4.0_rna_from_genomic.fna.gz	2021-04-23 17:13	39M
GCF_000004515.6_Glycine_max_v4.0_translated_cds.faa.gz	2021-04-23 17:13	17M
Glycine_max_AR104_annotation_report.xml	2021-04-23 17:13	124K
README.txt	2020-09-02 16:26	43K
README_Glycine_max_annotation_release_104	2021-04-23 17:13	714
annotation_hashes.txt	2021-04-23 17:13	410
assembly_status.txt	2022-02-23 08:24	14
md5checksums.txt	2021-04-23 17:14	24K

On this page, Users can click the link marked above to download in the Windows system, and then transfer to the shell. Alternatively, we recommend right-clicking and copying the link marked above and downloading directly in the shell.

```
wget https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/515/GCF_000004515.6_Glycine_max_v4.0/GCF_000004515.6_Glycine_max_v4.0_genomic.fna.gz
wget https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/515/GCF_000004515.6_Glycine_max_v4.0/GCF_000004515.6_Glycine_max_v4.0_genomic.gtf.gz
```

Please ensure that you move files into a folder called ref:

```
mkdir ref
cp GCF_000004515.6_Glycine_max_v4.0_genomic.fna.gz ref/
cp GCF_000004515.6_Glycine_max_v4.0_genomic.gtf.gz ref/
```

Warning: if you use custom annotation files, ensure that you adhere to the gtf standard. Wrongly formatted files are likely to cause problems with downstream tools.

1.3.2 Input .fastq files

These are the input files provided by the user. Both single end and paired end data is supported.

Note: Make sure to compress the file in .gz format. The naming format of fastq files should conform to the format of <condition>_<replicate>.fastq.gz. If the sequencing data is pairedend, the fastq files need to be named separately <condition>_<replicate>_R1.fastq.gz and <condition>_<replicate>_R2.fastq.gz

Please ensure that you move all input .fastq.gz files into a folder called fastq:

```
mkdir fastq
cp *.fastq.gz fastq/
```

1.3.3 Sample sheet and configuration file

To run RNApipe, you must provide a sample worksheet and configuration file. Templates for both files are available in the configs/ folder. Configuration files are used to allow users to easily customize certain settings, such as adapter sequences. The sample table is used to specify the relationship of the input .fastq file (conditions/replications, etc.). Users can modify the config.yaml and samples.tsv files in the configs directory according to their own experimental data.

Customize the config.yaml according to your experiment and data situation. It contains the following variables:

- PROJECT: #Project name
- READPATH: #Path of the sequencing file fastq.gz
- END: #Sequencing data type is SE (single-end) or PE (paired-end)
- OUTPUTPATH: #The path of the output file
- GENOME: # Genome file path
- ANNOTATION: #Annotation file path
- CONTROL: #Name of the control group, consistent with the condition in samples.tsv
- TREAT: #processing group name
- SPEICES_ABBREVIATION: Species abbreviation, query in speices_abbreviations

The user can also set the parameters and software tools of the RNApipe automated analysis workflow through the following variables.

- TRIM: fastp # [fastp, cutadapt]
- ADAPTER: AGATCGGAAGAGC # for SE_cutadapt or fastp. #ADAPTER_a: AGATCGGAAGAGC #adapter_A: GCTCTTCCGATCT for PE_cutadapt
- ALIGN: hisat2 # [hisat2, STAR]
- BIGWIG: ["no"] # [yes, no]

- DEA: DESeq2 # [DESeq2, edgeR, limma]
- QUANTIFY: featureCounts #[featureCounts, htseq]
- THREAD: "10" # The number of threads
- TIME: "5" # The latency for running individual rules programs in main.py

As seen in the samples.tsv template. Please ensure not to replace any tabulator symbols with spaces while changing this file. And you must rewrite this file to fit the previously downloaded .fastq.gz files.

condition	replicate
A	1
A	2
A	3
B	1
B	2
B	3

Note: This is just an example, please refer to our example-workflow for another example.

1.3.4 Download speices database in kobas

In order to perform functional annotation analysis of your species using Kobas, you will need to download the database for the respective species using the following command. Your species abbreviations can be looked up in the configs/speices_abbreviation.tsv file.

```
cd ./software/kobas-3.0/sqlite3/
wget ftp://ftp.cbi.pku.edu.cn/pub/KOBAS_3.0_DOWNLOAD/sqlite3/your_species_abbreviation.db
```

```
cd -
```

1.4 Output files

In the following tables all important output files of the workflow are listed.

Output directory	Important files
trim/selected software/	<condition>_<replicate>_trimmed.fastq.gz
align/selected software/	<condition>_<replicate>.sort.bam
quantify/selected software/	genome_count.tsv genome_tpm.tsv
DEA/selected software/	dea_control_treat.tsv deg_control_treat.tsv
annotation/go_kegg/	kobas_deg_control_treat.txt
annotation/gsea/	control.pdf treat.pdf *_detail_info
visualize/report/	report_quality_control_after_trimming.html report_align_count.html
visualize/exp/	correlation_control_treat.pdf cumulative_control_treat.pdf fvid_control_treat.pdf heatmap_control_treat.pdf PCA_control_treat.pdf.pdf violinplot_control_treat.pdf
visualize/DEA/select software	heatmap_control_treat.pdf valcano_control_treat.pdf
visualize/annotation/go_kegg	GO_control_treat_barplot.pdf GO_control_treat.pdf KEGG_control_treat.pdf

visualize/annotation/gsea	control.pdf treat.pdf

Note: Files create as intermediate steps of the workflow are omitted from this list. (e.g. .bam files)

Note: For more details about the output files, please refer to the analysis results.

1.5 Tool and parameters

Tool	Version	Description
python	3.7.12	Python interpreter
snakemake	7.0.0	Workflow management system
miniconda	4.11.0	Environment manager
fastqc	0.11.9	Quality control reporter
multiqc	1.12	Tools for aggregated reports
fastp	0.23.2	Adapter removal and quality trimming
cutadapt	3.7	Adapter removal and quality trimming
hisat2	2.2.1	Mapping reads
STAR	2.7.10a	Mapping reads
subread	2.0.1	Quantification of each gene
htseq	1.99.2	Quantification of each gene
qualimap	2.2.2d	Evaluate NGS mapping to a reference genome
smatools	1.15	Tools for working with bam files
deeptools	3.5.1	Tools for working with bam files
kobas	3.0	Tools for Pathway Enrichment
R	4.1	R interpreter
DESeq2	1.34.0	Differential expression analysis
edgeR	3.36.0	Differential expression analysis
limma	3.50.0	Differential expression analysis
clusterprofiler	4.2.0	R packages for pathway enrichment
r-tidyverse	1.3.1	R packages for data science
r-ggplot2	3.3.5	R packages for creating graphics
r-pheatmap	1.0.12	R packages for drawing clustered heatmaps

r-yaml	2.3.5	The libyaml YAML parser and emitter for R
r-ggpubr	0.4.0	R packages for creating graphics based on ggplot2
r-reshape2	1.4.4	R packages for reshaping a data frame
r-ggrepel	0.9.1	R packages for repelling overlapping text labels
r-dplyr	2.1.1	R packages for data science
r-cowplot	1.1.1	R packages for provides various theme features
r-scales	1.1.1	R packages for perceptual properties
r-FactoMineR	2.4	R packages for multivariate analysis
r-factoextra	1.0.7	R packages for multivariate analysis

2. Workflow configuration

This workflow allows customization of parameters and selection of software tools so that users can process different types of input data. Options that the user can set to easily customize the workflow are shown in the config.yaml file.

To explain what customizations are possible, we'll first look at the default workflow.

2.1 Default analysis softwares:

- TRIM: fastp # [fastp, cutadapt]
- ALIGN: hisat2 # [hisat2, STAR]
- DEA: DESeq2 # [DESeq2, edgeR, limma]
- QUANTIFY: featureCounts #[featureCounts, htseq]

According to the running speed, we choose relatively faster software as the default execution software of RNApipe, including: fastp, hisat2 and featureCounts. Users can view the running time of each program in the logs/ directory. In addition, in order to facilitate users to choose the corresponding software tools according to their own experimental purposes, we have listed some comparative descriptions of the same type of software tools in some literatures.

fastp can perform quality control, adapter trimming, and quality filtering. The tool is developed in C++ and has multithreading support. So it is 2-5 times faster than other FASTQ preprocessing tools like Cutadapt[1]. Cutadapt is a common adapter spinner that also provides some read filtering. So when the user's raw data is of good quality, cutadapt can be used to simply trim the adapter. When the raw data quality is low, we recommend using fastp for various preprocessing.

The aligner used in RNApipe is HISAT2, which has lower memory requirements and is faster than the STAR aligner [2]. But the sensitivity of hisat2 is relatively lower, while STAR is more sensitive, in addition, the unique mapping ratio of STAR is higher [3].

RNApipe integrates the current mainstream differential expression tools: DEseq2, edgeR and Limma. These tools have different models and advantages. Negative binomial modeling-based DESeq and edgeR have improved specificity and sensitivity, as well as good control over false-positive errors, with comparable performance. However, methods based on other distributions, such as limma, are advantageous and improve the modeling of genes expressed under one condition [4]. In addition, DESeq2 has higher recall and edgeR has higher precision, which means that edgeR is more conserved in the differential expression of reporter genes [2]. Users can choose any one of them in config.yaml for analysis according to their own experimental requirements.

The high computational efficiency of featureCounts is due to its ultra-fast feature search algorithm and efficient implementation entirely in the C programming language. However, htseq-count has its own advantages. This Python-based software was developed by scientists at the European Molecular Biology Laboratory who also participated in the development of many well-known differential expression tools (eg DESeq, DESeq2) across RNA-Seq analysis. Therefore, it appears to be more reliable and compatible with count-based differentially expressed RNA-Seq analysis.

2.2 Default parameters

- THREAD: "10" # The number of threads.
- TIME: "5" # The latency for running individual rules programs in main.py.

Note that some steps of the workflow require significant memory or time, depending on the size of the input data and hardware resources. We run each rule with 10 threads by default, you can adjust the **THREAD** variable in the config.yaml file according to your experimental data and hardware resources.

In order to process the data properly, we added a TIME variable to the main program to represent the delay. The default value is 5s, which can be modified by the user according to the actual running situation.

2.3 Hardware configuration

RNApipe can run smoothly on 4-core 8-thread laptops, 8-core 16-thread server login nodes, 14-core 28-thread server computing nodes, and 14-core 56-thread workstations. Users can adjust threads according to actual computing resources and core hardware resources.

2.4 Reference

- [1] Chen S, Zhou Y, Chen Y, Gu J. fastp: an ultra-fast all-in-one FASTQ preprocessor. Bioinformatics. 2018 Sep 1;34(17):i884-i890. doi: 10.1093/bioinformatics/bty560. PMID: 30423086; PMCID: PMC6129281.
- [2] Zhang X, Jonassen I. RASflow: an RNA-Seq analysis workflow with Snakemake. BMC Bioinformatics. 2020 Mar 18;21(1):110. doi: 10.1186/s12859-020-3433-x. PMID: 32183729; PMCID: PMC7079470.
- [3] Sahraeian SME, Mohiyuddin M, Sebra R, Tilgner H, Afshar PT, Au KF, Bani Asadi N, Gerstein MB, Wong WH, Snyder MP, Schadt E, Lam HYK. Gaining comprehensive biological insight into the transcriptome by performing a broad-spectrum RNA-seq analysis. Nat Commun. 2017 Jul 5;8(1):59. doi: 10.1038/s41467-017-00050-4. PMID: 28680106; PMCID: PMC5498581.
- [4] Rapaport F, Khanin R, Liang Y, Pirun M, Krek A, Zumbo P, Mason CE, Socci ND, Betel D. Comprehensive evaluation of differential gene expression analysis methods for RNA-seq data. Genome Biol. 2013;14(9):R95. doi: 10.1186/gb-2013-14-9-r95. Erratum in: Genome Biol. 2015;16:261. PMID: 24020486; PMCID: PMC4054597.

3. Example workflow

Before running the actual data using RNApipe, we recommend that you test with sample data from the example_data/directory to ensure smooth output of RNApipe.The retrieval of input files and running the workflow locally and on a server cluster is demonstrated using an example with data available from NCBI. This dataset is available under the accession number PRJNA321304(GSE81332).

Note: Ensure that you have miniconda3 installed. Please refer to the overview for details on the installation.

3.1 Setup

We can download the latest version of RNApipe into the newly created project folder and unpack it.

```
wget https://github.com/ywu019/RNApipe/archive/refs/heads/main.zip
unzip main.zip
mv RNApipe-main/ RNApipe; rm main.zip;cd RNApipe;
```

3.2 Retrieve and prepare input files

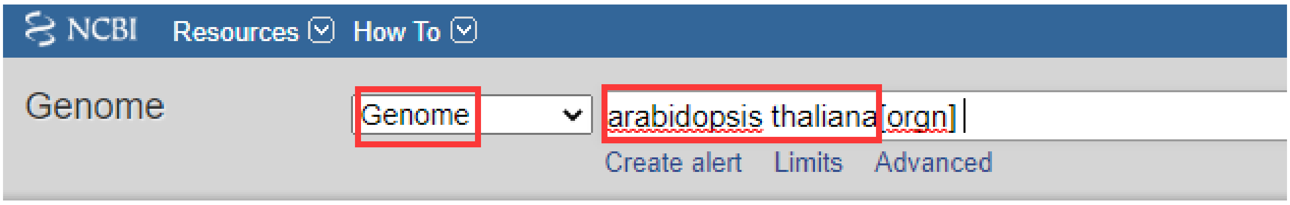
Before starting the workflow, we have to create the project directory and changing to it.

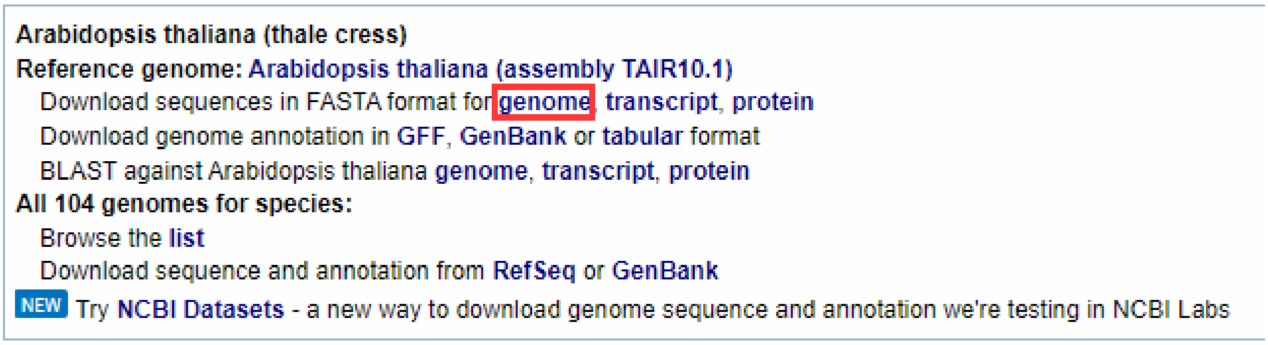
```
cd project
mkdir At_data
cd At_data
```

Then 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

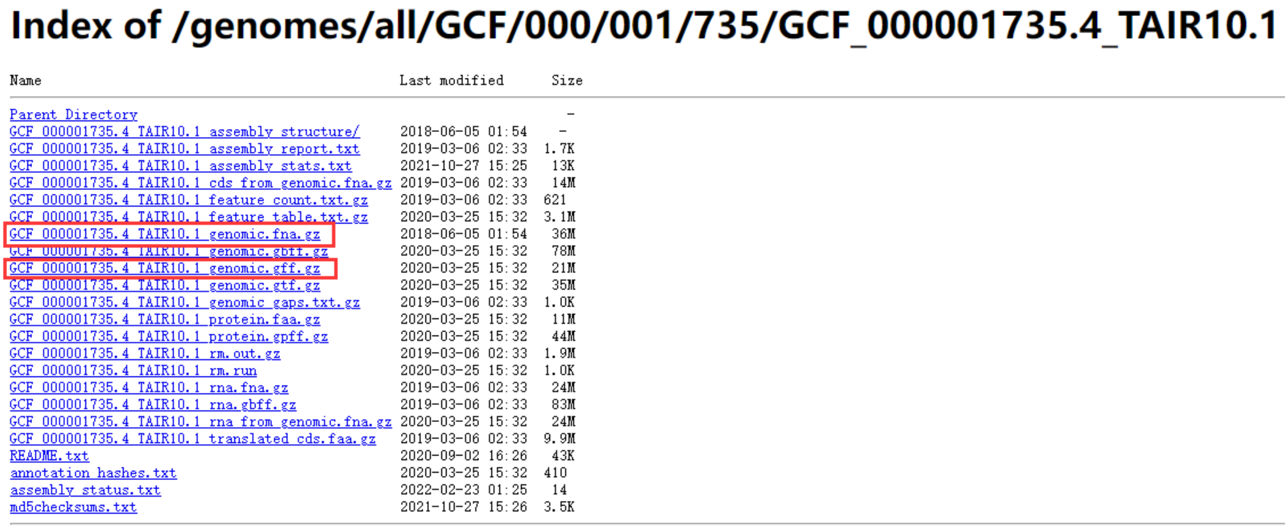
First, we want to retrieve the annotation file and the genome file. In this case, we can find both on NCBI by typing "arabidopsis thaliana" in the search box.





Arabidopsis thaliana (thale cress)
Reference genome: Arabidopsis thaliana (assembly TAIR10.1)
Download sequences in FASTA format for genome, transcript, protein
Download genome annotation in GFF, GenBank or tabular format
BLAST against Arabidopsis thaliana genome, transcript, protein
All 104 genomes for species:
Browse the list
Download sequence and annotation from RefSeq or GenBank
NEW Try NCBI Datasets - a new way to download genome sequence and annotation we're testing in NCBI Labs

Then right-click 'genome' in the above picture, copy the link, edit the link to the previous directory and open the link in a new browser window to get the picture below.



Index of /genomes/all/GCF/000/001/735/GCF_000001735.4_TAIR10.1

```
Name                                              Last modified     Size

Parent Directory                                                      -
GCF_000001735.4_TAIR10.1_assembly_structure/      2018-06-05 01:54    -
GCF_000001735.4_TAIR10.1_assembly_report.txt      2019-03-06 02:33  1.7K
GCF_000001735.4_TAIR10.1_assembly_stats.txt       2021-10-27 15:25   13K
GCF_000001735.4_TAIR10.1_cds_from_genomic.fna.gz  2019-03-06 02:33   14M
GCF_000001735.4_TAIR10.1_feature_count.txt.gz     2019-03-06 02:33  621
GCF_000001735.4_TAIR10.1_feature_table.txt.gz     2020-03-25 15:32  3.1M
GCF_000001735.4_TAIR10.1_genomic.fna.gz           2018-06-05 01:54   36M
GCF_000001735.4_TAIR10.1_genomic.gbff.gz          2020-03-25 15:32   78M
GCF_000001735.4_TAIR10.1_genomic.gff.gz           2020-03-25 15:32   21M
GCF_000001735.4_TAIR10.1_genomic.gtf.gz           2020-03-25 15:32   35M
GCF_000001735.4_TAIR10.1_genomic_gaps.txt.gz      2019-03-06 02:33  1.0K
GCF_000001735.4_TAIR10.1_protein.faa.gz           2020-03-25 15:32   11M
GCF_000001735.4_TAIR10.1_protein.gpff.gz          2020-03-25 15:32   44M
GCF_000001735.4_TAIR10.1_rm.out.gz                2019-03-06 02:33  1.9M
GCF_000001735.4_TAIR10.1_rm.run                   2020-03-25 15:32  1.0K
GCF_000001735.4_TAIR10.1_rna.fna.gz               2019-03-06 02:33   24M
GCF_000001735.4_TAIR10.1_rna.gbff.gz              2019-03-06 02:33   83M
GCF_000001735.4_TAIR10.1_rna_from_genomic.fna.gz  2020-03-25 15:32   24M
GCF_000001735.4_TAIR10.1_translated_cds.faa.gz    2019-03-06 02:33  9.9M
README.txt                                        2020-09-02 16:26   43K
annotation_hashes.txt                             2020-03-25 15:32  410
assembly_status.txt                               2022-02-23 01:25   14
md5checksums.txt                                  2021-10-27 15:26  3.5K
```

On this page, you can directly retrieve both files by clicking on the according download links in the Windows system, and then transfer to the shell. Alternatively, you can directly download them using the following commands:

```
wget https://ftp.ncbi.nlm.nih.gov/genomes/refseq/plant/Arabidopsis_thaliana/latest_assembly_versions/GCF_000001735.4_TAIR10.1/GCF_000001735.4_TAIR10.1_genomic.gtf.gz
wget https://ftp.ncbi.nlm.nih.gov/genomes/refseq/plant/Arabidopsis_thaliana/latest_assembly_versions/GCF_000001735.4_TAIR10.1/GCF_000001735.4_TAIR10.1_genomic.fna.gz
```

Then, we unpack and rename both files.

```
gunzip *.gz
mv GCF_000001735.4_TAIR10.1_genomic.gtf At10.gtf
mv GCF_000001735.4_TAIR10.1_genomic.fna At10.dna.fa
```

Notes: There may be some wrong lines in the At10.gtf annotation file downloaded from NCBI, which will affect the use of subsequent software, so use the following custom script to correct it as At10_new.gtf

```
python cut_wrong_line.py
```

We will move them into a folder called ref

```
mkdir ref
mv At10.gtf At10_new.gtf At10.dna.fa ref
```

3.2.2 .fastq files

Next, we want to acquire the fastq files. The fastq files are available under the accession number PRJNA321304(GSE81332) on NCBI. The files have to be downloaded using the Sequence Read Archive (SRA).
As we already have conda installed, the easiest way is to install the sra-tools:

```
conda create -n sra-tools -c bioconda -c conda-forge sra-tools pigz
```

This will create a conda environment containing the sra-tools. Using these, we can simply pass the SRA identifiers and download the data:

```
conda activate sra-tools;
```

```
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos2/sra-pub-run-3/SRR3498212/SRR3498212.1 &
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos1/sra-pub-run-8/SRR3498213/SRR3498213.1 &
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos2/sra-pub-run-3/SRR3498214/SRR3498214.1 &
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos2/sra-pub-run-3/SRR3498215/SRR3498215.1 &
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos2/sra-pub-run-3/SRR3498216/SRR3498216.1 &
nohup wget https://sra-downloadb.st-va.ncbi.nlm.nih.gov/sos1/sra-pub-run-8/SRR3498217/SRR3498217.1 &

fasterq-dump SRR3498212.1; pigz -p 4 SRR3498212.1.fastq; mv SRR3498212.1.fastq.gz root_1.fastq.gz &
fasterq-dump SRR3498213.1; pigz -p 4 SRR3498213.1.fastq; mv SRR3498213.1.fastq.gz root_2.fastq.gz &
fasterq-dump SRR3498214.1; pigz -p 4 SRR3498214.1.fastq; mv SRR3498214.1.fastq.gz root_3.fastq.gz &
fasterq-dump SRR3498215.1; pigz -p 4 SRR3498215.1.fastq; mv SRR3498215.1.fastq.gz shoot_1.fastq.gz &
fasterq-dump SRR3498216.1; pigz -p 4 SRR3498216.1.fastq; mv SRR3498216.1.fastq.gz shoot_2.fastq.gz &
fasterq-dump SRR3498217.1; pigz -p 4 SRR3498217.1.fastq; mv SRR3498217.1.fastq.gz shoot_3.fastq.gz &
conda deactivate;
```

Warning: If you have a bad internet connection, this step might take some time. If you prefer, you can also use your own .fastq files. But ensure that you use the correct fastq files and compress them in .gz format (using gzip, pigz, etc...)

This will download compressed files for each of the required .fastq files. We will move them into a folder called fastq.

```
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 can modify them in configs/ folder.

The sample file looks as follows:

condition	replicate
root	1
root	2
root	3
shoot	1
shoot	2
shoot	3

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

Next, we are going to set up the config.yaml. In our example, this will lead to the following config.yaml file:

- PROJECT: example_data
- READSPATH: project/example_data/fastq
- SAMPLES: configs/samples.tsv
- END: single
- OUTPUTPATH: project/example_data/output
- GENOME: project/example_data/ref/At10.dna.fa
- ANNOTATION: project/example_data/ref/At10_new.gtf
- CONTROL: ["root"]
- TREAT: ["shoot"]

- SPEICE_ABBREVIATION: ath
- TRIM: fastp # [fastp, cutadapt]
- ADAPTER: AGATCGGAAGAGC # for SE_cutadapt or fastp. #ADAPTER_a: AGATCGGAAGAGC #adapter_A: GCTCTTCCGATCT for PE_cutadapt
- ALIGN: hisat2 # [hisat2, STAR]
- BIGWIG: ["no"] # [yes, no]
- DEA: DESeq2 # [DESeq2, edgeR, limma]
- QUANTIFY: featureCounts #[featureCounts, htseq]
- THREAD: "10" # The number of threads.
- TIME: "5" # The latency for running individual rules programs in main.py.

3.2.4 Download speices database in koabs

Use the following commands to download the database for the corresponding species. Note that this process requires networking, and ensure that the current directory is under RNApipe/ with the following commands.

```
cd ./software/kobas-3.0/sqlite3/
wget ftp://ftp.cbi.pku.edu.cn/pub/KOBAS_3.0_DOWNLOAD/sqlite3/ath.db
cd -
```

3.3 Running the workflow

Now that all the required files are prepared, we can start running the workflow, either locally or in a cluster environment. Before that,remember to activate the RNApipe environment first and in RNApipe/ directory.

```
conda activate RNApipe
cd RNApipe
```

Use the following steps when you plan to execute the workflow on a single server, cloud instance or workstation.
Start the workflow locally from this folder by running:

```
python main.py
```

If you want to run RNApipe in a cluster environment,We recommend that you submit commands to compute nodes interactively to monitor your data performance in real time, such as using `qsub -I` conmmands.

Note: we strongly recommend avoiding the use of compute nodes when using RNApipe to analyze data for the first time, since RNApipe requires the network to set up the Conda microenvironment when it first runs the process.
Note: We do not recommend adding the local python and R interpreters to the environment variables (~/.bashrc) to avoid conflicts with the python and R interpreters in the RNApipe environment, making the RNApipe automation process unable to run smoothly

---

3.4 Results

The output results are stored in trim/, align/, quantify/, DEA/, annotation/, visualize/, corresponding to quality control, comparison, quantitative, difference analysis, functional annotation, and graphic visualization results respectively.

Output directory	Important files
trim/fastp/	<condition>_<replicate>_trimmed.fastq.gz
align/hisat2/	<condition>_<replicate>.sort.bam
quantify/featureCounts/	genome_count.tsv genome_tpm.tsv
DEA/DESeq2/	dea_root_shoot.tsv deg_root_shoot.tsv
annotation/go_kegg/	kobas_deg_root_shoot.txt
annotation/gsea/	root.pdf shoot.pdf *_detail_info
visualize/report/	report_quality_control_after_trimming.html report_align_count.html
visualize/exp/	correlation_root_shoot.pdf cumulative_root_shoot.pdf fvid_root_shoot.pdf heatmap_root_shoot.pdf PCA_root_shoot.pdf violinplot_root_shoot.pdf
visualize/DEA/DESeq2/	heatmap_root_shoot.pdf valcano_root_shoot.pdf
visualize/annotation/go_kegg/	GO_root_shoot_barplot.pdf GO_root_shoot.pdf KEGG_root_shoot.pdf
visualize/annotation/gsea/	root.pdf shoot.pdf