

SEAsnake vignette

snakemake pipeline from fastq to counts

Kim Dill-McFarland, kadm@uw.edu

version January 21, 2022

Contents

Introduction	1
Software	2
Resource requirements	2
AWS installation	2
Other linux installation	3
Updating SEAsnake	4
SEAsnake	4
fastq data	4
Reference genome	4
Run SEAsnake	5
Step 1	5
Customize config	7
Step 2	9
Save results	9
Example results	10
Contribute to SEAsnake	10
Cite SEAsnake	10

Introduction

[SEAsnake](#) is a [snakemake](#) pipeline to process bulk RNA-seq data from fastq sequences to gene counts. It includes the following steps. You can see an in-depth example of how to run these steps separately outside SEAsnake in our [first tutorial](#).

1. Quality assess sequences with [FastQC](#)
2. Remove adapters and filter low quality sequences with [AdapterRemoval](#)
 - max N = 1
 - min length = 15
 - min quality = 30
 - 5' and 3' adapters specified in config file
3. Align to reference genome with [STAR](#)
4. Quality filter alignments with [samtools view](#)
 - header retained
 - paired reads where both mapped
 - min MAPQ = 30

- remove unmapped, non-primary alignments, and PCR duplicates
5. Quality assess alignments with [samtools flagstat](#) and/or [Picard CollectRnaSeqMetrics](#)
 6. Count reads in genes with [Subread featureCounts](#)
 - exons only

Here, we provide an example of how to install and run SEAsnake on human bulk RNA-seq.

Software

Resource requirements

CPU: Many of the steps in SEAsnake run in parallel. Thus, the time to run the pipeline on 2 CPUs is roughly half of 1 CPU (not counting non-parallel processes like genome indexing). Thus, you could run as many CPUs as you have fastq files. However, this is not recommended as there is a trade-off with RAM usage.

RAM: Estimating RAM usage is difficult. The most impactful values to consider are number of CPUs, fastq file size, and reference genome size. Each CPU uses equivalent RAM for parallel jobs. So 2 CPUs require twice the RAM as 1 CPU. Sample and genome size also contribute as larger files use more RAM on each CPU being run. The most common pipeline failure is running out of RAM, so when in doubt, choose the larger option.

Storage: SEAsnake itself is a small program (< 150 MB). Thus, storage needs depend almost entirely on your sample size and sequencing depth of those samples. Larger raw fastq result in larger result files.

Our recommendations: We follow a general rule of $\text{CPU} = \text{fastqs} / 3$ and $\text{RAM} = \text{CPU} * 2 + 40$ where each CPU gets 2 GB of RAM and there is an additional 40 GB for the human genome. If your fastq are large (> 5 GB), you may need to increase the CPU factor to 2.5 or 3. The exact number of CPUs depends on what is available on AWS, and these parameters are commonly met by m5 instance types. For storage, we recommend 65 GB per 10 fastq files. This comes to roughly 1 TB for 150 fastq or 75 paired-end samples.

Note that indexing the human genome requires a minimum of 40 GB of RAM, so do not go below this if you need to complete that step.

AWS installation

SEAsnake can be pre-installed on any AWS EC2 instance.

1. Launch a new instance from the AWS console. Search 'seasnake' and select the Community AMI SEAsnake.
2. Choose your instance type keeping your resource requirements in mind. m5 types are our most used class. For this tutorial's data, use **r5a.2xlarge** (sample fastq are very small)
3. Leave configuration as defaults.
4. Add additional EBS storage to hold your data and results. Please leave the root storage as 16 GB; this holds the underlying software.
5. Tag your instance if desired.
6. Configure security as you normally would.
7. Launch the instance. This will take a couple minutes to complete before you can log-in.

Once the instance is running, log-in and complete setup as follows. First, define your AWS account information.

```
AWS_ACCESS_KEY="XXXX"
AWS_SECRET_ACCESS_KEY="XXXX"
AWS_REGION="xxxxx"
```

Then, run the following script.

```
#### Basic AWS update ####
sudo yum upgrade -y
```

```

sudo yum update -y

#### Configure AWS ####
## Configure your account
export AWS_ACCESS_KEY_ID=$AWS_ACCESS_KEY
export AWS_SECRET_ACCESS_KEY=$AWS_SECRET_ACCESS_KEY
export AWS_DEFAULT_REGION=$AWS_REGION

## Setup fuse keys
echo $AWS_ACCESS_KEY:$AWS_SECRET_ACCESS_KEY > ~/.passwd-s3fs
chmod 600 ~/.passwd-s3fs

#### Setup EBS volume ####
## Get addtl volume name
## If this does not give the correct volume name, find it with lsblk
ebs_name=$(lsblk -o NAME -n -i -r | tail -n 1)
echo $ebs_name
## Format volume
sudo mkfs -t ext4 /dev/$ebs_name
## Attach SEAsnake directory to volume
sudo mount /dev/$ebs_name ~/SEAsnake
## Change permissions to read-write
sudo chmod 777 -R ~/SEAsnake/
## Remove default subdir
sudo rm -R ~/SEAsnake/lost+found

## Clone SEAsnake from GitHub
git clone https://github.com/BIGslu/SEAsnake ~/SEAsnake

```

Other linux installation

If you are not using AWS, you can install SEAsnake and its dependencies using the following scripts.

```

#### Install conda ####
## Download conda
sudo mkdir -m 777 -p ~/apps/anaconda
cd ~/apps/anaconda
sudo curl -O https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86_64.sh

## Compile and install conda
sudo bash Anaconda3-2021.11-Linux-x86_64.sh -b -p /home/ec2-user/apps/anaconda -u
eval "$(/home/ec2-user/apps/anaconda/bin/conda shell.bash hook)"
conda init
sudo chmod 777 -R ~/apps/

```

Restart your terminal for conda initiation to take effect.

```

## Configure addtl conda channels
conda config --add channels bioconda
conda config --add channels conda-forge
conda config --set allow_conda_downgrades true

#### Install mamba ####
conda install -n base -c conda-forge mamba -y --repodata-fn repodata.json

```

```
#### Install SEAsnake ####
## Install git
sudo yum install git -y

## Clone SEAsnake from GitHub
sudo mkdir -m 777 ~/SEAsnake
git clone https://github.com/BIGslu/SEAsnake ~/SEAsnake

## Create environment and install SEAsnake software with mamba
mamba env create --name SEAsnake --file ~/SEAsnake/environment/Hissss_env.yaml
```

Updating SEAsnake

If you previously installed SEAsnake, you can update it by pulling the latest version from GitHub.

```
cd ~/SEAsnake
git pull
```

SEAsnake

fastq data

Note: SEAsnake is only setup to run paired-end fastq.gz files in its data/ directory.

If you are using fuse to access your fastq data on S3, link your bucket to the SEAsnake data/ directory.

```
# Define the name of your data bucket
DATA_BUCKET="MY_BUCKET_NAME"
# Make directory for data
sudo mkdir -m 777 ~/SEAsnake/data
# Fuse bucket to directory
s3fs $DATA_BUCKET ~/SEAsnake/data \
    -o passwd_file=~/.passwd-s3fs \
    -o default_acl=public-read -o uid=1000 -o gid=1000 -o umask=0007
```

If you are transferring data by another method, create the SEAsnake/data/ directory and copy your data there. For example, to run this vignette's data.

```
# Make directory for data
sudo mkdir -m 777 ~/SEAsnake/data
# Copy fastqs
cp ~/SEAsnake/vignette/data/*fastq.gz ~/SEAsnake/data
```

Reference genome

If you are using fuse to access your STAR-formatted genome index on S3, link your bucket to the SEAsnake ref/ directory. *Hawn/Altman labs: These are in the bucket human-ref.*

```
# Define the name of your data bucket
REF_BUCKET="MY_BUCKET_NAME"
# Make directory for data
sudo mkdir -m 777 ~/SEAsnake/ref
# Fuse bucket to directory
s3fs $REF_BUCKET ~/SEAsnake/ref \
    -o passwd_file=~/.passwd-s3fs \
    -o default_acl=public-read -o uid=1000 -o gid=1000 -o umask=0007
```

If you do not have a pre-built index, the pipeline will make one for you!

Run SEAsnake

Set the number of cores you would like to use. This should be no more than your total CPU - 1. Then, activate the conda environment, which contains all pre-installed software, and move into the SEAsnake directory.

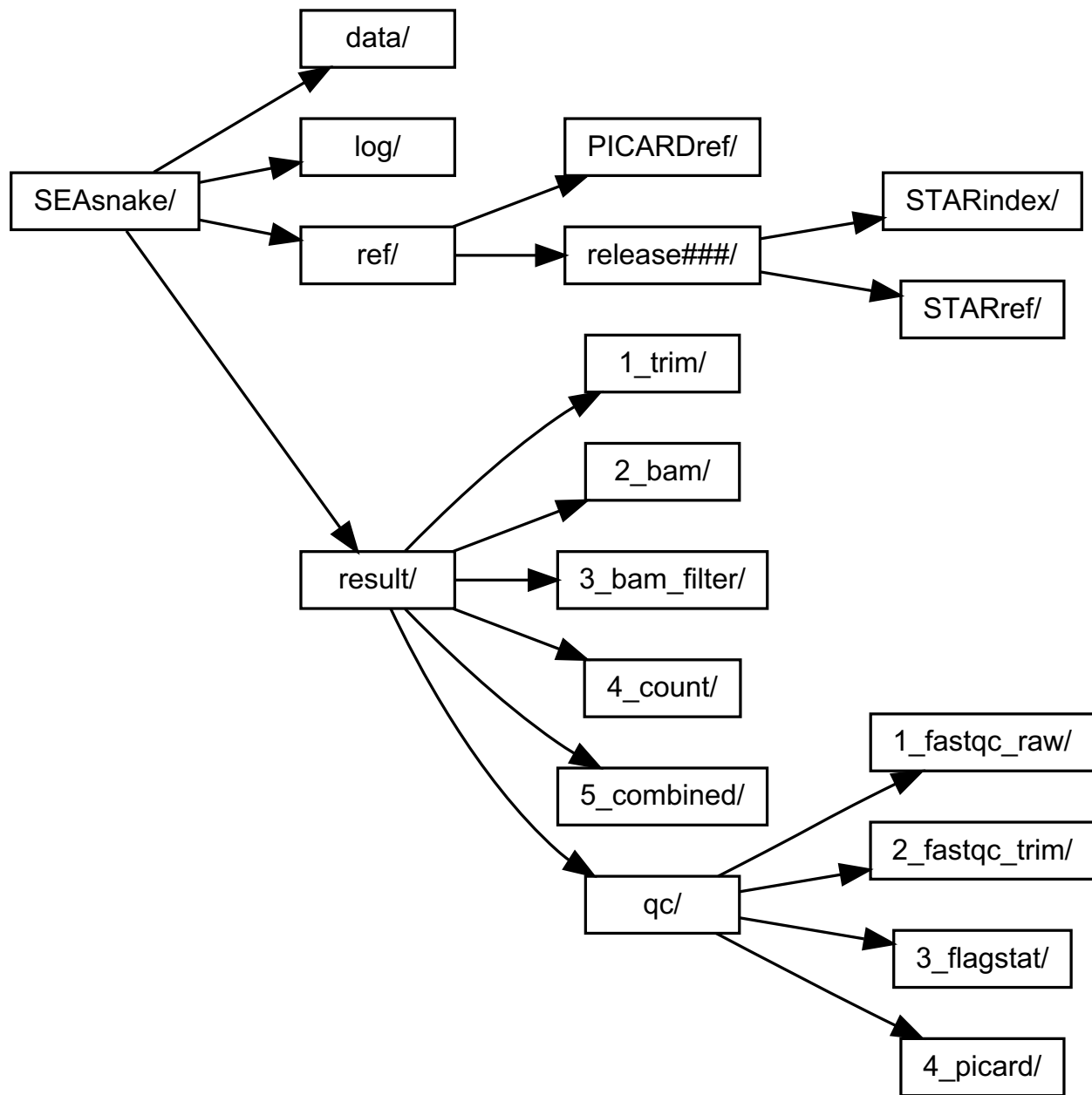
```
cores=15
conda activate SEAsnake
cd ~/SEAsnake
```

Step 1

Next, run SEAsnake step 1. This completes initial sequence quality assessment, directory structure setup, and config file creation. *Note that **nohup** and piping into a log file means nothing will appear in your terminal window. This prevents timeout and retains all messages and errors in the log file.*

```
nohup snakemake --snakefile Snakefile_step1 --cores $cores >> log/SEAsnake_step1.log 2>&1
```

Your directory structure will look like



Once step 1 is complete, you will see the command prompt \$ re-appear in your console. If you want to check progress or the terminal window times out, open a new window, log back in, and check progress in the log.

```
tail ~/SEAsnake/log/SEAsnake_step1.log
```

Or ask SEAsnake to summarize how many tasks need to still be completed. The `-n` flag is a “dry run” where SEAsnake does not actually run anything.

```
conda activate SEAsnake
snakemake --snakefile ~/SEAsnake/Snakefile_step1 -n
```

You’ll see something like this. This example shows that 2 of the fastq files still need to complete FastQC.

```
Job stats:
job      count  min threads  max threads
-----
-----
```

all	1	1	1
fastqc_raw	2	1	1
total	3	1	1

This was a dry-run (flag -n). The order of jobs does not reflect the order of execution.

Customize config

Step 1 creates `result/config.yaml` which allows some customization of the workflow. You may wish to change some defaults. Below is an example from the vignette data with all defaults.

```
SampleList:
  test_S1:
    sample: 'test_S1'
    R1: 'data/test_S1_L005_R1_001.fastq.gz'
    R2: 'data/test_S1_L005_R2_001.fastq.gz'
  test_S2:
    sample: 'test_S2'
    R1: 'data/test_S2_L005_R1_001.fastq.gz'
    R2: 'data/test_S2_L005_R2_001.fastq.gz'

# Adapter removal
## Base pairs to trim from 5' end
trim5p: 10
## Removal of 3' adapter sequences? Default are Illumina Universal adapters
trimAdapt: True
adapter1: AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC
adapter2: AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT

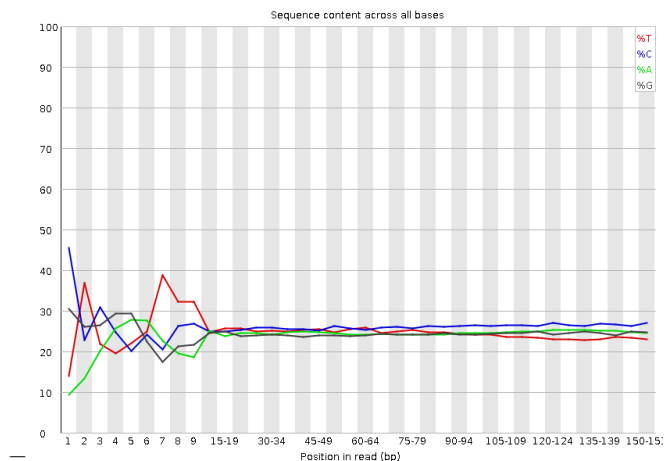
# Genome alignment
## Human genome release number
release: '105'

# Alignment metrics
## Run Picard?
picard: True

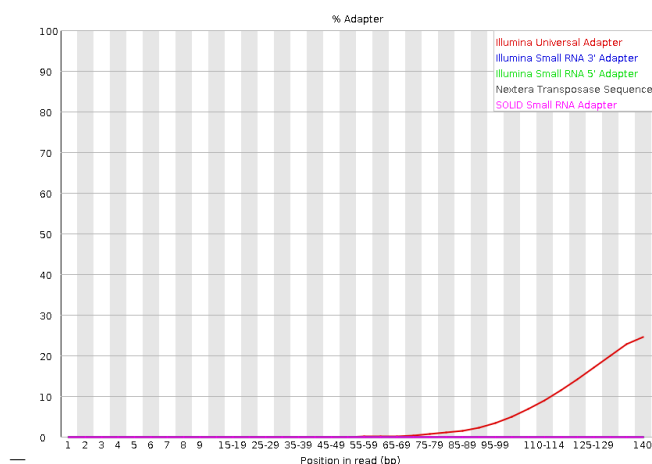
# Other
threads: 1
cores: 15
```

Customization options

- SampleList
 - SEAsnake lists all `fastq.gz` from the `data/` directory and will run the pipeline for all of these samples. To remove a sample, simply remove the 4 lines relevant to that sample. To rename result files for a sample, change the “sample” field.
- Adapter removal
 - `trim5p`: Number of base pairs to trim from the 5’ end. You determine this from FastQC “Per base sequence content”. For example, this sample appears to have disproportionately high calls up to 15 bp. This supports trimming the first 15 bp of these sequences (default).



- trimAdapt: Sequencing adapters may also exist at the 3' end. These can be seen in FastQC "Adapter content". You remove these by aligning to adapter sequences provided in the config file. Default are Illumina Universal adapters but you can provide any adapter sequence.
- If you do not see 3' adapter contamination, **set trimAdapt to False!** You can leave the adapter sequences as is as they will not be used with the False setting. This dramatically speeds up adapter trimming.



- Genome alignment
 - Currently SEAsnake only supports human genome alignment. Specify the ENSEMBL release to use [here](#).
- Alignment metrics
 - picard: This program provides additional alignment quality metrics listed [here](#). If you do not need these metrics, set to False.
- threads
 - This sets the maximum number of threads **per job** as opposed to the SEAsnake `--cores` parameter which sets the maximum number of cores to run jobs on in parallel. The total cores being used is threads * cores.
 - By default, threads is your total CPU minus 1 and is automatically scaled for each step in SEAsnake based on its individual requirements. You should not need to change this but can reduce it if you experience issues with running out of CPU or RAM space.
 - *MAC and WINDOWS*: Auto core detection may not work. Please verify that the config file contains a usable numeric value.

You can also find a more thorough introduction to quality assessment in our [first tutorial](#).

Step 2

Run SEAsnake step 2. This completes adapter trimming, alignment, filtering, quality assesment, and exon counting.

```
nohup snakemake --snakefile Snakefile_step2 --cores $cores >> log/SEAsnake_step2.log 2>&1
```

This step can take quite some time to run. Because of `nohup`, you can close the terminal at any time and SEAsnake will continue to run. Check in on progress in a new terminal window using the `log` or `-n` flag as you did for step 1.

For example,

```
tail ~/SEAsnake/log/SEAsnake_step2.log
```

```
conda activate SEAsnake
snakemake --snakefile ~/SEAsnake/Snakemakefile_step2 -n
```

Which gives something like the following.

```
Job stats:
job          count  min threads  max threads
-----
STAR_align      2           1           1
STAR_index      1           1           1
adapterremoval  2           1           1
align_filter    2           1           1
all             1           1           1
combine         1           1           1
fastqc_trim     4           1           1
fcount          2           1           1
flagstat        2           1           1
picard           2           1           1
total          19           1           1
```

This was a dry-run (flag `-n`). The order of jobs does not reflect the order of execution.

Save results

Once complete, save your `result/` and `log/` directories because these will be lost once you stop your EC2 instance. We recommend using `aws s3 sync` to save to an S3 bucket like so.

```
aws s3 sync ~/SEAsnake/result/* s3://RESULT_BUCKET
aws s3 sync ~/SEAsnake/log/* s3://RESULT_BUCKET
```

You may also wish to save your genome index for use in future runs. This saves about an hour of run time for human samples! *Hawn/Altman labs: Save to the **human-ref** bucket.*

```
aws s3 sync ~/SEAsnake/ref/* s3://human-ref
```

Then, close the conda environment.

```
conda deactivate
```

And un-fuse any buckets in use.

```
fusermount -u ~/SEAsnake/data
fusermount -u ~/SEAsnake/ref
```

Example results

You main results will be in `result/5_combined/` where all samples have been combined into a single table per data type.

Counts table.

```
## # A tibble: 6 x 3
##   Geneid      test_S1 test_S2
##   <chr>      <dbl>   <dbl>
## 1 ENSG00000284662      0      0
## 2 ENSG00000186827      0      0
## 3 ENSG00000186891      0      0
## 4 ENSG00000160072      0      0
## 5 ENSG00000041988      0      0
## 6 ENSG00000260179      0      0
```

Flagstat alignment metrics.

```
## # A tibble: 2 x 17
##   libID QC_pass primary secondary supplementary duplicate primary_duplicate
##   <chr>   <dbl>   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 test_S1  11990   11990      0      0      0      0
## 2 test_S2   5980    5980      0      0      0      0
## # ... with 10 more variables: mapped <dbl>, primary_mapped <dbl>, paired <dbl>,
## #   read1 <dbl>, read2 <dbl>, paired_proper <dbl>, paired_mapped <dbl>,
## #   singleton <dbl>, paired_diff_chr <dbl>, paired_diff_chr5 <dbl>
```

Picard alignment metrics.

```
## # A tibble: 2 x 31
##   libID PF_BASES PF_ALIGNED_BASES RIBOSOMAL_BASES CODING_BASES UTR_BASES
##   <chr>   <dbl>    <dbl> <lgl>    <dbl>    <dbl>
## 1 test_S1 1656408    1638795 NA      797841    490355
## 2 test_S2  825848     816834 NA      392505    246977
## # ... with 25 more variables: INTRONIC_BASES <dbl>, INTERGENIC_BASES <dbl>,
## #   IGNORED_READS <dbl>, CORRECT_STRAND_READS <dbl>,
## #   INCORRECT_STRAND_READS <dbl>, NUM_R1_TRANSCRIPT_STRAND_READS <dbl>,
## #   NUM_R2_TRANSCRIPT_STRAND_READS <dbl>, NUM_UNEXPLAINED_READS <dbl>,
## #   PCT_R1_TRANSCRIPT_STRAND_READS <dbl>, PCT_R2_TRANSCRIPT_STRAND_READS <dbl>,
## #   PCT_RIBOSOMAL_BASES <lgl>, PCT_CODING_BASES <dbl>, PCT_UTR_BASES <dbl>,
## #   PCT_INTRONIC_BASES <dbl>, PCT_INTERGENIC_BASES <dbl>, ...
```

Contribute to SEAsnake

SEAsnake is an open source workflow. We would love your feedback including error/bug reports and requests for additional features! Please let us know on our [GitHub](#). We also welcome community code additions through pull requests!

Cite SEAsnake

Coming soon!