Author: Bernadette Johnson
Updated: December 2022

How to Install and Use HISAT2 and StringTie

About this Protocol

This protocol is for users who are interested in mapping next-generation sequence reads to a reference genome. There are similarities between this general pipeline, and one used for de novo assembly (TRINITY→RSEM), however this requires you also have a reference genome to map to. This guide is useful for novice bioinformaticians users who would like to set-up and run HISAT2 and StringTie for the first time. This protocol ends by preparing your StringTie output for use in a differential expression analysis program such as EBSeq, DESeq2, or edgeR. It is important that you use already trimmed reads for this protocol. If you do not have trimmed reads, please visit my protocol website: bernadettebiology.weebly.com/protocols--tutorials.html

I recommend users read the original manual while planning any analysis. Find them here: HISAT2 (http://daehwankimlab.github.io/hisat2/), and StringTie (https://ccb.jhu.edu/software/stringtie/).

BLUF

Computer requirements and recommendations:

- A Linux subsystem, or a Linux virtual box
- Additional hard drive, other than your Local Disk (C:) drive, with ~2 TB of free space.
- Demands on RAM and CPU can be set low (but will have longer run time). I will use 40 GB RAM.

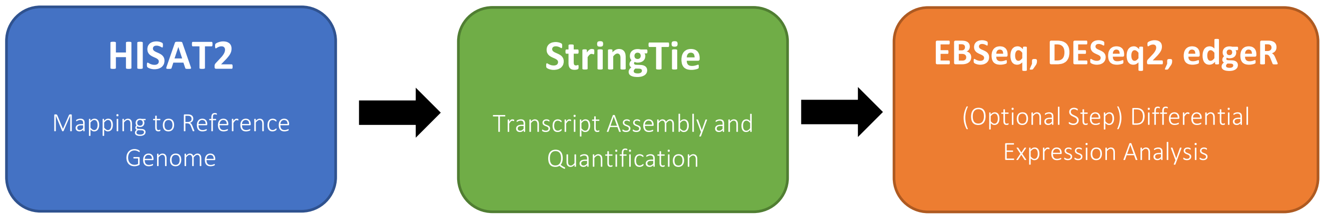
Total programs downloaded for this protocol: make, gcc, libz-dev, g++, samtools, HISAT2, Stringtie, and Python 3

Information for working example:

Context: For this project, I am interested in mapping trimmed, paired-end reads of male and female samples of a species of pipefish. I would like to follow this up with a differential expression analysis, so I will also prepare the data for use in EBSeq. Red text will indicate my specific path or file name.

System: I will be using the Ubuntu subsystem for on Windows 10 (Install Ubuntu on Windows). I have a total of 5 TB of free space, 128 GB of RAM, and 12 CPU logical processors. In my home directory, I have a folder named 'shared' which I will work in for most of the process and examples.

General Pipeline


HISAT2
Mapping to Reference Genome
StringTie
Transcript Assembly and Quantification
EBSeq, DESeq2, edgeR
(Optional Step) Differential Expression Analysis

Let's get started

Update and upgrade

1. Update and upgrade your Ubuntu subsystem before starting. Open a new terminal window, then run the following commands.

```
> sudo apt-get update
> sudo apt-get upgrade
```

Install prerequisites

2. Install these programs if they are not already installed. We will need these to install our other programs.

```
> sudo apt-get install make
> sudo apt-get install gcc
> sudo apt-get install libz-dev
> sudo apt-get install g++
```

Installing samtools

3. Install the latest version of samtools. You can download it from the official site (github.com/samtools/). The latest version for me is samtools v1.16.1. Once it is done downloading, move the downloaded folder into the desired location or directory. Alternatively, you can use the command line to download the file into your working directory (this command is below).

```
> git clone --recurse-submodules
https://github.com/samtools/htslib.git

> git clone https://github.com/samtools/bcftools.git
> cd bcftools
> make
> export BCFTOOLS_PLUGINS=/path/to/bcftools/plugins
# For me this is: ~/bcftools
```

4. Test for installation.

```
> cd
> samtools --help
#If samtools is properly installed, you will see the usage and
commands guide (see Tips on installing programs section for more
information).
```

Installing HISAT2

5. Install using command line.

```
> sudo apt install hisat2
```

6. Test for installation.

```
> cd
> hisat2 --help
```

Installing StringTie

7. Downloading through GitHub usually insures an up-to-date version. This will download into your current working directory.

```
> git clone https://github.com/gpertea/stringtie
> cd stringtie
> make stringtie
```

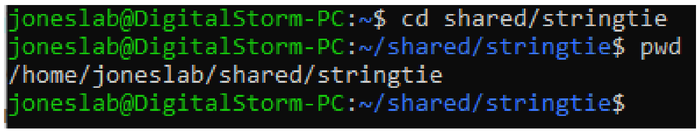
8. Then we can add StringTie to our PATH to make it easily accessible, even when we are not working in the downloaded StringTie folder.

```
> cd
> nano .bashrc
```

Please do NOT change any other part of this file, simply scroll down to the end of the file and append this to the end:

```
#stringtie

export PATH=$PATH:~/shared/stringtie
```

*This is my path to my stringtie folder. Your folder might be in a different location. Whenever you want to check your current PATH, or the location of your folder you can do one of two things: (1) you can navigate to the folder of interest and use the *pwd* command, (2) alternatively, you can navigate to the folder of interest and check the command line that lets you know where you are (the blue text). The first option gives you the full PATH name, the second starts from '~' which symbolizes your home directory.



```
joneslab@DigitalStorm-PC:~$ cd shared/stringtie
joneslab@DigitalStorm-PC:~/shared/stringtie$ pwd
/home/joneslab/shared/stringtie
joneslab@DigitalStorm-PC:~/shared/stringtie$
```

You can always check what is on your *.bashrc* script from any directory, with the command:

```
>echo $PATH.
```

9. Test for installation.

```
> exit
#This will close your terminal window. We want to restart your
session. Open a new terminal window.
> stringtie --help
```

Running HISAT2

10. I am using trimmed, paired-end reads. This means I have used Trimmomatic to remove low quality and short reads, as well as Illumina adapters. **It is important that you use trimmed reads**. Installation and usage of Trimmomatic is not included in this protocol. Please see my website for this protocol: *bernadettebiology.weebly.com/protocols--tutorials.html*

11. Start by building the HISAT2 index. You will build an index once per genome. Your genome file should be in FASTA format (i.e. it should have an extension like .fna, .fasta, or .fa.). We will build the index in its own folder.

```
> mkdir hisat2_index            #Make a new folder
> mv genome.fna hisat2_index    #Move your genome file to folder
> cd hisat2_index               #Change directory into folder

> hisat2-build genome.fna genome_index
#Make sure this is the name to your genome file.
```

12. Now we will align all the read files to the genome. You will need to run this command for each sample. Depending on the number of samples you have, this will take some time. It might be easier to open a new text document (such as Notepad ++) type this command out for each sample, double check it, then run this as a shell script in your terminal.

For this command, please read the HISAT2 manual to determine which parameters are right for your analysis. I am using the following set of parameters:

- `--dta` #This option is needed for StringTie.
- `-x` #This is the folder where you built your genome index. Make sure this is the path to your folder.
- `-1` #This is a list of your forward reads.
- `-2` #These are your reverse reads.
- `-S` #This is an output and is a SAM alignment file for each read.
- *Hisat2Output.txt* #While your reads are aligning to the genome, HISAT2 will produce an alignment summary for each read. This output will be saved to a text file called *Hisat2Output.txt*.

```
#First sample, called pipefish1
> hisat2 --dta -x ~/pipefishRNA/hisat2_index/genome_index -1
pipefish1_R1_paired.fastq -2 pipefish1_R2_paired.fastq -S
pipefish1.sam &>> Hisat2Output.txt
```

```
#Second sample, called pipefish2
> hisat2 --dta -x ~/pipefishRNA/hisat2_index/genome_index -1
pipefish2_R1_paired.fastq -2 pipefish2_R2_paired.fastq -S
pipefish2.sam &>> Hisat2Output.txt

#Keep going until you have run this command for each sample. You
will see the generation of a new .sam file for each sample
successfully completed. Once all of your samples are mapped, you
can open up your Hisat2Output.txt file and check the alignment
scores. A successful run will look like this:
```

```
68406586 reads; of these:
  68406586 (100.00%) were paired; of these:
    10014637 (14.64%) aligned concordantly 0 times
    51347569 (75.06%) aligned concordantly exactly 1 time
    7044380 (10.30%) aligned concordantly >1 times
    ----
    10014637 pairs aligned concordantly 0 times; of these:
      600779 (6.00%) aligned discordantly 1 time
    ----
    9413858 pairs aligned 0 times concordantly or discordantly; of these:
      18827716 mates make up the pairs; of these:
        10258846 (54.49%) aligned 0 times
        6456328 (34.29%) aligned exactly 1 time
        2112542 (11.22%) aligned >1 times
92.50% overall alignment rate
```

Prepare files for StringTie

13. You will need to convert each .sam file into a .bam file. This command should be run with each sample's .sam file. The input file is the .sam file, and the output file is the .bam file.

```
> samtools view -b -S pipefish1.sam > pipefish1.bam
> samtools view -b -S pipefish2.sam > pipefish2.bam
```

14. You will also need to sort out your bam files. This command should be run with each .bam file. The input file is the .bam file, and the output file is the sorted.bam file. I am using 40 GB of memory. Adjust this value appropriately for the amount of RAM you have.

```
> samtools sort -m 40G pipefish1.bam -o pipefish1_sorted.bam
> samtools sort -m 40G pipefish2.bam -o pipefish2_sorted.bam
```

Running StringTie

15. Now we will quantify our samples using StringTie. You will run this command for each sorted.bam file. I generally recommend against using a genome annotation (-G parameter option). There are two reasons for this. First, it might negatively impact your results if the available genome annotation is not great. Second, it may introduce transcript naming errors. This may eventually get fixed with new updates; however, this was an issue at the time of writing this guide. For this reason, I am not using a genome annotation for this guide.

    I am using the following set of parameters:

    - `-o` `#This is an output file. It is a .gtf file and provides information connecting your transcripts to their genomic location and abundance.`
    - `-A` `#This is an output and is a .tab file containing abundances for each transcript.`

    ```
    > stringtie pipefish1_sorted.bam -o pipefish1_gtf.gtf -A
    pipefish1_abund.tab
    > stringtie pipefish2_sorted.bam -o pipefish2_gtf.gtf -A
    pipefish2_abund.tab
    ```

16. You will need to merge the transcripts from all samples to produce a single .gtf file. This command can be extremely long, depending on the number of samples you have and their location. You should make sure you include all your samples in this one command, otherwise you won't be merging your samples. I suggest entering this all in a text file, and either copying and pasting the command into your terminal or running the text file as a shell script. Here are both options:

    ```
    #OPTION 1 (using a text file): Your text file should contain
    this. Save it as a .txt file or a .sh file.
    stringtie --merge \
    -o merged_transcripts.gff \
    path/to/pipefish1.gtf \
    path/to/pipefish2.gtf \
    #Then in the command line you can run your text file.
    > ./textfile.txt
    #OPTION 2 (using command line):
    > stringtie --merge -o stringtie_merged.gff
    path/to/pipefish1.gtf path/to/pipefish2.gtf
    ```

17. We will now estimate transcript abundances using the newly generated merged .gff file. This is an important step for creating table counts for a downstream differential expression analysis.

```
> mkdir ~/pipefishRNA/stringtie/
#I am making a new directory or folder for all of the files we
are about to generate.

> stringtie pipefish1_sorted.bam -B -e -o
~/pipefishRNA/stringtie/pipefish1_merged.gtf -G
stringtie_merged.gff -A
~/pipefishRNA/stringtie/pipefish1_abund_merged.tab -C
~/pipefishRNA/stringtie/pipefish1_cov_refs_merged.gtf
```

For this command, I am using the following set of parameters:

- **sample_sorted.bam** #This is the input file. It contains information on the RNA-Seq read alignments and their genomic location.
- **-B** #This option returns a Ballgown input table file, which will be useful for generating a table of actual read counts for a differential expression analysis.
- **-e** #StringTie will not assemble the input read alignments but instead will only estimate the expression levels of the reference transcripts. This option is used with the '-G' parameter to indicate the reference transcripts.
- **-o** #This is your output .gtf file for the merged transcripts. It is different than the first one we generated.
- **-G** #Tells StringTie that you plan to use a file with reference annotation transcripts. In this case, we are using the previously generated merged .gff file.
- **-A** #This is an output file of gene abundances and will be in tab delimited format (like an Excel spreadsheet).
- **-C** #This is an output file that contains information on the transcripts from the reference annotation transcript file that are fully covered by the reads in your sample file.
- **~/pipefishRNA/stringtie/** #I am directing all of my new output into a separate folder called "stringtie". This will make it easier for the generating a gene count matrix (next step).

Producing a gene count matrix for differential expression analysis

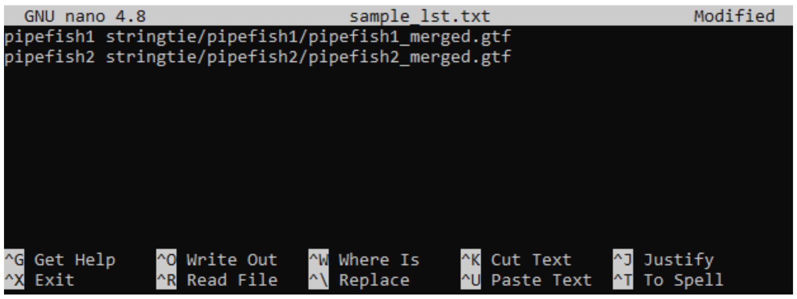
18. If your intention is to conduct a differential expression analysis with your transcriptome, it is likely that you will want to generate a gene count matrix. You will start by making sure all of your output files, generated in the previous step, are properly organized. To do this, navigate into the folder containing these outputs. My folder is called **`stringtie`**. This folder should contain the following files: **`sample_merged.gtf`**, **`sample_abund_merged.tab`**, and **`sample_cov_refs_merged.gtf`**. You will want to place each of the three files per sample, into their own folder. Each folder should be named with the sample name.

```
> cd stringtie
#I am moving into my folder called "stringtie".
> mkdir ~/pipefishRNA/stringtie/pipefish1
#I am making a new folder with the name of my sample
> mv ~/pipefishRNA/stringtie/pipefish1_merged.gtf
~/pipefishRNA/stringtie/pipefish1
> mv ~/pipefishRNA/stringtie/pipefish1_abund_merged.tab
~/pipefishRNA/stringtie/pipefish1
> mv ~/pipefishRNA/stringtie/pipefish1_cov_refs_merged.gtf
~/pipefishRNA/stringtie/pipefish1
#I am moving these three output files into my new sample folder.
#Then you can repeat these steps for each sample.
> mkdir ~/pipefishRNA/stringtie/pipefish2
> mv ~/pipefishRNA/stringtie/pipefish2_merged.gtf
~/pipefishRNA/stringtie/pipefish2
> mv ~/pipefishRNA/stringtie/pipefish2_abund_merged.tab
~/pipefishRNA/stringtie/pipefish2
> mv ~/pipefishRNA/stringtie/pipefish2_cov_refs_merged.gtf
~/pipefishRNA/stringtie/pipefish2
```

19. In the directory that is equivalent to my **`~/pipefishRNA/`** (this might include your **`hisat_index`** and **`stringtie`** folders), we are going to create a text document describing the location of our newly organized files. You can either use a text editor like *Notepad ++* or use a program like *nano* for in line editing.

```
> nano sample_lst.txt
#We are now in a blank nano environment. Enter in the following
for each sample (make sure you use your sample name and path):
samplename /path/to/samplename.gtf
```

#Each sample should have its own line, and your document should look similar to the screenshot below. To save press 'CTRL' and 'X'. It will ask if you "Save modified buffer?", press 'Y'. It will say "File Name to Write: sample_lst.txt", press 'ENTER'.



```
  GNU nano 4.8                  sample_lst.txt                    Modified
pipefish1 stringtie/pipefish1/pipefish1_merged.gtf
pipefish2 stringtie/pipefish2/pipefish2_merged.gtf
```

> cat sample_lst.txt

#This command can be used to check the contents of the file and make sure that this step was successful.

20. Check for or install Python 3 first.

    > python3 --help #Check for installation, if its not installed proceed with the next command

    > sudo apt-get install python3.6 #This was the version at the time of the protocol

21. Check to make sure you have the following files within your working directory: *stringtie_merged.gff, prepDE.py3* (this file is from StringTie), *sample_lst.txt*, and your folder that contains all your sample outputs generated in step 18 (mine is called *stringtie*). Now we will run the StringTie python script using Python.

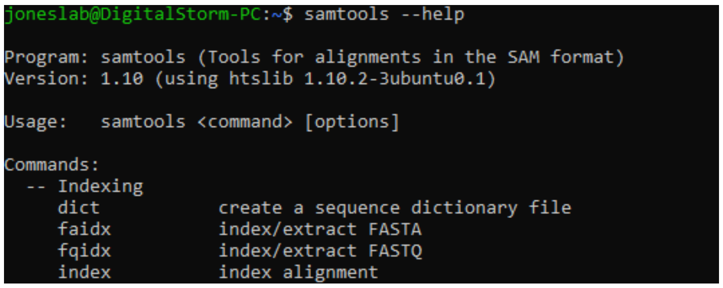
    > python3 prepDE.py3 -i sample_lst.txt

    #If it produces a file called gene_count_matrix.csv, then it was a success. This file contains count matrices for genes and transcripts. This is the file you will use for your differential expression analysis in EBSeq, DESeq2, or edgeR.

Tips on installing programs

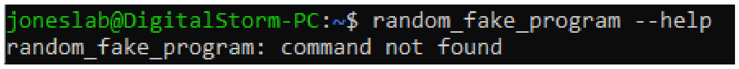
In this How-To document, we install several programs using the command line. A quick and easy way to check to see if a program is properly installed is to ask the program to display its usage and commands guide. For example, if we wanted to check our samtools installation we would use:

```
> samtools --help
```



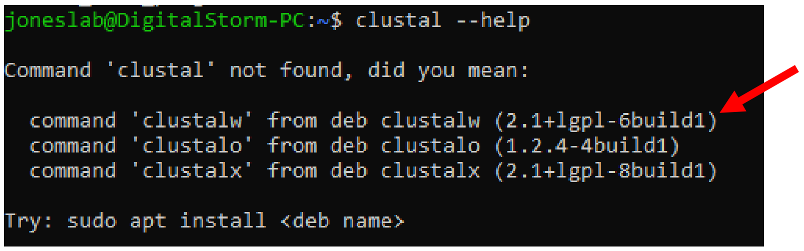
```
joneslab@DigitalStorm-PC:~$ samtools --help

Program: samtools (Tools for alignments in the SAM format)
Version: 1.10 (using htslib 1.10.2-3ubuntu0.1)

Usage:   samtools <command> [options]

Commands:
  -- Indexing
     dict           create a sequence dictionary file
     faidx          index/extract FASTA
     fqidx          index/extract FASTQ
     index          index alignment
```

If a program is not properly installed, you will instead see a message like this:



```
joneslab@DigitalStorm-PC:~$ random_fake_program --help
random_fake_program: command not found
```

If a program is not properly installed, but Linux knows a program with a similar name, you will instead see a message like this:



```
joneslab@DigitalStorm-PC:~$ clustal --help

Command 'clustal' not found, did you mean:

  command 'clustalw' from deb clustalw (2.1+lgpl-6build1)
  command 'clustalo' from deb clustalo (1.2.4-4build1)
  command 'clustalx' from deb clustalx (2.1+lgpl-8build1)

Try: sudo apt install <deb name>
```

If Linux is suggesting the correct program, then most times you can go ahead with its suggestion and install. You should note the version (arrow) and make sure that is the correct version you want to install. Linux might suggest an older version than what is currently available. If in this case I accept the suggestion, I would enter the following command:

```
> sudo apt install clustalw
```