Bulk RNAseq Data Pre-Processing

Snakemake Workflow

Dr. Rudramani Pokhrel

Computational Research Scientiest, The University of Arizona, Immunobiology Department
rpokhrel@email.arizona.edu

Last modified: 19 Apr 2022

The aim of this tutorial is to implement Snakemake workflow to generate gene vs sample count matrix from raw fastq Bulk RNAseq read files. It starts with preparing the virtual environment and Snakemake workflow generate count data. At the end I have provided the necessary script files to process the method in batches.

Install the miniconda package from conda repo for linux platform.

Type following commands in the terminal:

- Download: ***wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh***
- Install: ***bash Miniconda3-latest-Linux-x86_64.sh***
- Initialize the conda environment: ***conda init bash***

Note: You can download your version according to the platforms (linux, macOS, Windows) from this link: https://docs.conda.io/en/latest/miniconda.html. For Apple M1 please follow this link: https://naolin.medium.com/conda-on-m1-mac-with-miniforge-bbc4e3924f2b

Create the virtual environment of named *rna-seq* and install all the required packages and download reference Genomes.

Run following commands in the terminal:

- Create virtual environment named *rna_seq*: ***conda create -y -n rna_seq python=3***
- Activate the environment: ***conda activate rna_seq***
- Install required packages: ***conda install -y -c bioconda snakemake fastp bwa htseq samtools==1.11***
- Use Ensembl (https://uswest.ensembl.org/info/data/ftp/index.html) or from USCS genome browser (https://genome.ucsc.edu) or from NCBI (https://www.ncbi.nlm.nih.gov) or from GENECODE (https://www.gencodegenes.org) to download reference genome and annotation files

The work flow implementation via Snakemake here has following steps:

- Generate bwa index
- Perform QC (adapter trimming and filtering) on read fastq file using ***fastp*** package
- ***bwa mem*** to map fastq files to reference genome
- Use ***samtools*** to convert the mapped ***sam*** files to ***bam*** and sort and index the ***bam*** files
- Use ***htseq-count*** to generate read count to sample matrix data
- Use custom python script ***merge_htseq_counts.py*** to merge individual count matrix data

Let's begin:

- Make a folder for you working directory
  - ***mkdir rnaseq***
  - ***cd rnaseq***
- Make other two folders ***reference*** and ***files***
  - ***mkdir reference files***
    move genome reference ***(fasta and gff3 or gtf)*** to ***reference*** and ***fastq*** reads to ***files***
- In your working directly create a ***config.yaml*** file for ***Snakefile*** input
  - ***config.yaml:***

```
core: 10
  ref:
     fa: reference/reference.fasta
     gff3: reference/reference.gff3

read_dir: files/

read_ext: _R1_001.fastq.gz
```

  - Feel free to edit this ***config.yaml*** file according to your data names and available cpu cores
  - ***Snakefile:***

```
configfile: "config.yaml"
core = config["core"]-1
ext = config["read_ext"]
SAMPLES, = glob_wildcards(config['read_dir']+"{sample}" + ext)

rule all:
    input:
       expand(config["read_dir"] + "trimmed/{sample}_trimmed_R1_fastq.gz",  sample = SAMPLES),
       expand("mapped_read/{sample}.bam", sample = SAMPLES),
       expand("mapped_read/{sample}_sorted.bam", sample = SAMPLES),
       expand("mapped_read/{sample}_sorted.bam.bai", sample = SAMPLES),
       expand("htseq_count/{sample}.txt", sample = SAMPLES),
       expand("htseq_count/merged_counts.csv")

rule bwa_index:
    input:
        ref_fasta  = config["ref"]["fa"]
    output:
        ref  = touch("reference/bwa_index")
    shell:
        "bwa index  {input.ref_fasta} -p {output.ref}"

rule fastp:
    input:
       R1 = config["read_dir"]+ "{sample}" + config["read_ext"]
       R2 = config["read_dir"]+ "{sample}" + config["read_ext"]
    output:
      R1= config["read_dir"]+ "trimmed/{sample}_trimmed_R1_fastq.gz",
      R2= config["read_dir"] + "trimmed/{sample}_trimmed_R2_fastq.gz",
      html= config["read_dir"]+ "trimmed/{sample}.fastp_report.html",
      json= config["read_dir"]+ "trimmed/{sample}.fastp_report.json",
    threads: core
    shell:
      "fastp -w {threads}  -i {input.R1} -I {input.R2}  --detect_adapter_for_pe   -o {output.R1} -O {outpu
t.R2} -h {output.html} -j {output.json}"

rule bwa_mem:
    input:
        ref = "reference/bwa_index",
        R1 = config["read_dir"]+ "trimmed/{sample}_trimmed_R1_fastq.gz",
        R2 = config["read_dir"]+ "trimmed/{sample}_trimmed_R2_fastq.gz"
    output:
       "mapped_read/{sample}.bam"
    log:
       "mapped_read/{sample}.bwa_map.log"
    threads: core
    shell:
       "bwa mem -t {threads} {input.ref} {input.R1} {input.R2} | samtools view -@ {core} -Sb - > {output}
2> {log}"

rule samtools_sort:
    input:
        "mapped_read/{sample}.bam"
    output:
        "mapped_read/{sample}_sorted.bam"
    shell:
        "samtools sort   -o {output}  {input}"

rule samtools_index:
    input:
        "mapped_read/{sample}_sorted.bam"
    output:
        "mapped_reads{sample}_sorted.bam.bai"
    shell:
        "samtools index {input}"

rule htseq_count:
    input:
      gff3 = config["ref"]["gff3"],
      bam = "mapped_read/{sample}_sorted.bam",
      bai = "mapped_read/{sample}_sorted.bam.bai"
    output:
        "htseq_count/{sample}.txt"
    shell:
        "htseq-count -n {core} --format=bam --stranded=no "
        --type=gene --order=pos --idattr=ID {input.bam}  {input.gff3}  > {output}"

rule merge_count:
    input:
        expand("htseq_count/{sample}.txt", sample = SAMPLES)
    output:
        "htseq_count/merged_counts.csv"
    shell:
      "cd htseq_count"
      "wget https://raw.githubusercontent.com/githubrudramani/Bioinformatics/master/merge_htseq_counts.py"
      "python merge_htseq_counts.py"
```

Note: the indentation in ***config.yaml*** and ***Snakefile*** is not tap seperated.

Run command in your working directory ***rnaseq:***

```
snakemake --snakefile Snakefile --core 10
```

Note: While running this Snakefile, I found the clash of library between ***snakemake*** and ***htseq***.
In such case you can create individual environment for ***htseq***.
For that purpose modify the ***rule htseq_count***:

```
rule htseq_count:
    input:
       gff3 = config["ref"]["gff3"],
       bam = "mapped_read/{sample}_sorted.bam",
       bai = "mapped_read/{sample}_sorted.bam.bai"
    output:
        "htseq_count/{sample}.txt"
    conda:
        "htseq.yaml"
    shell:
        ".snakemake/conda/*/bin/htseq-count -n {core} --format=bam --stranded=no "
        --type=gene --order=pos --idattr=ID {input.bam}  {input.gff3}  > {output}"
```

In ***htseq.yaml*** file insert following:

```
channels:
  - bioconda
  - conda-forge
dependencies:
  - samtools =1.1
  - htseq
```

Then run command in your working directory ***rnaseq:***

```
snakemake --snakefile Snakefile --core 10 --use_conda
```

Output:

merged_counts.csv

After that you can use differential analysis statistical packages like **DEseq2**, **edgeR**, **Limma** from Bioconductor.