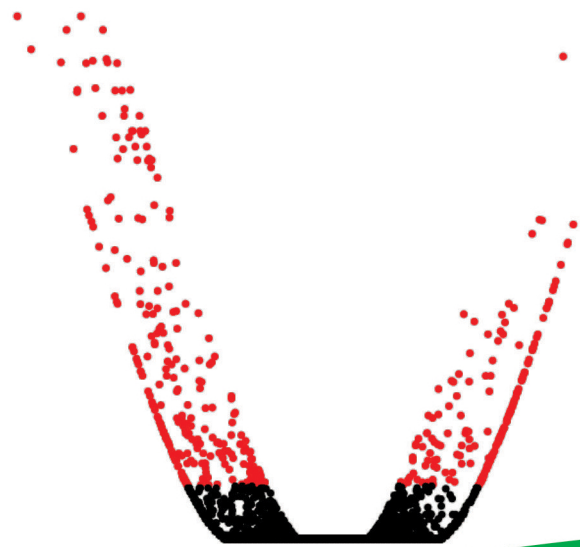
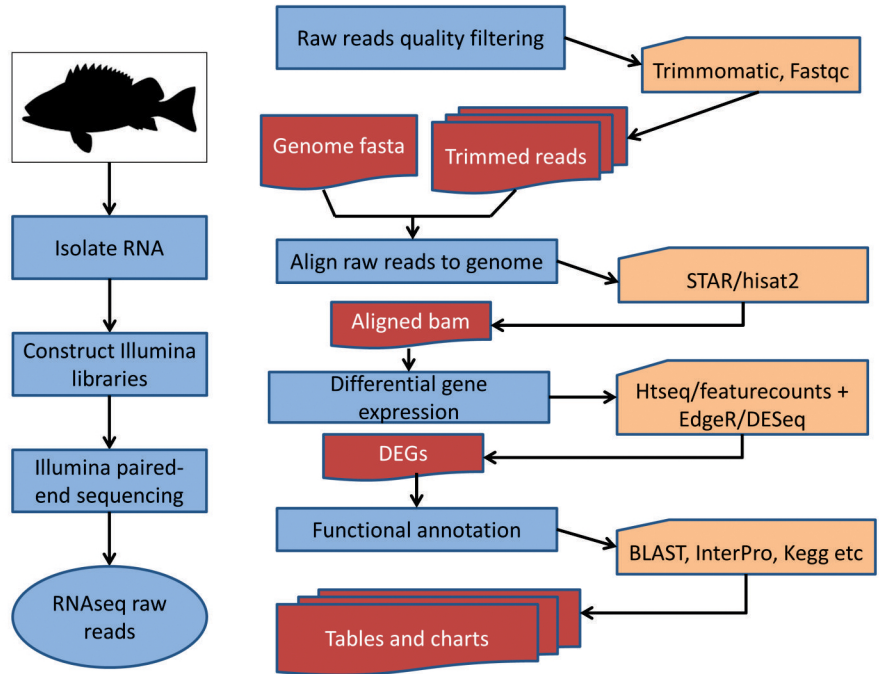
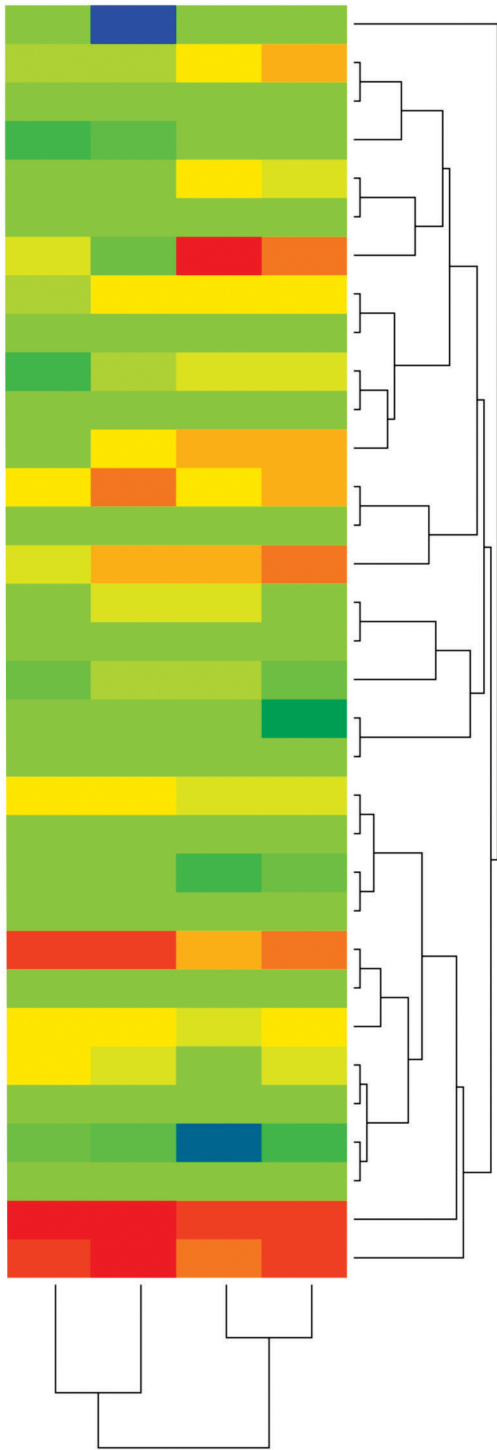




# Training Manual on RNA-Seq Data Analyses



**Manual**  
**RNAseq Data Analyses – Hands on Training**

Organized By

**GENETICS AND BIOTECHNOLOGY UNIT**

Prepared by

<b>K. VINAYA KUMAR</b>	<b>J. ASHOK KUMAR</b>	<b>B. SIVAMANI</b>
<b>K. P. GANGARAJ</b>	<b>K. P. SUDHEESH</b>	<b>K. KARTHIC</b>
<b>MISHA SOMAN</b>	<b>RAYMOND J ANGEL</b>	<b>SHERLY TOMY</b>
<b>M. S. SHEKHAR</b>		

**ICAR – CENTRAL INSTITUTE OF BRACKISHWATER AQUACULTURE**

**75, SANTHOME HIGH ROAD, RA PURAM**

**MRC NAGAR, CHENNAI 600028**

**Published by**

**Dr. K. P. Jithendran**

**Director, ICAR-CIBA**

## Table of Contents

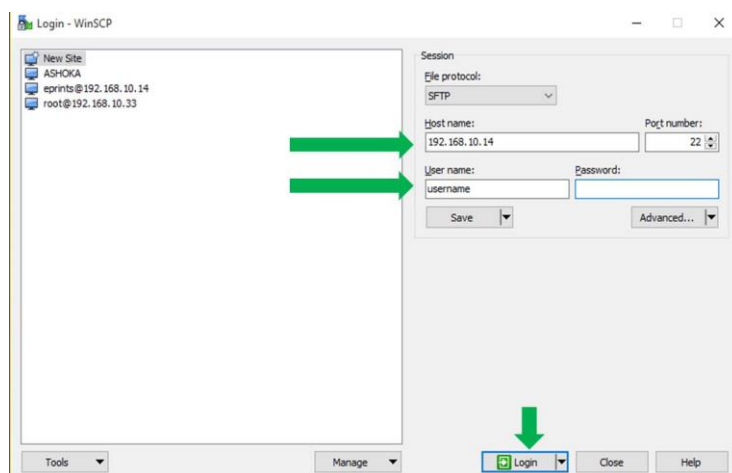
<b>S. No.</b>	<b>Chapter</b>	<b>Page No.</b>
1.	Essential Linux commands for running bioinformatics programs	1
2.	Introduction to programming in R	5
3.	Understanding the Illumina datasets	13
4.	Checking quality of Illumina paired-end sequence datasets	19
5.	Quality trimming of RNAseq datasets – Trimmomatic	21
6.	Discovery of unigenes using RNAseq data	24
7.	Genome-guided analysis with RNA-seq data	28
8.	Annotation	44



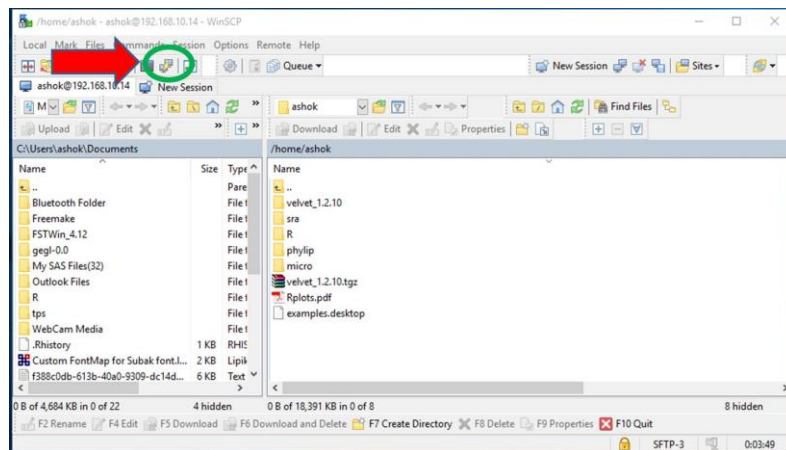
Most of the bioinformatics programs are Linux based and expertise is required in running Linux commands for submitting bioinformatics jobs. Linux, an open source operating system (OS) originally built based on Unix has become choicest OS worldwide for servers as well as desktops in academic circles. There are different variants of Linux which include Debian, Redhat, Ubuntu, Fedora, CentOS, knoppix etc. Many bioinformatics tools which require high performance computing facility are native to Linux OS. So it is important for a bioinformatician to have exposure to Linux commands. Here we give a list of most commonly used linux commands and procedure to execute Perl /python programs. As advanced programming is beyond the scope of this training, we provide here the basic constructs of Perl/python programs which could be used for writing scripts for simple bioinformatics tasks.

## Linux commands

**Accessing the Linux environment:** You can access Linux server using any windows based ssh client from your system. This could be achieved by installing winSCP or Putty (both are free software) on your system. Once installed WinSCP, open and enter the Host name (or IP address of the server), user name columns provided by your system administrator and click on login button which will prompt for password. After successful login and selecting putty from menu bar, command line console window pops up and you will see a Dollar prompt where in you can submit commands for all the operations you wish to perform on the Linux server.



WinSCP login window



Selecting Putty from winSCP

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
ashok@192:~$ █
```

### Linux console

The dollar prompt (\$) shown in Fig. 3 is for users and the hash (#) prompt will be displayed for administrators. Users who have the administrative privileges on the server can only work with hash (#) prompt.

File system in linux: All the folders and files of the Linux system will be under root (/) directory. Users will have access to their home directories for which the path looks like /home/user\_name

Once you login to the Linux system by default you will be taken to your home directory. For example is if the user name is “**david**”, after login into Linux the current directory which he will be accessing is /home/david. Users can input their commands after the dollar (\$) prompt. Some of the most commonly used Linux commands are given in the table below.

Function	Command
Listing the file names	\$ls
Listing with file names along with other details	\$ls -l
Change to preexisting directory by name 'test'	\$cd test
Make a new directory by name 'trial'	\$mkdir trial
Viewing a preexisting file	\$vi mydata.txt \$nano mydata.txt

	\$more mydata.txt \$cat mydata.txt
Creating a new file	\$touch myfile.txt \$vi myfile.txt \$nano myfile.txt
Renaming or moving the file	\$mv file1.txt file2.txt \$mv /home/ram/file1.txt /home/ram/test/
Making duplicate of file	\$cp file1.txt file2.txt \$cat file1.txt > file2.txt
Appending two text files	\$cat file1.txt file2.txt > file3.txt
To display date	\$date
To find number of lines in a file	\$wc -l xyz.txt
To display first (top) 100 lines of a file	\$head -100 xyz.txt
To display last (bottom) 100 lines of a file	\$tail -100 xyz.txt
Search for a pattern in a file	\$grep "pattern" file.txt
Search for pattern at beginning of line	\$grep '^pattern' file.txt
Search for pattern at the end of a line	\$grep 'pattern\$' file.txt
Search for only pattern in the line	\$grep '^pattern\$' file.txt
Copy required columns from one file to other file	\$cut -f 1,2,3 abc.txt > xyz.txt
Sort contents of the file	\$sort file.txt > file1.txt
Remove duplicate entries from the file	\$uniq file.txt > file1.txt

### Running perl /python programs.

Perl program files will have extension “.pl”. Command to execute the programmes is

```
$ ./test_programme.pl
```

Or

```
$perl test_programme.pl
```

Options of the program may be checked from the help files of the software/programs.

Same way python program files will have “.py” extension and they could be executed by giving following command.

```
$python test_programmes.py
```

### Standalone blast:

NCBI Blast is used for comparing nucleotide and protein sequences with the sequence databases to find significant matches. Alignment of sequences using blast can be done either by using web-tool available on NCBI site or by installing blast on local servers.



Blast can be installed on local servers along with the databases available in public domain. In addition, users can make their own databases on local servers. If you have your own protein dataset then local databases can be created by

```
$makeblastdb -in xyz.fasta -dbtype 'prot' -out xyzdb
```

Now you can run the blast using your own database

```
$blastp -db xyzdb -query abc.fasta -out out.fasta
```

More general blast Command

```
$blastn -query nucl.fasta -db xyzdb -outfmt 6 -evalue 1e-05 -out  
output.txt
```

For fetching the sequences in fasta file format from output make a file with IDs of hits and run the following command

```
fastacmd -d database_name -i blast_output > hits.fasta
```

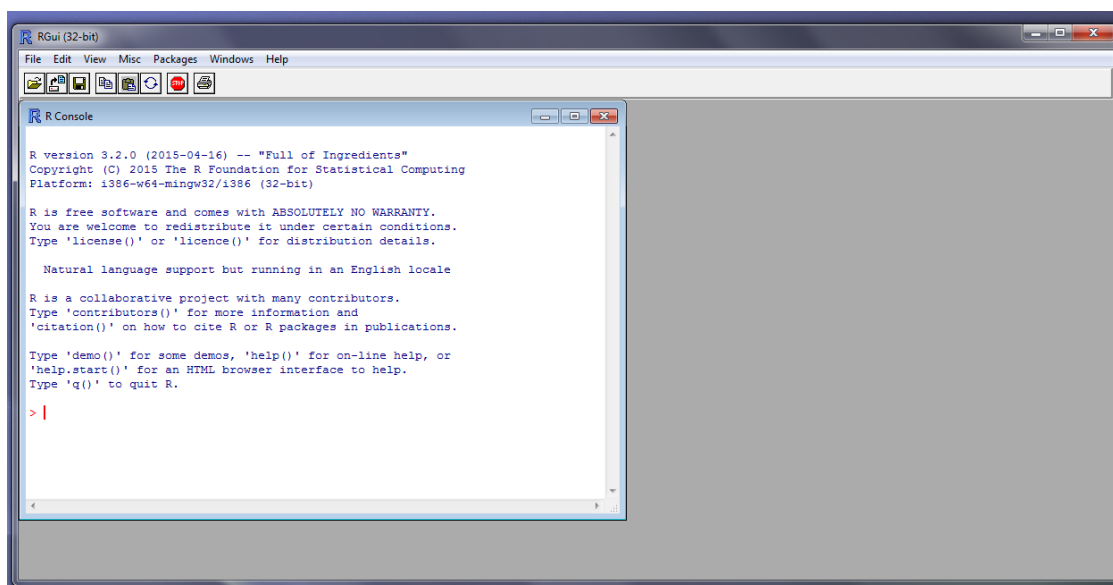
R is a programming environment for data analysis and graphics. The language was initially written by Ross Ihaka and Robert Gentleman at the Department of Statistics at the University of Auckland. Since its birth, a number of people have contributed to the package. It is open source statistical software which can be downloaded free of cost. Base package and all the contributory packages could be downloaded from <http://www.r-project.org/>

R is available for all operating systems like windows, Linux and Mac OS. This training material is based on R stats package installed in windows operating system.

## Invoking R stats

Start → All programmes → R → R i386 4.2.0 (for 32 bit installation)

Start → All programmes → R → R x64 4.2.0 (for 64 bit installation)



R Stats Graphical user interface in windows

## Procedure to install additional packages

We need to add additional libraries to Base installation to utilize full potential of R. This can be achieved by following command.

### ➤ `Install.packages('name of the package')`

Once the above command is executed R system asks the user to select a CRAN mirror out of several listed mirrors. User can select mirror of any location.

There is a package/library called 'Rcmdr' which can be used for carrying out most commonly used statistical procedure with graphical user interface. The command to install 'Rcmdr' is

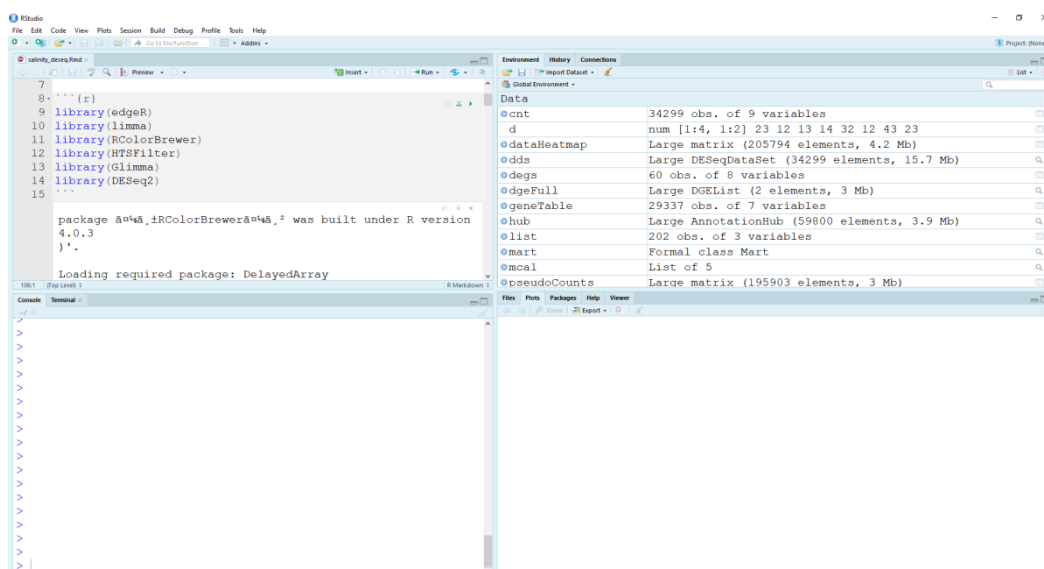
### ➤ `Install.packages('Rcmdr')`

Command to invoke the Rcmdr

### ➤ `Library('Rcmdr')`

## R studio

R studio is integrated development environment (IDE) for R. This IDE features R notebook for writing scripts, console for command input, graphics viewer, package window and environment window all in single framework. R studio has facility to create R notebooks in which R commands can be written and run in chunks. These notebooks can be saved for later use.



Snapshot of R environment

## R files input and output.

First set the working directory

Command to know the location of present working directory is

➤ **getwd()**

Command to set the working directory to any other folder

➤ **setwd("E:/data/")**

Basic command to read the files is

➤ **read.table()**

and command to create the data files is

➤ **write.table()**

## Importing data

Data with different file formats i.e., text files, excel files, SPSS data files, SAS data files etc., can be input into R stats for data analysis. It is advised that excel files may first be converted to comma separated files for easy input into R stats.

Command to read a comma separated text file with variable names in the first row

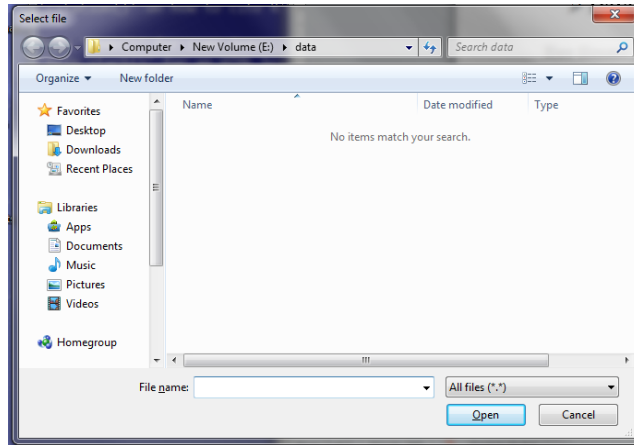
➤ **Data <- read.table('filename', header=TRUE, sep=",")**

Here filename is name of the text file with extension, header statement is to specify whether variable names are included in the first row of the data file and 'sep' parameter tells the separator present between variables (columns) like comma, space, tab etc., in the file.

If the specified text file is not in present working directory and you wish to select it through graphical interface use the following command

➤ **Data <- read.table( file.choose(), header=TRUE, sep=",")**

Upon entering the above command a file selector window will pop up and one can select the file located at any drive/directory/folder other than the present working directory.



Popup window for selecting files

For other text files like space separated and tab separated one need to change only ‘sep’ parameter of the above command with either “ ” or “\t”.

In the previous command ‘data’ is a dataframe which will contain all the variable names and data

Data in the dataframe can be edited and assigned the changed file contents to other dataframe

```
➤ data1<- edit(data)
```

Upon entering the above command a popup window appears for editing the data and all the edits will be saved in data frame called ‘data1’

	date	price	var3	var4	var5	var6	var7
1	01-Jan-95	234.5933					
2	01-Feb-95	236.287					
3	01-Mar-95	238.3368					
4	01-Apr-95	236.5613					
5	01-May-95	243.2256					
6	01-Jun-95	243.1197					
7	01-Jul-95	242.9369					
8	01-Aug-95	232.8706					
9	01-Sep-95	232.8755					
10	01-Oct-95	227.3323					
11	01-Nov-95	224.8783					
12	01-Dec-95	224.4019					
13	01-Jan-96	229.5794					
14	01-Feb-96	235.0023					
15	01-Mar-96	221.101					
16	01-Apr-96	219.7778					
17	01-May-96	227.1254					
18	01-Jun-96	232.3365					
19	01-Jul-96	229.9162					

Data editor window

## Exporting data

Data in the dataframe can be exported as a text file with the following command

- `write.table(data, file="xyz.csv", col.names=TRUE, sep=",")`

## Creating data files manually within Rstats

Data files can be created within Rstats by giving simple commands

Here we explain creating example table with variable names into R stats

S.No	Bodyweight	length	species
1	25	15	aa
2	35	14	ab
3	65	27	ac
4	27	18	bb
5	45	22	cc

The above table can be created as a dataframe by giving the following commands

- `bodyweight <- c(25,35,65,27,45)`
- `length <- c(15,14,27,18,22)`
- `species<-c("aa","ab","ac","bb","cc")`
- `lengthweight <-cbind(bodyweight,length,species)`

## Descriptive statistics

Suppose we have a variable by name 'x' and our task is to calculate all the descriptive statistical parameters like mean, median, standard deviation, variance etc. for the variable x in

R stats. First create a variable x by giving the following command

- `x <- c(20,15,19,22,26,24,23,17,18,22)`

### Other way of creating variable 'x' is

- `x <- scan()`  
1: 20 15 19 22 26 24 23 17 18 22  
11:  
Read 10 items

## Basic commands for descriptive statistics

- mean (x) # mean
- median (x) # median
- var (x) # sample variance
- sd(x) # sample std. deviation
- quantile (x,p) # sample quantile , p could be 0.25, 0.5,0.75
- min (x) # minimum of x
- max (x) # maximum of x
- range () # range of x
- library(e1071)
- skewness (x) # skewness
- kurtosis (x) # kurtosis

## Commands for statistical tests

### Single sample t-test

- t.test(y,mu=10)  
here y is a variable; mu is population mean

### Two sample t-test

- t.test(y1,y2, var.equal=TRUE)

y1 and y2 are the two independent samples

### Paired t-test

- t.test(y1,y2,paired=TRUE)  
y1 and y2 are the two paired samples

### Chi-square test for goodness of fit

- n<- cbind(y1,y2)
- chisq.test(n)

n is a datamatrix /contingency table

## Correlation

- `n <- cbind(y1,y2) # create dataframe n`
- `cor(n)`

where y1 and y2 are two variables and n is matrix of y1 and y2

## Regression

- `fit <- lm(y~x)`

for multiple regression

- `fit <- lm(y~x1+x2+x3)`

## Completely randomised design

- `tr <- c(1,1,1,2,2,2,3,3,3) # create treatment variable`
- `yield<-c(25,41,54,65,45,65,25,12,35) # create dependent variable`
- `fit <- aov(yield ~ factor(tr)) # model statement`
- `summary(fit)`

## Randomised Block Design

- `tr <- c(1,1,1,2,2,2,3,3,3) # create treatment variable`
- `rep <-c(1,2,3,1,2,3,1,2,3) # create replication variable`
- `yield<-c(25,41,54,65,45,65,25,12,35) # create dependent variable`
- `fit <- aov(yield ~ factor(tr) + factor(rep))`
- `summary(fit)`

## Two way factorial Design

- `fit <- aov(yield ~ factor(A) + factor(B) + factor(A) : factor(B) + factor(rep))`
- `summary(fit)`



## Installing Bioconductor in R

Enter following commands in R console to install bioconductor packages.

```
if (!require("BiocManager", quietly = TRUE))
  install.packages("BiocManager")

BiocManager::install("edgeR")
```

There are several libraries are available in R for different bioinformatics procedures. The purpose of this chapter is to introduce the R environment and to provide hands-on for exploring the functionalities available in R.

\*\*\*

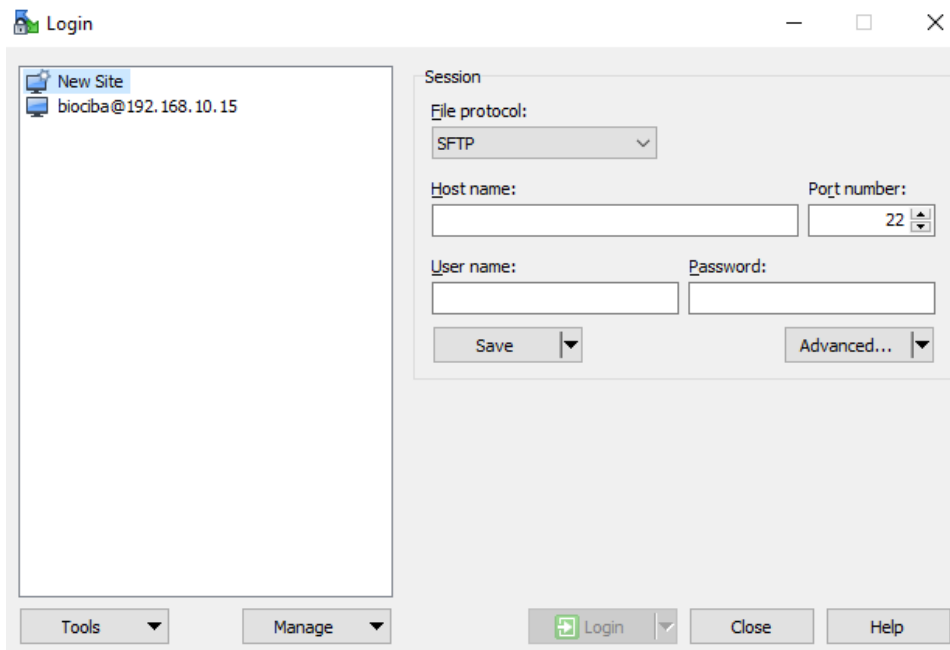
The Next Generation Sequencing (NGS) platforms have evolved over the past decade to generate high quality and high throughput sequence data at low cost and less time. The popular NGS sequencing platforms include Illumina, Pacbio, Nanopore, Ion Torrent etc. as evidenced from recently published manuscripts. A feature common to all these platforms is massively parallel sequencing of single or clonally amplified DNA molecules. Of different platforms available till date, the one offered by Illumina is more popular for RNA data. In case of Illumina, right from the Genome Analyzer Iix, the HiSeq XXXX series, the MiSeq, the NextSeq XXX series to the latest NovaSeq 6000, there is an improvement in data output while reducing the sequencing time. Whereas, the Iso-sequencing application of Pacbio platform is getting popular in recent years to generate isoform-level full-length transcript sequences.

### **Paired-end sequencing:**


Paired-end sequencing is one of the popular sequencing chemistry of Illumina platform with applications like, finding differentially expressed transcripts in experimental samples compared to control sample, establishing unigenes in various tissues/development stages/species etc. In this chapter, we understand the structure of paired-end sequence datasets generated on Illumina platform. The raw sequence data files generated on Illumina platform are delivered as fastq files (which have the extension, .fastq or .fq). For every sample, two files are provided, one read\_1 (or forward read/ pass 1/ left read) and the other read\_2 (or reverse read/ pass 2/ right read). The order of reads in forward and reverse sequence reads files should not be altered as they are linked.

### **Download datasets from GenBank:**

Here, we shall learn to download datasets through command-line interface. First, connect to the server using WinSCP tool. Open the WinSCP application (you should see the following window) and enter the host name as told by the tutor. Enter the 'user name' and 'password' as shared by the tutor to login to your user account.



After logging in, the window of WinSCP tool appears. The window has two panels. The left panel is the file system of your computer. The right panel is the file system of your user account in the server.

Click on the icon displaying ‘two connected computers’  in the top toolbar to open the putty window where we run jobs. Enter the log in credentials on prompt.

Separately, open any web browser and access NCBI Genbank page. Browse the Bioproject, PRJNA494937. Notice that this bioproject has 8 SRA (Sequence Read Archive) datasets. The accession, SRX4808138 is the paired-end RNAseq data generated using muscle tissue of *Panaeus indicus*. The run ID for this accession is, SRR7975326. We shall download this accession and learn some applications.

The fastq-dump argument is used to download the NCBI datasets in command-line mode. Let us install the required software, SRA toolkit. Open a search page and visit the page, <https://www.ncbi.nlm.nih.gov/sra> (you may use the key word, download sra toolkit for search). Here, click on the ‘Download SRA Toolkit’ under *Tools and Software* to open the download page. Then, we shall copy the link address for the appropriate version of SRA toolkit software which in our case would be, [Ubuntu Linux 64 bit architecture](#) as our servers are linux-based.

Use the following argument to download the software directly through command-line interface,

```
wget<>paste_link_address
```

You should see the following screenshot after successful download and find a tar ball file (file.tar.gz) in your folder.



```
vinay@192:~$ wget https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/3.0.0/sratoolkit.3.0.0-ubuntu64.t
--2022-06-01 09:46:38-- https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/3.0.0/sratoolkit.3.0.0-ubu
Resolving ftp-trace.ncbi.nlm.nih.gov (ftp-trace.ncbi.nlm.nih.gov)... 130.14.250.11, 130.14.250
Connecting to ftp-trace.ncbi.nlm.nih.gov (ftp-trace.ncbi.nlm.nih.gov)|130.14.250.11|:443... co
HTTP request sent, awaiting response... 200 OK
Length: 85777118 (82M) [application/x-gzip]
Saving to: 'sratoolkit.3.0.0-ubuntu64.tar.gz'

sratoolkit.3.0.0-ubuntu64.tar.gz      100%[=====]
2022-06-01 09:50:21 (378 KB/s) - 'sratoolkit.3.0.0-ubuntu64.tar.gz' saved [85777118/85777118]
vinay@192:~$ █
```

Then extract the tar ball with the following command,

```
tar<>xvzf<>sratoolkit.3.0.0-ubuntu64.tar.gz
```

In your user account go to the *bin* sub-folder of 'sratoolkit.3.0.0-ubuntu64' folder. Then use the following command,

```
./vdb-config -interactive
```

You will see the following window popped-up.

```

vinay@192: ~/sratoolkit.3.0.0-ubuntu64/bin
SRA configuration
[ save ] [ exit ] [ discard ] [ default ]
MAIN CACHE AWS GCP NET TOOLS
[X] Enable Remote Access
[ ] Prefer SRA Lite files with simplified base Quality scores

GUID: 7d6cad08-1446-42bb-aad2-91c867757b7f

save configuration

```

Use tab buttons to save and exit the window. Now, the tool is ready to download the sequence datasets. Proceed with the following argument to download muscle tissue RNAseq data. The symbol, <> in the argument denotes space.

```
/fastq-dump<>-I<>--split-files<>--gzip<>--defline-qual<>'+'<>SRR7975326
```

You find that a pair of files are downloaded, one forward and one reverse reads file. The files are of *fastq* type and compressed with *gzip*. Unzip the files with the following command lines. Here, the argument ‘-c’ retains the original zipped file. This can be skipped if you do not need the original file in zip format. We are identifying the forward reads file as ‘F.fastq’ and reverse reads file as ‘R.fastq’.

```
gunzip<>-c<> SRR7975326_1.fastq.gz > F.fastq
```

```
gunzip<>-c<> SRR7975326_2.fastq.gz > R.fastq
```

### Check the file format for RNAseq dataset:

Let us see the first few lines of forward reads file and understand the structure of fastq files. Use the following command,

```
head<>F.fastq
```

```

vinay@192:/SANbackup/vinay/training$ head F.fastq
@SRX4808138.1 1 length=150
CCGGTAGTACGACCAGAGGGCTAGAGGGAGAGCACGGCCTGGATGGTAACATAGGTGGCAGGTACGTTGAAAGACTCAAACATGATCTG
+
AAAAAEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE/EEEEEEAA/EEEEEEEE
@SRX4808138.2 2 length=150
GGACTGGACATTTATCATGGAATATCCATGTGTAAACAACGTATATTTGACAGGCATTTTCAGGTGCCTAAAGTGACGCATTTTTTCT
+
AAAAAEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
@SRX4808138.3 3 length=150
CGATGGCAGAGCCTTTCAGGGCCTGCTCGGCCGCTCTATGTGCATACGTATGACAAAGGCCCGAGTCGTAGCTGTTATCCTGCTGT

```

You find that, the information about each sequence read is represented in four lines.

Line 1: has information about the accession ID, read number and read length.

Line 2: the sequence of the read which is the familiar A, T, G and C

Line 3: a plus (+) sign

Line 4: the quality scores of the sequence bases

You may also try the commands, `more<>F.fastq` AND `tail<>F.fastq`

We have downloaded these RNAseq datasets from Genbank. But, if you have received the same datasets from a sequencing company, you find additional information in line 1. This includes, instrument ID, run ID, flow cell ID, lane ID, tile ID, X and Y coordinates of clusters, read number, status about the read is filtered or not and control sample status etc.

You may visit the following page to understand more about the quality scores.

[https://www.illumina.com/documents/products/technotes/technote\\_understanding\\_quality\\_scores.pdf](https://www.illumina.com/documents/products/technotes/technote_understanding_quality_scores.pdf)

The symbols in line 4 represent quality scores of bases. The quality scores ranges from 0 to 40. A score of 40 indicates that the base called is of high quality. In this case, the error probability infers that one base call in 10,000 base calls would be incorrect. The following table illustrates the relation between the symbols and the corresponding quality scores.

Table 1. List of symbols corresponding to quality scores of bases in Illumina sequence datasets.

Symbol	Quality Score
!	0
"	1
#	2
\$	3
%	4
&	5
'	6
(	7
)	8
*	9
+	10
,	11
-	12
.	13
/	14
0	15
1	16
2	17
3	18
4	19
5	20

Symbol	Quality Score
6	21
7	22
8	23
9	24
:	25
;	26
<	27
=	28
>	29
?	30
@	31
A	32
B	33
C	34
D	35
E	36
F	37
G	38
H	39
I	40

\*\*\*

Illumina paired-end (PE) sequencing reads are commonly used for RNAseq studies and assembling of genomes. As you have learned from the previous chapter the sequencer prints output data in two paired *.fastq* files. In this chapter, we discuss about the quality issues pertaining to PE reads. A better understanding of these helps in better planning of read processing to extract quality data for further studies.

One of the basic software useful to understand the quality of PE reads file is 'FastQC'. The software is available for download at the following link.

<https://www.bioinformatics.babraham.ac.uk/projects/download.html#fastqc>

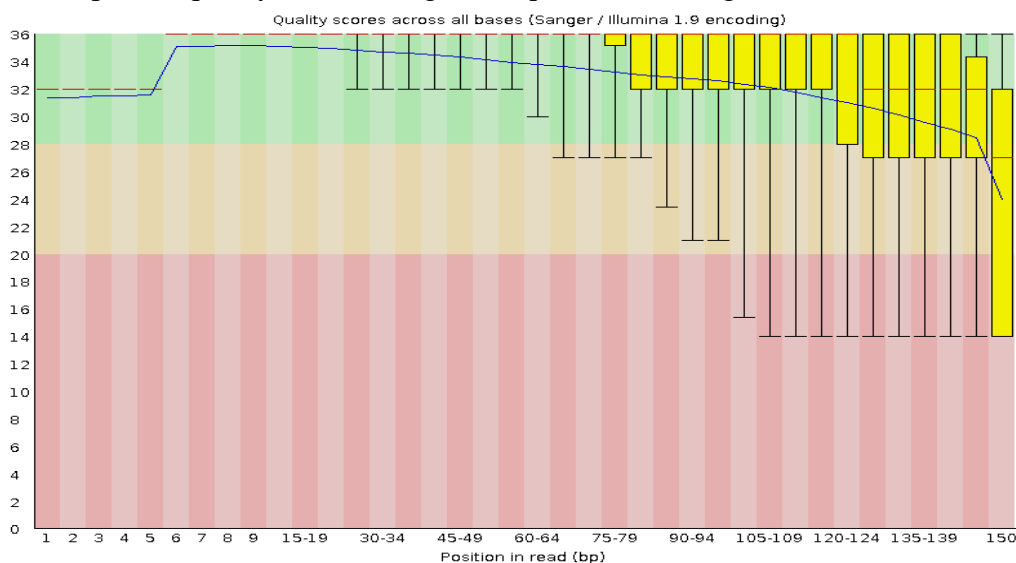
Download 'fastqc\_v0.11.9.zip' and save it to your user account in server through winscp. Unzip the file. Then, make the program executable with '`chmod <>+x <>*`'.

We shall check the quality of the dataset that we have downloaded in the previous chapter. In your account, find two files, F.fastq and R.fastq. We shall check the quality of these files using FastQC tool. To do this, run the following command at your prompt.

```
$ /fastqc <space> F.fastq
```

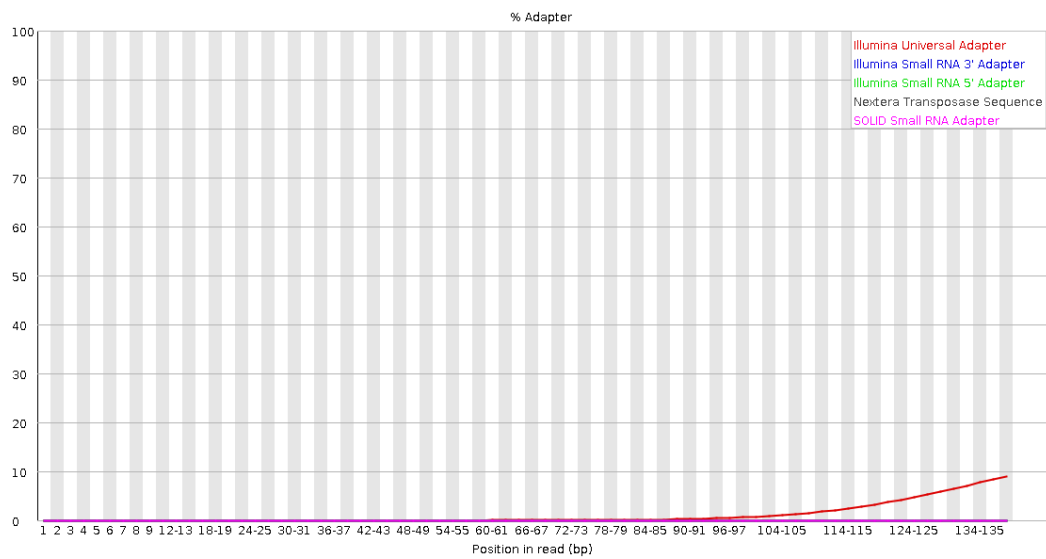
In about five minutes, the analysis would be completed and two output files are printed, F\_fastqc.html and F\_fastqc.zip. Save these files to your computer and open the .html file in any browser. Check all images and understand their meaning. Observe carefully for the following aspects in the file.

1. The file contains 29,463,688 sequence reads of 150 bp length. This number shall tally with the number that you had observed with the *tail* command.
2. Box plot of quality scores along the sequence read length.





3. The reads are contaminated with adapter sequences used during sequencing.



The quality report warrants us to do some data processing which includes,

1. Removal of poor quality reads that are pulling down the average of quality scores.
2. Removal of poor quality bases at end of sequence reads.
3. Removal of adapter sequences contaminating the reads.

Perform similar exercise for reverse reads file and make an assessment of the requirements for quality trimming.

\*\*\*

There are several freeware available for processing of paired-end sequence reads. In this chapter we shall use ‘Trimmomatic’ for quality control of PE reads. First, *log in* to your account using WinSCP tool. Open PuTTY SSH terminal. In your account, find two files named, F.fastq and R.fastq. You must have already checked the quality of both the paired files using FastQC tool.

Download the ‘trimmomatic’ tool and copy the same in to your user account at the server. The current version is 0.39. Copying the entire folder in to your user account is enough. There are no other installation requirements for trimmomatic tool.

Run the following command and observe the changes in quality of trimmed files. The ‘<>’ sign used in the command argument indicates ‘space’.

The command:

```
java<>-jar<>path_to_trimmomatic-0.36.jar<>PE<>-threads<>10<>-
trimlog<>log.txt<>F.fastq<>R.fastq<>F_P.fastq<>F_S.fastq<>R_P.fastq<
>R_S.fastq<>ILLUMINACLIP:path_to_Truseq3-PE-
2.fa:2:30:10<>LEADING:3<>TRAILING:13<>SLIDINGWINDOW:4:15<>MINLEN:100
```

De-coding the command:

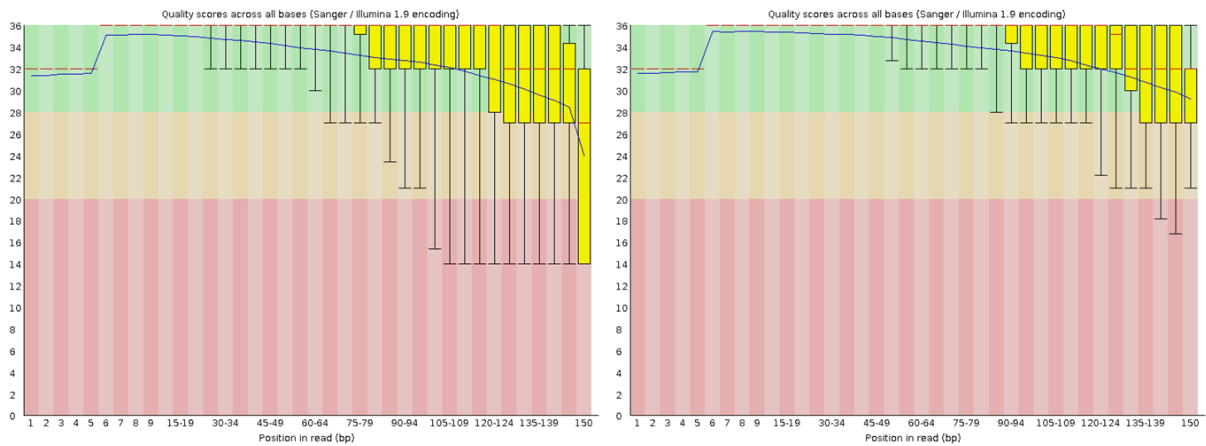
Each argument in the command has a purpose of improving the quality of trimmed files. It is important to check the initial quality of sequence data and then apply the relevant arguments to improve the quality.

Argument	Meaning
PE	Paired-end mode. Use this for processing of PE reads data
threads	The argument to specify number of threads. Trimmomatic supports running arguments with multiple threads.
trimlog	To specify a file name that stores log of the run.
F.fastq	Input file name of forward or R1 reads
R.fastq	Input file name of reverse or R2 reads
F_P.fastq	Output file name of trimmed forward or R1 reads. This file is used for subsequent analysis.
F_S.fastq	Output file containing surviving forward reads of good quality. The paired sequences in R2 file are discarded.
R_P.fastq	Output file name of trimmed reverse or R2 reads. This file is

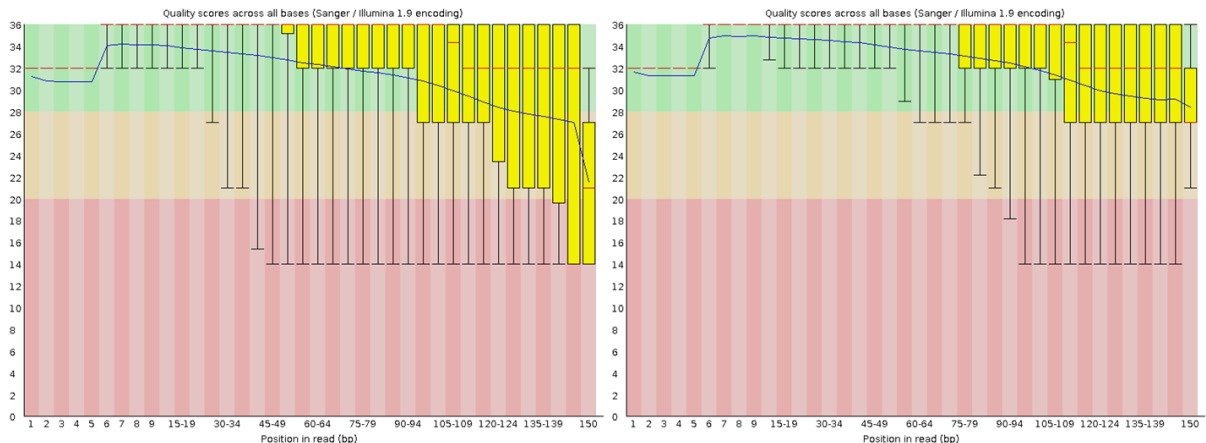
	used for subsequent analysis.
R_S.fastq	Output file containing surviving reverse reads of good quality. The paired sequences in R1 file are discarded.
ILLUMINACLIP:TruSeq3-PE-2.fa:2:30:10	Illuminaclicp is used to remove adapter sequences from reads. The TruSeq3-PE-2.fa is the file containing adapter sequences.
LEADING:3	To remove bases at the start of the read, if quality is below 3
TRAILING:13	To remove bases at the end of the read, if quality is below 13
SLIDINGWINDOW:4:15	This is an argument that trims reads based on base quality. Each read is scanned from 5' end. Four continuous bases are taken as a window. The average quality of all windows in a read should be higher than 15. Otherwise, the read gets trimmed from poor quality window to the 3' end of the read.
MINLEN:100	To discard reads shorter than 100 bases after performing all the steps.

Check the quality of trimmed files with FastQC.

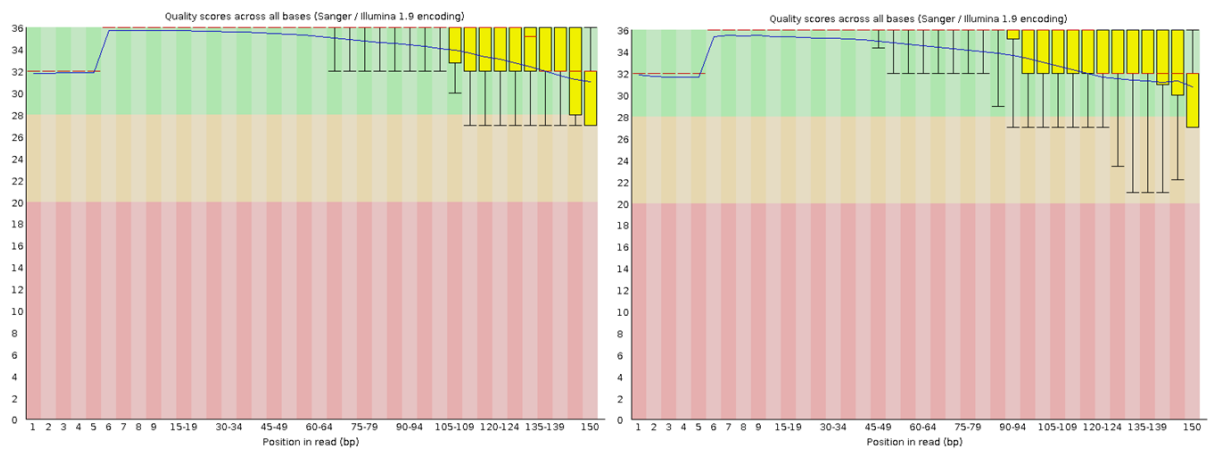
Below are the quality of forward sequence reads before (left) and after (right) trimming.



Below are the quality of reverse sequence reads before (left) and after (right) trimming.



The extent of trimming that we have performed is not good enough. There are bases with a quality score of less than 20. Therefore, we shall repeat trimming with further stringent criterion. We shall change ‘SLIDINGWINDOW’ to ‘4:22’. Below are the qualities of forward sequence reads (left) and reverse sequence reads (right) after trimming with changed parameters.



You may observe that the adapter contamination is also removed in trimmed reads. Now, these trimmed files would be taken up for building transcriptome assembly.

\*\*\*

One of the major applications of RNAseq is the discovery of unigenes in a species. Till an annotated whole genome is made available, researchers were dependent on the RNAseq data to establish transcriptome. In this chapter we shall generate a transcriptome using RNAseq data derived using muscle tissue of *P. indicus*. The same datasets were used for quality trimming in previous sessions. We shall use Trinity to generate a *de novo* transcriptome assembly.

Use the following argument:

```
Trinity<>--seqType<>fq<>--left<>F_P.fastq<>--right<>R_P.fastq<>--
CPU<>70<>--max_memory<>350G<>--SS_lib_type<>FR<>--
output<>trinity_muscle
```

The command arguments details are,

Argument	Meaning
Trinity	
seqType<>fq	Input files are in <i>fastq</i> format
--left<>F_P.fastq	The left or forward reads file name is F_P.fastq
--right<>R_P.fastq	The right or reverse reads file name is R_P.fastq
--CPU<>70	Use 70 threads
--max_memory<>350G	Limit maximum memory to 300 GB
--SS_lib_type<>FR	Data obtained from strand-specific library as forward and reverse reads
--output<>trinity_muscle	Output is stored in this folder

The assembly is completed when you see the messages printed as shown below.

```
Number of Commands: 43397
succeeded(43397) 100% completed.

All commands completed successfully. :-)

** Harvesting all assembled transcripts into a single multi-fasta file...

Monday, May 30, 2022: 19:18:52 CMD: find /SANbackup/vinay/training/trinity_muscle/read_partitions/ -name '*inity.
rinity/opt/trinity-2.12.0/util/support_scripts/partitioned_trinity_aggregator.pl --token_prefix TRINITY_DN --outpu
y_muscle/Trinity.tmp
-Relocating Trinity.tmp.fasta to /SANbackup/vinay/training/trinity_muscle/Trinity.fasta
Monday, May 30, 2022: 19:18:53 CMD: mv Trinity.tmp.fasta /SANbackup/vinay/training/trinity_muscle/Trinity.fasta

#####
Trinity assemblies are written to /SANbackup/vinay/training/trinity_muscle/Trinity.fasta
#####
```

Browse to the folder and find the assembled transcripts file, *Trinity.fasta*. Rename the file as ‘*muscle.fasta*’ for easy identification.

### Evaluate the quality of assembly:

1. N50 statistics: Compute N50 statistic by running the following command,

```
TrinityStats.pl<>muscle.fasta<>><>muscle_stats.txt
```

The transcriptome assembly was observed to contain 61,509 transcripts with an N50 of 2368 bases.

2. Read representation: The proportion of paired-reads represented in the assembled transcripts is another parameter that helps in evaluating the assembly. We shall use bowtie2 tool for this. First an index is to be made and then reads are to be aligned on to transcripts. Run the following two commands.

```
bowtie2-build<>muscle.fasta<>muscle.fasta
```

```
bowtie2<>-x<>muscle.fasta<>-q<>--fr<>-1<>F_P.fastq<>-2<>R_P.fastq<>-S<>samfile <>--no-unal<>-p<>20
```

```
15428899 reads; of these:
 15428899 (100.00%) were paired; of these:
 1022415 (6.63%) aligned concordantly 0 times
 5513389 (35.73%) aligned concordantly exactly 1 time
 8893095 (57.64%) aligned concordantly >1 times
-----
 1022415 pairs aligned concordantly 0 times; of these:
 254309 (24.87%) aligned discordantly 1 time
-----
 768106 pairs aligned 0 times concordantly or discordantly; of these:
 1536212 mates make up the pairs; of these:
 412007 (26.82%) aligned 0 times
 120796 (7.86%) aligned exactly 1 time
 1003409 (65.32%) aligned >1 times
98.66% overall alignment rate
```

As per the statistics shown above, the overall alignment rate is 98.66% which is good.

3. Transcriptome completeness assessment: We shall use BUSCO (Benchmarking Universal Single-copy Orthologs) scores. The BUSCO helps us in performing a quantitative assessment of transcriptome completeness. First, install the busco tool with the following arguments.

```
conda<>create<>-n<>busco
```

```
conda<>install<>-c<>conda-forge<>-c<>bioconda<>busco=5.3.2
```

Run busco tool on muscle transcriptome with the following argument,

```
busco<>-i<>muscle.fasta<>-m<>trans<>-l<>arthropoda_odb10<>-c<>20<>-o<>busco_muscle
```

The command arguments details are,

Argument	Meaning
busco	Calls busco tool
-i muscle.fasta	Input file name is muscle.fasta
-m trans	Run busco in transcriptome mode
-l arthropoda_odb10	Use BUSCO genes of arthropoda lineage. The transcriptome assembly belongs to shrimp that falls in Arthropoda
-c 20	Use 20 threads
-o busco_muscle	Output is stored in the folder named 'busco_muscle'

You would see the following result after completing the BUSCO run. The result indicates the following, 1,013 BUSCO genes are there in Arthropoda lineage and the transcriptome is 92.5% complete.

```
-----  
|Results from dataset arthropoda_odb10  
-----  
|C:83.8%[S:59.5%,D:24.3%],F:8.7%,M:7.5%,n:1013  
|849 Complete BUSCOs (C)  
|603 Complete and single-copy BUSCOs (S)  
|246 Complete and duplicated BUSCOs (D)  
|88 Fragmented BUSCOs (F)  
|76 Missing BUSCOs (M)  
|1013 Total BUSCO groups searched  
-----  
2022-06-01 20:11:40 INFO: BUSCO analysis done. Total running time: 370 seconds  
2022-06-01 20:11:40 INFO: Results written in /home/vinay/training/busco_muscle
```

### Predict coding regions in the assembled transcripts:

We shall use Transdecoder software to predict coding regions in the assembled transcripts. First, we shall install Transdecoder by using the following commands.

```
conda create -n transdecoder
```

```
conda install -c bioconda transdecoder
```

You may see the total documentation at the github page, <https://github.com/TransDecoder/TransDecoder/wiki>

Run transdecoder on muscle transcripts using the following command,

```
TransDecoder.LongOrfs -t muscle.fasta
```

Count the number of predicted coding transcripts in 'longest\_orfs.cds' file. You may use *grep* command to count.

```
grep -c '>' longest_orfs.cds
```

Observe that 53,040 transcripts were predicted to be having coding potential by the Transdecoder. Now, we shall remove the redundant entries to get a final set of non-redundant transcripts. We shall use CD-HIT software for achieving this. Install cd-hit software using the following command lines,

```
conda<>create<>-n<>cdhit
```

```
conda<>install<>-c<>bioconda<>cd-hit
```

Cluster similar transcripts with the following command line,

```
cd-hit-est<>-i<>longest_orfs.cds.fasta<>-o<>unigenes.fasta<>-c<>0.95<>-B<>1<>-g<>1
```

The command arguments details are,

Argument	Meaning
cd-hit-est	Calls the tool
-i<>longest_orfs.cds.fasta	Input file name is longest_orfs.cds.fasta
--o<>unigenes.fasta	Output file name is unigenes.fasta
-c<>0.95	Sequence identity threshold, default is 0.9
-B<>1	Sequences are stored on hard drive
-g<>1	A sequence is clustered with the most similar cluster rather than the first encountered cluster for the threshold

You would see the following output when program run is successfully completed.

```

----- Output -----
total seq: 53040
longest and shortest : 26466 and 297
Total letters: 45068196
Sequences have been sorted

Approximated minimal memory consumption:
Sequence      : 7M
Buffer        : 1 X 18M = 18M
Table         : 1 X 17M = 17M
Miscellaneous : 4M
Total         : 47M

Table limit with the given memory limit:
Max number of representatives: 537809
Max number of word counting entries: 94013531

comparing sequences from 0 to 53040
..... 10000 finished 5995 clusters
..... 20000 finished 11162 clusters
..... 30000 finished 14881 clusters
..... 40000 finished 18071 clusters
..... 50000 finished 21227 clusters
...
53040 finished 22261 clusters

Approximated maximum memory consumption: 225M
writing new database
writing clustering information
program completed !

```

The tool identified 22,261 unigenes in muscle transcriptome. We shall understand annotating them in a separate lecture.

\*\*\*



Transcriptomic approaches have been very useful in determining gene functions. Transcriptomics also enables the discovery of pathways related to different treatment groups. Transcriptome Analysis is the use of high-throughput technologies to study the transcriptome, or the entire collection of RNA transcripts generated by the genome, under specific conditions or in a specific cell.

The purpose of this hands-on workshop is to complete some basic tasks in RNA-seq data analysis. High quality RNA-seq data (High quality reads) will be aligned to the genome of the Pacific white shrimp (*Penaeus vannamei*) using the STAR aligner. The normalized counts will be generated using RSEM from alignment file, and the counts will be analysed to find differentially expressed genes using edgeR.

Genome-guided transcriptome analysis is applicable if reference genome is available for the species of interest on which RNAseq experiments are conducted. Unlike *de novo* transcriptome assembly, genome-guided/reference based transcriptome analysis requires limited computing facilities. The procedure for deriving differentially expressed genes with tools like STAR and edgeR are explained hereunder.

Input files required for reference based/genome guided RNA-seq analysis

File	Format	Description
Genome	.fasta	Reference genome for indexing and mapping
gff3 annotation	.gtf	Gene transfer format of reference genome
High quality reads	.fq	RNA-Seq reads generated from your experiment

### Software requirements

Software	Version	Purpose
STAR (Spliced Transcripts Alignment to a Reference)	2.7.9a	Genome Indexing and read mapping (Available at: <a href="https://github.com/hbctraining/Intro-to-rnaseq-hpc-O2">https://github.com/hbctraining/Intro-to-rnaseq-hpc-O2</a> )
RSEM (RNA-Seq by Expectation-Maximization)	v1.3.3	Generate count and normalization (Available at: <a href="https://deweylab.github.io/RSEM/">https://deweylab.github.io/RSEM/</a> )
faSomeRecords		To extract sequence from multifasta file (Available at: <a href="https://github.com/santiagosnchez/faSomeRec">https://github.com/santiagosnchez/faSomeRec</a>

			<a href="#">ords</a> )
R		3.38.1	Programming language for statistical computing and graphics (Available at: <a href="https://www.r-project.org/">https://www.r-project.org/</a> )
	edgeR	4.1.2	Differential gene expression (Available at: <a href="https://bioconductor.org/packages/release/bioc/html/edgeR.html">https://bioconductor.org/packages/release/bioc/html/edgeR.html</a> )
	RStudio (optional)	2022.02.0	RStudio is an integrated development environment for R (Available at: <a href="https://www.rstudio.com/">https://www.rstudio.com/</a> )
	limma	4.2	Dependency for edgeR to analyze gene expression (Available at: <a href="https://bioconductor.org/packages/release/bioc/html/limma.html">https://bioconductor.org/packages/release/bioc/html/limma.html</a> )
	ggplot2	3.3.5	Used for plotting (Available at: <a href="https://ggplot2.tidyverse.org/">https://ggplot2.tidyverse.org/</a> )
	gplots	3.1.1	Used for plotting (Available at: <a href="https://www.rdocumentation.org/packages/gplots/versions/3.1.1">https://www.rdocumentation.org/packages/gplots/versions/3.1.1</a> )

## Downloading datasets

Create a ‘rnaseq’ directory, type the following command line:

```
mkdir rnaseq
cd rnaseq
```

For this tutorial, we used two biological replicates (i.e., two sample x two replicates) of RNASeq data (NCBI-SRA bioproject: PRJNA421143) from pacific white shrimp (*Litopenaeus vannamei*) in response to *Vibrio parahaemolyticus* inoculation. Only four sample data sets were chosen from the six samples available for the bioproject (SRX3445056, SRX3445057, SRX3445059 and SRX3445060). Among the data SRX3445056 and SRX3445057 were Healthy (Control) samples and, SRX3445059 and SRX3445060 were *Vibrio parahaemolyticus* infected samples (Treated). After performing a quality check (explained in Chapter 5) on the retrieved data, it was renamed and resized (To save time, ten million reads were taken for the study.). The data available in ‘rnaseqdata/’ directory from ngs user.

To copy the read files from 'rnaseqdata' and paste it in your rnaseq directory

```
cp /home/ngs/rnaseqdata/*.fq /home/user/rnaseq/
```

**user**: User name

The selected fastq read details for this analysis

	Control (Healthy)		Treated (Infected)		
	Forward	Reverse		Forward	Reverse
Replication 1	P_C1_1.fq	P_C1_2.fq	Replication 1	P_T1_1.fq	P_T1_2.fq
Replication 2	P_C2_1.fq	P_C2_2.fq	Replication 2	P_T2_1.fq	P_T2_2.fq

## Reference Genome and Annotation Files

We will use the *Penaeus vannamei* genome assembly version ASM378908v1 available as a multifasta file (scaffold level) from the NCBI. The annotation file is generally saved in a .GFF or .GFF3 (General Feature Format) or a .GTF/GFF2.5 (Gene Transfer Format corresponding to the GFF2.5) file, which is a 9-column tab-delimited file containing information on individual features (gene, transcript, exon, etc.). Download *Penaeus vannamei* genome sequence (.fna) of and annotation (.gtf) file

A: Go to <https://www.ncbi.nlm.nih.gov>, select *Penaeus vannamei* genome

The screenshot shows the NCBI website search interface. The search bar is highlighted with a red box and contains 'Genome' in the dropdown menu and 'Penaeus vannamei' in the text input field. A red box around the search button is also highlighted. Hand icons and numbers 1, 2, and 3 indicate the steps: 1) Click the dropdown menu, 2) Type the search term, and 3) Click the search button.

- 1) Select **Genome** option from selection bar
- 2) Type '*Penaeus vannamei*' search option
- 3) Submit **Search** button

**B:** Click 'Assembly' button in 'Related information' menu

Genome   Create alert Limits Advanced Help

**Penaeus vannamei (Pacific white shrimp)**  
 Representative genome: Penaeus vannamei (assembly ASM378908v1)  
 Download sequences in FASTA format for genome, transcript, protein  
 Download genome annotation in GFF, GenBank or tabular format  
 BLAST against Penaeus vannamei genome, protein  
 All 2 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

Display Settings: Overview Send to: ID: 10710

**Organism Overview** : [Genome Assembly and Annotation report \[2\]](#) : [Organelle Annotation Report \[1\]](#)

**Penaeus vannamei (Pacific white shrimp)**  
 A marine shrimp that is used in shrimp farming

Lineage: Eukaryota[9487]; Metazoa[4496]; Ecdysozoa[2019]; Arthropoda[1861]; Crustacea[74]; Multicrustacea[51]; Malacostraca[34]; Eumalacostraca[34]; Eucarida[22]; Decapoda[22]; Dendrobranchiata[5]; Penaeoidea[5]; Penaeidae[5]; Penaeus[5]; Penaeus vannamei[1]  
*Litopenaeus vannamei*, or pacific white shrimp, is a commercially valuable resource. It can be farm-raised, and can be raised in a low saline

**Related Information**  
 Assembly  Hand cursor  
 BioProject  
 Gene  
 Components  
 Protein  
 PubMed

**C:** Click 'ASM378908v1' button in 'Items' menu

Assembly   Advanced Browse by organism Help

Organism group: Animals (2) Summary - Sort by Significance  
 Status: Latest (2)   
 Latest GenBank (2)  
 Latest RefSeq (1)

Assembly level: Complete genome (0)  
 Chromosome (0)  
 Scaffold (1)  
 Contig (1)

RefSeq category: Reference (0)  
 Representative (1)

Exclude: Exclude partial (0)  
 Exclude from large multi-isolate project (0)  
 Exclude anomalous (0)

**Links from Genome**  
 Items: 2  
 Filters activated: Latest. Exclude anomalous. [Clear all](#)

1.  **ASM378908v1** Hand cursor  
 Organism: [Penaeus vannamei](#) (Pacific white shrimp)  
 Intraspecific name: Breed: Kehai No.1  
 Sex: male  
 Submitter: Institute of Oceanology, Chinese Academy of Sciences  
 Date: 2018/11/16  
 Assembly level: Scaffold  
 Genome representation: full  
 RefSeq category: representative genome

**NCBI Datasets**  
 Download a genome dataset including genome, transcript and protein sequence, annotation and a data report. [Learn more](#)

**Find related data**  
 Database:

**Recent activity**  
 Turn Off Clear  
 Q Assembly for Genome (Select 10710) (2) Assembly

**D:** Click 'FTP directory for RefSeq assembly' from Access the data menu.

Full Report Send to:

**ASM378908v1**  
 Organism name: [Penaeus vannamei](#)  
 Intraspecific name: Breed: Kehai No.1  
 Sex: male  
 BioSample: [SAMN08721527](#)  
 BioProject: [PRJNA438564](#)  
 Submitter: Institute of Oceanology, Chinese Academy of Sciences  
 Date: 2018/11/16  
 Assembly type: na  
 Assembly level: Scaffold  
 Genome representation: full  
 RefSeq category: representative genome  
 GenBank assembly accession: [GCA\\_003789085.1 \(latest\)](#)

See [Genome](#) information for [Penaeus vannamei](#)

Pathogen Detection Resources  
[Isolate Browser](#)  
[SNP Tree Viewer](#)

There are 2 assemblies for this organism  
[See more](#)

**Access the data**  
 Genome Data Viewer  
 RefSeq Annotation Report  
 BLAST the assembly  
 Run Primer-BLAST  
 Full sequence report  
 Statistics report  
 **FTP directory for RefSeq assembly** Hand cursor  
 FTP directory for GenBank assembly  
 NCBI Datasets

**E:** Copy the 'link addresses' from '[GCF\\_003789085.1\\_ASM378908v1\\_genomic.fna.gz](https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003789085.1_ASM378908v1_genomic.fna.gz)' and '[GCF\\_003789085.1\\_ASM378908v1\\_genomic.gtf.gz](https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003789085.1_ASM378908v1_genomic.gtf.gz)'

## Index of /genomes/all/GCF/003/789/085/GCF\_003789085.1\_ASM378908v1

Name	Last modified	Size
Parent Directory		-
Evidence_alignments/	2019-06-24 13:09	-
GCF_003789085.1_ASM378908v1_assembly_structure/	2019-06-24 13:08	-
Gnomon_models/	2019-06-24 13:09	-
RefSeq_transcripts_alignments/	2019-06-24 13:09	-
GCF_003789085.1_ASM378908v1_assembly_report.txt	2021-10-27 12:46	466K
GCF_003789085.1_ASM378908v1_assembly_stats.txt	2021-12-15 21:51	7.2K
GCF_003789085.1_ASM378908v1_cds_from_genomic.fna.gz	2019-06-24 13:08	13M
GCF_003789085.1_ASM378908v1_feature_count.txt.gz	2019-06-24 13:08	448
GCF_003789085.1_ASM378908v1_feature_table.txt.gz		
GCF_003789085.1_ASM378908v1_genomic.fna.gz		
GCF_003789085.1_ASM378908v1_genomic.gbff.gz		
GCF_003789085.1_ASM378908v1_genomic.gff.gz		
GCF_003789085.1_ASM378908v1_genomic.gtf.gz		
GCF_003789085.1_ASM378908v1_genomic.gvgz.txt.gz		
GCF_003789085.1_ASM378908v1_genomic.gvgz.txt.gz		
GCF_003789085.1_ASM378908v1_protein.faa.gz	2019-06-24 13:09	7.0M
GCF_003789085.1_ASM378908v1_protein.gbff.gz	2019-06-24 13:09	17M
GCF_003789085.1_ASM378908v1_pseudo_without_product.fna.gz	2019-06-24 13:09	2.4M
GCF_003789085.1_ASM378908v1_rm.out.gz	2019-06-24 13:09	114M
GCF_003789085.1_ASM378908v1_rm.run	2019-06-24 13:09	900
GCF_003789085.1_ASM378908v1_rna.fna.gz	2019-06-24 13:09	17M
GCF_003789085.1_ASM378908v1_rna.gbff.gz	2019-06-24 13:09	48M
GCF_003789085.1_ASM378908v1_rna_from_genomic.fna.gz	2019-06-24 13:09	19M
GCF_003789085.1_ASM378908v1_translated_cds.faa.gz	2019-06-24 13:09	8.7M

1 Right click in 'GCF\_003789085.1\_ASM378908v1\_genomic.fna.gz' and copy the link to paste in terminal

2 Right click in 'GCF\_003789085.1\_ASM378908v1\_genomic.gtf.gz' and copy the link to paste in terminal

**F:** The following script can be used for downloading reference genome sequence in fasta format (.fna).

Paste the copied 'link addresses' and download the Genome sequence, assembly (ASM378908v1).

```
mkdir genome
cd genome
wget https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/789/085/GCF_003789085.1_ASM378908v1/GCF_003789085.1_ASM378908v1_genomic.fna.gz

#wait, it will take time
```

```
--2022-05-27 14:19:52-- https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/789/085/GCF_003789085.1_ASM378908v1/GCF_003789085.1_ASM378908v1_genomic.fna.gz
Resolving ftp.ncbi.nlm.nih.gov (ftp.ncbi.nlm.nih.gov)... 130.14.250.13, 165.112.9.229, 2607:f220:41e:250::13, ...
Connecting to ftp.ncbi.nlm.nih.gov (ftp.ncbi.nlm.nih.gov)|130.14.250.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 435998212 (416M) [application/x-gzip]
Saving to: 'GCF_003789085.1_ASM378908v1_genomic.fna.gz.2'

GCF_003789085.1_ASM 1%[>] 7.98M 418KB/s eta 19m 5s
```

**G:** Paste the copied 'link addresses' and downloads the annotations

```
wget https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/789/085/GCF_003789085.1_ASM378908v1/GCF_003789085.1_ASM378908v1_genomic.gtf.gz

#wait, it will take time
```

```

gangaraj@mullet:~/rnaseq/genome/recc$ wget https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/789/085/GCF_003789085.1_ASM378908v1/GCF_003789085.1_ASM378908v1_genomic.gtf.gz
--2022-06-04 14:39:19-- https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/789/085/GCF_003789085.1_ASM378908v1/GCF_003789085.1_ASM378908v1_genomic.gtf.gz
Resolving ftp.ncbi.nlm.nih.gov (ftp.ncbi.nlm.nih.gov)... 165.112.9.228, 130.14.250.13, 2607:f220:41e:250::10, ...
Connecting to ftp.ncbi.nlm.nih.gov (ftp.ncbi.nlm.nih.gov)|165.112.9.228|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 8491987 (8.1M) [application/x-gzip]
Saving to: 'GCF_003789085.1_ASM378908v1_genomic.gtf.gz'

GCF_003789085.1_ASM378908v1_genomi 100%[=====>] 8.10M 437KB/s in 21s
2022-06-04 14:39:42 (392 KB/s) - 'GCF_003789085.1_ASM378908v1_genomic.gtf.gz' saved [8491987/8491987]
gangaraj@mullet:~/rnaseq/genome/recc$

```

**H:** Once this job has successfully finished, we should have two files in `genome` directory, with the following files:

```
# list out the downloaded files
ls
```

```

gangaraj@mullet:~/rnaseq2$ ls
GCF_003789085.1_ASM378908v1_genomic.fna.gz  GCF_003789085.1_ASM378908v1_genomic.gtf.gz

```

**I:** STAR needs to unzip the genome files. In order to unzip, we recommend unzipping both the `.fna.gz` and the `.gtf.gz` files using `gunzip` command

```
# unzip the files using gunzip; '*.gz' can be used for complete '.gz' file
gunzip *.gz
```

**J:** The particular annotation (`.gtf`) file containing GTF annotation from NCBI RefSeq contains empty `gene_id` (`gene_id ""`) values. This is disallowed in RSEM, So we need to remove the lines with empty `gene_id` values.

```
grep -v 'gene_id ""' GCF_003789085.1_ASM378908v1_genomic.gtf > GCF_003789085.1_ASM378908v1_genomic.gtf2
```

## Generate Genome index file

Genome indexing would create an organized version of the genome, allowing aligner to map sequences to it efficiently. STAR aligner and RSEM require genome indices for further analysis, so will build combined genome indices.

Before genome indexing we have to create 'rsem\_pv' directory in pwd

```
mkdir rsem_pv
```

To make the genome index, we need to run the following command

```
/home/gangaraj/GtoolZ/RSEM/rsem-prepare-reference -p 10 --star --star-path /home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/ --gtf GCF_003789085.1_ASM378908v1_genomic.gtf2 GCF_003789085.1_ASM378908v1_genomic.fna rsem_pv/pvidx
```

The basic options to generate genome indices using RSEM-STAR are as follows

-p	Number of threads
--star	Aligner using for indexing and mapping
--star-path	Installed path of STAR
--gtf	Annotation file

```
gangaraj@mullet:~/rnaseq2$ /home/gangaraj/GtoolZ/RSEM/rsem-prepare-reference -p 10 --star --star-path /home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/ --gtf GCF_003789085.1_ASM378908v1_genomic.gtf2 GCF_003789085.1_ASM378908v1_genomic.fna rsem_pv/pvidx
rsem-extract-reference-transcripts rsem_pv/pvidx 0 GCF_003789085.1_ASM378908v1_genomic.gtf2 None 0 GCF_003789085.1_ASM378908v1_genomic.fna
Parsed 200000 lines
Parsed 400000 lines
Parsing gtf File is done!
GCF_003789085.1_ASM378908v1_genomic.fna is processed!
38342 transcripts are extracted.
Extracting sequences is done!
Group File is generated!
Transcript Information File is generated!
Chromosome List File is generated!
Extracted Sequences File is generated!

rsem-preref rsem_pv/pvidx.transcripts.fa 1 rsem_pv/pvidx
Refs.makeRefs finished!
Refs.saveRefs finished!
rsem_pv/pvidx.idx.fa is generated!
rsem_pv/pvidx.n2g.idx.fa is generated!

/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64//STAR --runThreadN 10 --runMode genomeGenerate --genomeDir rsem_pv --genomeFastaFiles GCF_003789085.1_ASM378908v1_genomic.fna --sjdbGTFfile GCF_003789085.1_ASM378908v1_genomic.gtf2 --sjdbOverhang 100 --outFileNamePrefix rsem_pv/pvidx
/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64//STAR --runThreadN 10 --runMode genomeGenerate --genomeDir rsem_pv --genomeFastaFiles GCF_003789085.1_ASM378908v1_genomic.fna --sjdbGTFfile GCF_003789085.1_ASM378908v1_genomic.gtf2 --sjdbOverhang 100 --outFileNamePrefix rsem_pv/pvidx
STAR version: 2.7.9a compiled: 2021-05-04T09:43:56-0400 vega:/home/dobin/data/STAR/STARcode/STAR.master/source
Jun 04 17:05:12 ..... started STAR run
Jun 04 17:05:12 ... starting to generate Genome files
Jun 04 17:05:31 ..... processing annotations GTF
Jun 04 17:05:39 ... starting to sort Suffix Array. This may take a long time...
Jun 04 17:05:54 ... sorting Suffix Array chunks and saving them to disk...
Jun 04 17:11:10 ... loading chunks from disk, packing SA...
Jun 04 17:11:51 ... finished generating suffix array
Jun 04 17:11:51 ... generating Suffix Array index
Jun 04 17:14:35 ... completed Suffix Array index
Jun 04 17:14:35 ..... inserting junctions into the genome indices
Jun 04 17:15:53 ... writing Genome to disk ...
Jun 04 17:15:54 ... writing Suffix Array to disk ...
Jun 04 17:16:00 ... writing SIndex to disk
Jun 04 17:16:01 ..... finished successfully
gangaraj@mullet:~/rnaseq2$
```

Once the indexing successfully finished, the ‘rsem\_pv’ directory would show the following files

```

gangaraj@mullet:~/rnaseq/genome$ ls
GCF_003789085.1_ASM378908v1_genomic.fna  GCF_003789085.1_ASM378908v1_genomic.gff_aa  rec
GCF_003789085.1_ASM378908v1_genomic.gff  GCF_003789085.1_ASM378908v1_genomic.gff_ab  rsem_pv
gangaraj@mullet:~/rnaseq/genome$ rs
rsh      rstart      rstartd    rsync      rsyslogd
gangaraj@mullet:~/rnaseq/genome$ cd rsem_pv/
gangaraj@mullet:~/rnaseq/genome/rsem_pv$ ls
chrLength.txt      exonGetrInfo.tab  genomeParameters.txt  pvidx.gtf          pvidx.ti           sjdbInfo.txt
chrNameLength.txt  exonInfo.tab      Log.out               pvidx.idx.fa      pvidx.transcripts.fa  sjdbList.fromGTF.out.tab
chrName.txt        geneInfo.tab      pvidx.chrlist        pvidx.n2g.idx.fa  SA                 sjdbList.out.tab
chrStart.txt       Genome            pvidx.grp            pvidx.seq         SAindex            transcriptInfo.tab
gangaraj@mullet:~/rnaseq/genome/rsem_pv$ cd ..
gangaraj@mullet:~/rnaseq/genome$

```

The preceding command will create several genome indices files in the rsem\_pv directory, majority of them (except the index file 'pvidx') are intermediary files of STAR pipeline. For RSEM analysis, the 'pvidx.' prefixed files are created, of which only one (pvidx.transcripts.fa) is of importance to the user and includes the extracted reference transcripts in Multi-FASTA format. The others are utilized internally by RSEM. pvidx.idx.fa and pvidx.n2g.idx.fa files are not required if ‘—no-bam-output’ argument is given in the command.

Once indexing finished, you can move previous directory (rnaseq) using ‘cd ..’

```
cd ..
```

### Align reads to reference genome

After you have the genome indices generated, you can perform the read alignment (mapping).

**A:** To generate the mapping, run the commands given below for each sample.

Align P\_C1\_1.fq and P\_C1\_2.fq samples with genome (genome indices)

```
/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/STAR --runThreadN 10 --runMode alignReads --genomeDir genome/rsem_pv --outFileNamePrefix P_C1 --readFilesIn P_C1_1.fq P_C1_2.fq --outSAMtype BAM SortedByCoordinate --quantMode TranscriptomeSAM GeneCounts
```

```

gangaraj@mullet:~/rnaseq$ /home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/STAR --runThreadN 10 --runMode alignReads --genomeDir genome/rsem_pv --outFileNamePrefix P_C1 --readFilesIn P_C1_1.fq P_C1_2.fq --outSAMtype BAM SortedByCoordinate --quantMode TranscriptomeSAM
STAR version: 2.7.9a compiled: 2021-05-04T09:43:56-0400 vega:/home/dobin/data/STAR/STARcode/STAR.master/source
Jun 01 17:21:02 ..... started STAR run
Jun 01 17:21:02 ..... loading genome
Jun 01 17:21:16 ..... started mapping
Jun 01 17:25:08 ..... finished mapping
Jun 01 17:25:09 ..... started sorting BAM
Jun 01 17:25:25 ..... finished successfully

```



### Align P\_C2\_1.fq and P\_C2\_2.fq samples with genome (genome indices)

```
/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/STAR --runThreadN 10 --runMode alignReads --genomeDir genome/rsem_pv --outFileNamePrefix P_C2 --readFilesIn P_C2_1.fq P_C2_2.fq --outSAMtype BAM SortedByCoordinate --quantMode Transcriptome SAM GeneCounts
```

### Align P\_T1\_1.fq and P\_T1\_2.fq samples with genome (genome indices)

```
/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/STAR --runThreadN 10 --runMode alignReads --genomeDir genome/rsem_pv --outFileNamePrefix P_T1 --readFilesIn P_T1_1.fq P_T1_2.fq --outSAMtype BAM SortedByCoordinate --quantMode Transcriptome SAM GeneCounts
```

### Align P\_T2\_1.fq and P\_T2\_2.fq samples with genome (genome indices)

```
/home/gangaraj/GtoolZ/STAR-2.7.9a/bin/Linux_x86_64/STAR --runThreadN 10 --runMode alignReads --genomeDir genome/rsem_pv --outFileNamePrefix P_T2 --readFilesIn P_T2_1.fq P_T2_2.fq --outSAMtype BAM SortedByCoordinate --quantMode Transcriptome SAM GeneCounts
```

The basic options to map reads to genome indices using STAR are as follows

--runThreadN	Number of threads
runMode	'alignReads' star option for aligning reads
--genomeDir	Genome indices directory
--outFileNamePrefix	Output files prefix
readFilesIn	Input reads
--outSAMtype	Output type; BAM or SAM
--quantMode	used to produce a transcriptome bam file that will be used by RSEM
GeneCounts	Counting number of reads per gene

**B:** Once the mapping successfully finished, the 'rnaseq' directory would contained the following files

```

P_C2Log.progress.out      P_C2Log.out              P_T1Log.progress.out
P_C2Log.final.out        P_C2Log.progress.out     P_T1Log.out             
P_C1_1.fq                 P_C2ReadsPerGene.out.tab P_T1Log.final.out      
P_C1_2.fq                 P_C2SJ.out.tab           P_T1SJ.out.tab          
P_C1Aligned.sortedByCoord.out.bam P_C2Log.progress.out     P_T2_1.fq                
P_C1Aligned.toTranscriptome.out.bam P_C2ReadsPerGene.out.tab P_T2_2.fq                
P_C1Log.final.out         P_C2SJ.out.tab           P_T2Aligned.sortedByCoord.out.bam
P_C1Log.out               P_T1_1.fq                P_T2Aligned.toTranscriptome.out.bam
P_C1Log.progress.out      P_T1_2.fq                P_T2Log.final.out      
P_C1ReadsPerGene.out.tab  P_T1Aligned.sortedByCoord.out.bam P_T2Log.out             
P_C1SJ.out.tab            P_T1Aligned.toTranscriptome.out.bam P_T2Log.progress.out   
P_C2_1.fq                 P_T1Log.final.out        P_T2ReadsPerGene.out.tab
P_C2_2.fq                 P_T1Log.out              P_T2SJ.out.tab          
P_C2Aligned.sortedByCoord.out.bam P_T1Log.progress.out
gangaraj@mullet:~/rnaseq2$ ls
genome P_C2Aligned.toTranscriptome.out.bam P_T1ReadsPerGene.out.tab
P_C1_1.fq P_C2Log.final.out P_T1SJ.out.tab
P_C1_2.fq P_C2Log.out P_T2_1.fq
P_C1Aligned.sortedByCoord.out.bam P_C2Log.progress.out P_T2_2.fq
P_C1Aligned.toTranscriptome.out.bam P_C2ReadsPerGene.out.tab P_T2Aligned.sortedByCoord.out.bam
P_C1Log.final.out P_C2SJ.out.tab P_T2Aligned.toTranscriptome.out.bam
P_C1Log.out P_T1_1.fq P_T2Log.final.out
P_C1Log.progress.out P_T1_2.fq P_T2Log.out
P_C1ReadsPerGene.out.tab P_T1Aligned.sortedByCoord.out.bam P_T2Log.progress.out
P_C1SJ.out.tab P_T1Aligned.toTranscriptome.out.bam P_T2ReadsPerGene.out.tab
P_C2_1.fq P_T1Log.final.out P_T2SJ.out.tab
P_C2_2.fq P_T1Log.out
P_C2Aligned.sortedByCoord.out.bam P_T1Log.progress.out
gangaraj@mullet:~/rnaseq2$

```

STAR produces many output files within the current working directory, the most important of which are the following

**Log.final.out:** summary mapping statistics, the number and percentage of fragments that are mapped uniquely, those are mapped several times, and unmapped. To view alignment statistics in 'Log.final.out'. You can 'double click' on each 'Log.final.out' file or you can use 'nano' (eg: nano P\_C1Log.final.out ) command to view in terminal

**Aligned.sortedByCoord.out.bam:** the genome Binary Alignment Map (BAM) file sorted by coordinates.

**Aligned.toTranscriptome.out.bam:** the transcriptome BAM file. This file is using for downstream RSEM quantifications.

**SJ.out.tab:** splice junction details.

**ReadsPerGene.out.tab:** Counts number of reads per gene

## Transcript and Gene Quantifications

RSEM uses mapped reads to quantify the expression of transcripts and genes.

**A:** Preparing the RSEM reference files

The initial step of RSEM quantification is to produce genome indices. Previously we created combined genome indices for STAR and RSEM (~/genome/rsem\_pv/).

**B:** Running the Quantification Process

To calculate expression values, you should run the rsem-calculate-expression program

To get expression values of C1 run the command

```
/home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10
--paired-end P_C1Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx C1
```

```
gangaraj@mullet:~/rnaseq$ /home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10 --paired-end P_C1Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx C1
rsem-parse-alignments genome/rsem_pv/pvidx PC1.temp/PC1 PC1.stat/PC1 P_C1Aligned.toTranscriptome.out.bam 3 -tag XM
^Z
[1]+  Stopped                  /home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10 --paired-end P_C1Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx C1
gangaraj@mullet:~/rnaseq$ /home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 100 --paired-end P_C1Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx C1
rsem-parse-alignments genome/rsem_pv/pvidx PC1.temp/PC1 PC1.stat/PC1 P_C1Aligned.toTranscriptome.out.bam 3 -tag XM
Parsed 1000000 entries
Parsed 2000000 entries
Parsed 3000000 entries
Parsed 4000000 entries
Parsed 5000000 entries
Parsed 6000000 entries
Parsed 7000000 entries
Parsed 8000000 entries
ROUND = 4862, SUM = 6261140.00000001, bChange = 0.00129831, totNum = 2
ROUND = 4863, SUM = 6261140.00000001, bChange = 0.00129841, totNum = 2
ROUND = 4864, SUM = 6261140.00000001, bChange = 0.00119965, totNum = 1
ROUND = 4865, SUM = 6261140.00000001, bChange = 0.000893025, totNum = 0
Expression Results are written!
Time Used for EM.cpp : 0 h 01 m 14 s
rm -rf PC1.temp
gangaraj@mullet:~/rnaseq$
```

To get expression values of C2 run the command

```
/home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10
--paired-end P_C2Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx C2
```

To get expression values of T1 run the command

```
/home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10
--paired-end P_T1Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx T1
```

To get expression values of T2 run the command

```
/home/gangaraj/GtoolZ/RSEM/rsem-calculate-expression --bam --no-bam-output -p 10
--paired-end P_T2Aligned.toTranscriptome.out.bam genome/rsem_pv/pvidx T2
```

The basic options of rsem-calculate-expression are as follows

--bam	Alignment file
--no-bam-output	No need to generate bam file
-p	Number of threads
-- paired-end	Data information
--star/--hisat/ --bowtie2	Aligner information

rsem-calculate-expression command would generate two file and.

**sample\_name.isoforms.results** : File containing gene level expression estimates.

**sample\_name.genes.results**: File containing isoform level expression estimates.

The description for each column of sample\_name.isoforms.results :

transcript_id	The transcript name
gene_id	The gene name of the gene which this transcript belongs to (denote this gene as its parent gene).If no gene information is provided, 'gene_id' and 'transcript_id' are the same.
length	Transcript's sequence length
effective_length	counts only the positions that can generate a valid fragment. If no poly(A) tail is added, 'effective_length' is equal to (transcript length - mean fragment length) + 1. If one transcript's effective length is less than 1, this transcript's both effective length and abundance estimates are set to 0.
expected_count	The sum of the posterior probability of each read comes from this transcript over all reads
TPM (Transcripts Per Million)	Normalised count;It is a relative measure of transcript abundance
FPKM (Fragments Per Kilobase Million)	Normalised count;It is another relative measure of transcript abundance
IsoPct (Isoform Percentage)	It is the percentage of this transcript's abundance over its parent gene's abundance

**C:** Generate expression matrix using expression data.

For expression analysis we need to extract FPKM values from each result (isoforms.results).

```
cut -f 1,7 C1.isoforms.results | sed '1s/FPKM/C1/g' > C1.txt
```

```
cut -f 1,7 C2.isoforms.results | sed '1s/FPKM/C2/g' > C2.txt
```

```
cut -f 1,7 T1.isoforms.results | sed '1s/FPKM/T1/g' > T1.txt
```

```
cut -f 1,7 T2.isoforms.results | sed '1s/FPKM/T2/g' > T2.txt
```

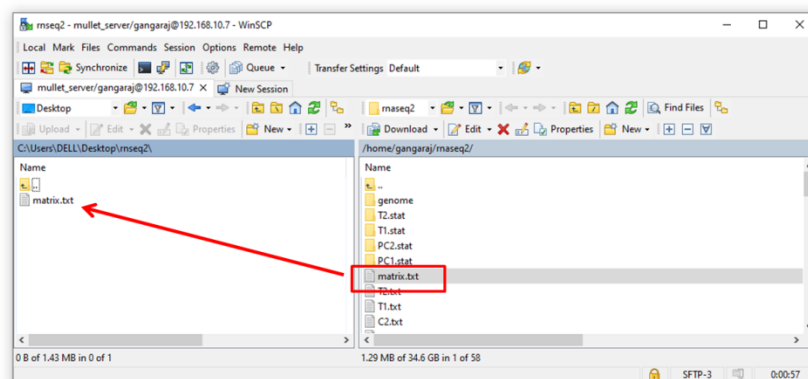
The FPKM values of each sample can be combined and can be stored in single 'text' file

```
paste T1.txt T2.txt C1.txt C2.txt | cut -f 1,2,4,6,8 > matrix.txt
```

## Differential Gene Expression Analysis

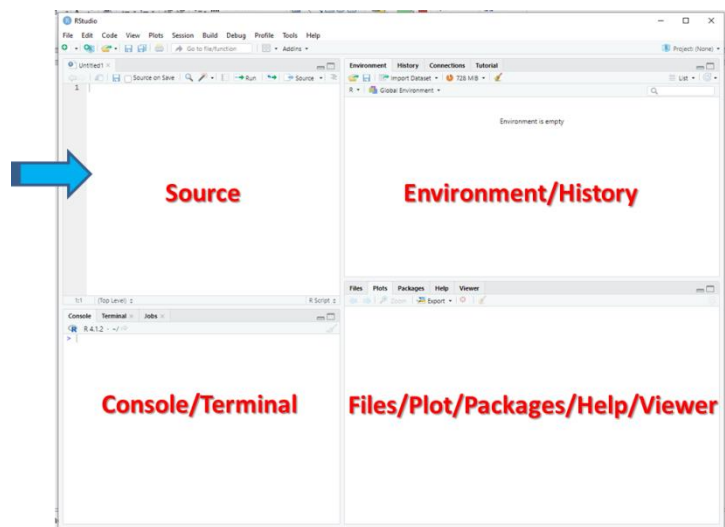
Differential expression analysis approach using the normalised read count data and performing statistical analysis to discover quantitative changes in expression levels between experimental groups. It is important to consider the experimental design when choosing an analysis method. We have to provide correct treated and control groups and their replications. Here we are using edgeR program for differential gene expression analysis.

**A:** Create **rnaseq** folder in desktop and copy '**matrix.txt**' file to that folder



**B:** Open matrix.txt file in Microsoft excel, change headers if you want and save as in matrix.csv

## C: Open R studio run edgeR command for DGE analysis



Snapshot of R studio

a: Install ggplot2, limma, edgeR and gplots; copy and paste these commands in R Studio 'source panel', select each command and click 'Ctrl+Enter'

```
#install ggplot2
install.packages("ggplot2")

#####
#install edgeR
if (!require("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("edgeR")

#####
#install limma
if (!require("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("limma")

#####
#install gplots
install.packages('gplots')
```

b: Load all Libraries

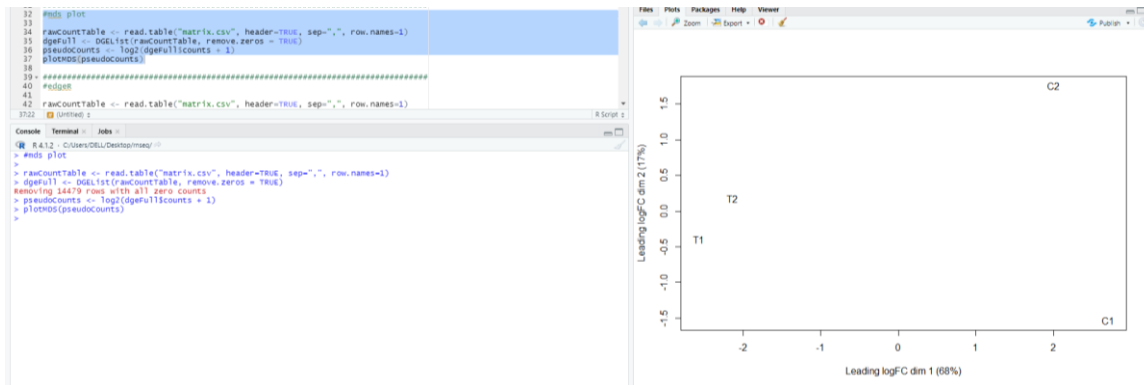
```
library(edgeR)
library(limma)
library(ggplot2)
library(gplots)
```

**c: Set working directory**

```
setwd("C:/Users/DELL/Desktop/rnseq")
getwd()
```

**d: Run MDS plot command**

```
rawCountTable <- read.table("matrix.csv", header=TRUE, sep="," , row.names=1)
dgeFull <- DGEList(rawCountTable, remove.zeros = TRUE)
pseudoCounts <- log2(dgeFull$counts + 1)
plotMDS(pseudoCounts)
```

**e: Run edgeR command**

```
rawCountTable <- read.table("matrix.csv", header=TRUE, sep="," , row.names=1)
group<-factor(c(1,1,2,2))
y <- DGEList(counts=rawCountTable,group=group,remove.zeros = TRUE)
y <- calcNormFactors(y)
et <- exactTest(y,dispersion=0.3)
et <- topTags(et, n = nrow(et$table))
sd1<- subset(et$table, FDR<0.05)
sups<-subset(sd1,logFC>2)
sdowns<-subset(sd1,logFC< (-2))
write.csv(sups,'up.csv')
write.csv(sdowns,'dn.csv')
```

Following the execution of the edgeR command, two.csv files (up.csv and dn.csv) will be created, each containing differential gene expression information. Those transcripts with a change of >2 fold change are considered up regulated, whereas those with a change of <-2 are considered down regulated. Open each file (up.csv and dn.csv) and check the differential gene expression statistics

f: Type **head(sups)** in the source panel to see the first part of up regulated transcripts, and **head(sdowns)** in the source panel to see the first part of down regulated transcripts.

head(sups)

```
> head(sups)
      logFC      logCPM      Pvalue      FDR
XM_027356017.1 10.015182 10.189611 4.899771e-16 1.948721e-12
XM_027357893.1  8.988249  6.508596 3.798011e-11 1.777097e-08
XM_027356137.1  9.098800  6.945322 4.009431e-11 1.839943e-08
XM_027354853.1 10.458735  6.442581 5.260340e-11 2.324583e-08
XM_027383403.1  8.667700  6.314460 9.215893e-11 3.927122e-08
XM_027371371.1 10.223868  6.210587 1.523416e-10 6.377767e-08
> |
```

head(sdowns)

```
> head(sdowns)
      logFC      logCPM      Pvalue      FDR
XM_027360710.1 -10.952277 11.207021 1.048144e-17 2.061888e-13
XR_003477395.1 -13.721569  9.691388 1.728105e-17 2.061888e-13
XM_027360709.1 -11.579104  9.850008 6.393281e-17 4.410276e-13
XM_027376154.1  -9.995500 12.050474 7.392660e-17 4.410276e-13
XM_027360707.1 -13.120660  9.094032 2.747152e-16 1.311106e-12
XM_027360712.1  -9.842495 10.747268 6.607557e-16 2.252516e-12
> |
```

The edgeR differential gene expression statistics

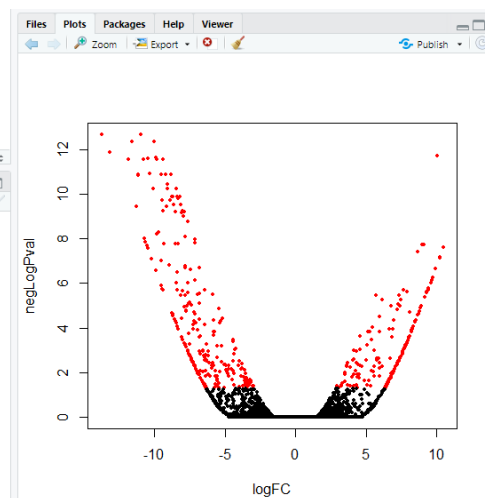
logFC	Log fold change
logCPM	log counts per million
Pvalue	Probability
FDR	False discovery rate ( $\leq 0.05$ )

g: Run **volcano plot** command

```
volcanoData <- cbind(et$table$logFC, -log10(et$table$FDR))
colnames(volcanoData) <- c("logFC", "negLogPval")
DEGs <- et$table$FDR < 0.05 & abs(et$table$logFC) > 2
point.col <- ifelse(DEGs, "red", "black")
plot(volcanoData, pch = 16, col = point.col, cex = 0.5)
```

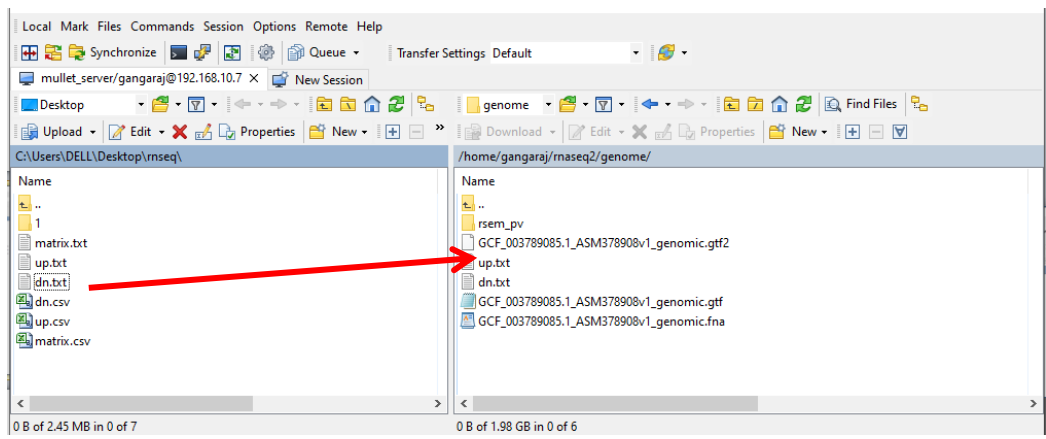
```
40 ##volcano plot
41 volcanoData <- cbind(et$table$logFC, -log10(et$table$FDR))
42 colnames(volcanoData) <- c("logFC", "negLogPval")
43 DEGs <- et$table$FDR < 0.05 & abs(et$table$logFC) > 2
44 point.col <- ifelse(DEGs, "red", "black")
45 plot(volcanoData, pch = 16, col = point.col, cex = 0.5)
46
47 #####
48
49
50
51
52
53
54
55
56
57
58
59
60
```

```
R 4.1.2 - C:/Users/DELL/Desktop/mseq/
> ##volcano plot
> volcanoData <- cbind(et$table$logFC, -log10(et$table$FDR))
> colnames(volcanoData) <- c("logFC", "negLogPval")
> DEGs <- et$table$FDR < 0.05 & abs(et$table$logFC) > 2
> point.col <- ifelse(DEGs, "red", "black")
> plot(volcanoData, pch = 16, col = point.col, cex = 0.5)
>
```



**D:Extract fasta sequences of differentially expressed genes**

- a: Copy DGE list (transcript id) into text file (up.txt and dn.txt)
- b: Copy the files to ‘genome’ directory in server



- c: Change your rnaseq directory to genome directory in server

```
cd genome
```

- d: Run faSomeRecords to extract differently expressed transcripts from transcriptome

To extract up regulated transcripts run the command

```
faSomeRecords --fasta rsem_pv/pvidx.transcripts.fa --list up.txt --outfile up.fasta
```

```
gangaraj@mullet:~/rnaseq2/genome$ faSomeRecords --fasta rsem_pv/pvidx.transcripts.fa --list up.txt --outfile up.fasta
Found 191 sequence(s)
Sequences saved to: up.fasta
gangaraj@mullet:~/rnaseq2/genome$
```

To extract down regulated transcripts run the command

```
faSomeRecords --fasta rsem_pv/pvidx.transcripts.fa --list dn.txt --outfile dn.fasta
```

```
gangaraj@mullet:~/rnaseq2/genome$ faSomeRecords --fasta rsem_pv/pvidx.transcripts.fa --list dn.txt --outfile dn.fasta
Found 264 sequence(s)
Sequences saved to: dn.fasta
gangaraj@mullet:~/rnaseq2/genome$
```

The basic options of faSomeRecords are as follows

--fasta	Genome or transcriptome sequence file (.fasta)
--list	List of transcript id
--outfile	Output sequence file (.fasta)
--exclude	To exclude the sequence

\*\*\*



Annotation refers to deriving the functional information of the genes based on sequence homology against known databases. In this chapter we will annotate the transcript sequences based on homology against NCBI's non-redundant protein database (nrdb), EggNOG database and Interpro database. and the annotations will be combined to obtain the gene ontology annotation.

### **Homology based annotation using Blastx against nrdb (*non-redundant database*)**

The blastx performs a local alignment of the query sequences (in our case the transcript file) against the protein database (in our case nrdb).

Files Required:

- a) query file in fasta format. (Transcript file or Differentially expressed genes file )
- b) Taxonomy ID list. ( the list of taxonomy ID's of selected organisms against which the blast search needs to be performed, in our case Arthropoda)
- c) Database (the curated database files against which blast need to be performed)

Note:- Taxonomy ID list is optional, can be used to reduce the analysis time if query is from a known species, for meta transcriptome analysis this option should not be used.

Software Required:

- a) NCBI's Standalone blast tool (BLAST+ 2.13.0)

### **Downloading nrdb**

1. Visit NCBI's ftp site (<https://ftp.ncbi.nlm.nih.gov/blast/db/>) and Download the nrdb files to local system or server.

Name	Last modified	Size
Parent_Directory		-
FASTA/	2022-05-30 18:41	-
cloud/	2020-02-11 16:27	-
v4/	2020-06-30 10:29	-
v5/	2022-06-02 00:33	-
16S_ribosomal_RNA-nucl-metadata.json	2022-05-26 05:36	467
16S_ribosomal_RNA.tar.gz	2022-05-26 05:36	37H
16S_ribosomal_RNA.tar.gz.md5	2022-05-26 05:36	59
16S_fungal_sequences-nucl-metadata.json	2022-05-26 05:36	489
16S_fungal_sequences.tar.gz	2022-05-26 05:36	32H
16S_fungal_sequences.tar.gz.md5	2022-05-26 05:36	62
28S_fungal_sequences-nucl-metadata.json	2022-05-26 05:37	489
28S_fungal_sequences.tar.gz	2022-05-26 05:37	33H
28S_fungal_sequences.tar.gz.md5	2022-05-26 05:37	62
Betacoronavirus-nucl-metadata.json	2022-06-02 00:33	925
Betacoronavirus_00.tar.gz	2022-06-02 00:33	664H
Betacoronavirus_00.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_01.tar.gz	2022-06-02 00:33	413H
Betacoronavirus_01.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_02.tar.gz	2022-06-02 00:33	422H
Betacoronavirus_02.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_03.tar.gz	2022-06-02 00:33	360H
Betacoronavirus_03.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_04.tar.gz	2022-06-02 00:33	468H
Betacoronavirus_04.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_05.tar.gz	2022-06-02 00:33	494H
Betacoronavirus_05.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_06.tar.gz	2022-06-02 00:33	547H
Betacoronavirus_06.tar.gz.md5	2022-06-02 00:33	60
Betacoronavirus_07.tar.gz	2022-06-02 00:33	572H
Betacoronavirus_07.tar.gz.md5	2022-06-02 00:33	60
IIS_RefSeq_Fungi-nucl-metadata.json	2022-05-27 05:38	489
IIS_RefSeq_Fungi.tar.gz	2022-05-27 05:38	34H
IIS_RefSeq_Fungi.tar.gz.md5	2022-05-27 05:38	58
IIS_eukaryote_sequences-nucl-metadata.json	2022-05-27 05:38	446
IIS_eukaryote_sequences.tar.gz	2022-05-27 05:38	44H
IIS_eukaryote_sequences.tar.gz.md5	2022-05-27 05:38	65
LSU_eukaryote_rRNA-nucl-metadata.json	2021-07-21 05:37	424
LSU_eukaryote_rRNA.tar.gz	2021-07-21 05:37	33H
LSU_eukaryote_rRNA.tar.gz.md5	2021-07-21 05:37	60
LSU_eukaryote_rRNA-nucl-metadata.json	2021-07-21 05:37	426
LSU_eukaryote_rRNA.tar.gz	2021-07-21 05:37	33H
LSU_eukaryote_rRNA.tar.gz.md5	2021-07-21 05:37	61
README	2020-09-29 17:03	7.8K
mouse_genome_01.tar.gz.md5	2021-05-28 10:08	57
nr-prnt-metadata.json	2022-05-30 16:36	3.7K
nr_00.tar.gz	2022-05-30 16:22	22G
nr_00.tar.gz.md5	2022-05-30 16:22	47
nr_01.tar.gz	2022-05-30 16:22	2.6G
nr_01.tar.gz.md5	2022-05-30 16:22	47
nr_02.tar.gz	2022-05-30 16:23	1.9G
nr_02.tar.gz.md5	2022-05-30 16:23	47
nr_03.tar.gz	2022-05-30 16:23	2.3G
nr_03.tar.gz.md5	2022-05-30 16:23	47
nr_04.tar.gz	2022-05-30 16:23	2.4G
nr_04.tar.gz.md5	2022-05-30 16:23	47
nr_05.tar.gz	2022-05-30 16:23	2.6G
nr_05.tar.gz.md5	2022-05-30 16:23	47
nr_06.tar.gz	2022-05-30 16:24	2.7G
nr_06.tar.gz.md5	2022-05-30 16:24	47
nr_07.tar.gz	2022-05-30 16:24	2.7G
nr_07.tar.gz.md5	2022-05-30 16:24	47
nr_08.tar.gz	2022-05-30 16:24	2.3G
nr_08.tar.gz.md5	2022-05-30 16:24	47
nr_09.tar.gz	2022-05-30 16:24	2.7G
nr_09.tar.gz.md5	2022-05-30 16:24	47
nr_10.tar.gz	2022-05-30 16:25	2.6G
nr_10.tar.gz.md5	2022-05-30 16:25	47
nr_11.tar.gz	2022-05-30 16:25	2.8G
nr_11.tar.gz.md5	2022-05-30 16:25	47
nr_12.tar.gz	2022-05-30 16:25	2.6G
nr_12.tar.gz.md5	2022-05-30 16:25	47
nr_13.tar.gz	2022-05-30 16:25	3.0G
nr_13.tar.gz.md5	2022-05-30 16:25	47
nr_14.tar.gz	2022-05-30 16:26	252M
nr_14.tar.gz.md5	2022-05-30 16:26	47
nr_15.tar.gz	2022-05-30 16:26	2.2G
nr_15.tar.gz.md5	2022-05-30 16:26	47
nr_16.tar.gz	2022-05-30 16:26	2.6G
nr_16.tar.gz.md5	2022-05-30 16:26	47
nr_17.tar.gz	2022-05-30 16:26	2.8G
nr_17.tar.gz.md5	2022-05-30 16:26	47
nr_18.tar.gz	2022-05-30 16:27	2.7G
nr_18.tar.gz.md5	2022-05-30 16:27	47
nr_19.tar.gz	2022-05-30 16:27	2.8G
nr_19.tar.gz.md5	2022-05-30 16:27	47
nr_20.tar.gz	2022-05-30 16:27	2.5G
nr_20.tar.gz.md5	2022-05-30 16:27	47
nr_21.tar.gz	2022-05-30 16:27	2.5G
nr_21.tar.gz.md5	2022-05-30 16:27	47
nr_22.tar.gz	2022-05-30 16:28	2.3G
nr_22.tar.gz.md5	2022-05-30 16:28	47
nr_23.tar.gz	2022-05-30 16:28	3.0G
nr_23.tar.gz.md5	2022-05-30 16:28	47
nr_24.tar.gz	2022-05-30 16:28	72H

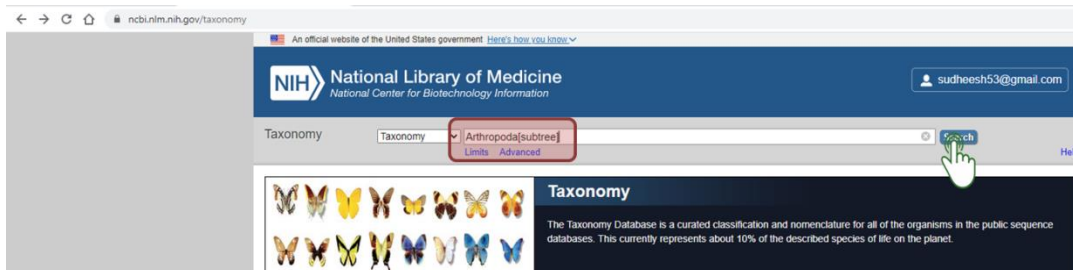
2. Extracting database from the tar.gz files

```
tar<>-xvzf<>*.tar.gz
```

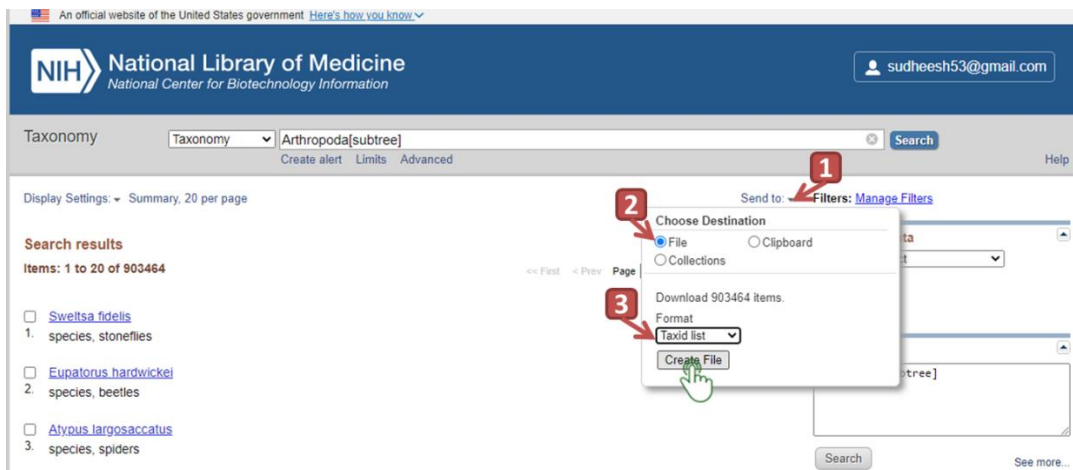
The above command will extract all the compressed nrdb files in to the local folder.

Getting Taxonomic IDs

1. Visit NCBI's taxonomy page (<https://www.ncbi.nlm.nih.gov/taxonomy>) and enter the search term (in our case 'Arthropoda[subtree]') and click the search button to display all the organisms in the phylum arthropoda.



2. The above function displays the organisms in the phylum Arthropoda, click on Send to, select file and change the format to Taxid list and hit the Create file button to save the taxidlist file in the local system.



### Downloading and installing Blast Stand alone

1. Download the appropriate installation module of standalone blast from the NCBI's ftp site (<https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>)

← → ↻ 🏠 <https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>

## Index of /blast/executables/blast+/LATEST

Name	Last modified	Size
<a href="#">Parent Directory</a>		-
<a href="#">ChangeLog</a>	2022-03-14 14:51	85
<a href="#">ncbi-blast-2.13.0+-1.src.rpm</a>	2022-03-14 14:51	53M
<a href="#">ncbi-blast-2.13.0+-1.src.rpm.md5</a>	2022-03-14 14:52	63
<a href="#">ncbi-blast-2.13.0+-1.x86_64.rpm.md5</a>	2022-03-14 14:52	66
<a href="#">ncbi-blast-2.13.0+-src.tar.gz</a>	2022-03-14 14:52	57M
<a href="#">ncbi-blast-2.13.0+-src.tar.gz.md5</a>	2022-03-14 14:52	64
<a href="#">ncbi-blast-2.13.0+-src.zip</a>	2022-03-14 14:52	61M
<a href="#">ncbi-blast-2.13.0+-src.zip.md5</a>	2022-03-14 14:52	61
<a href="#">ncbi-blast-2.13.0+-win64.exe</a>	2022-03-14 14:52	112M
<a href="#">ncbi-blast-2.13.0+-win64.exe.md5</a>	2022-03-14 14:52	63
<a href="#">ncbi-blast-2.13.0+-x64-arm-linux.tar.gz</a>	2022-03-14 14:52	222M
<a href="#">ncbi-blast-2.13.0+-x64-arm-linux.tar.gz.md5</a>	2022-03-14 14:52	74
<a href="#">ncbi-blast-2.13.0+-x64-linux.tar.gz</a>	2022-03-14 14:52	223M
<a href="#">ncbi-blast-2.13.0+-x64-linux.tar.gz.md5</a>	2022-03-14 14:52	70
<a href="#">ncbi-blast-2.13.0+-x64-macosx.tar.gz</a>	2022-03-14 14:52	182M
<a href="#">ncbi-blast-2.13.0+-x64-macosx.tar.gz.md5</a>	2022-03-14 14:52	71
<a href="#">ncbi-blast-2.13.0+-x64-win64.tar.gz</a>	2022-03-14 14:52	115M
<a href="#">ncbi-blast-2.13.0+-x64-win64.tar.gz.md5</a>	2022-03-14 14:52	70
<a href="#">ncbi-blast-2.13.0+.dmg</a>	2022-03-14 14:52	183M
<a href="#">ncbi-blast-2.13.0+.dmg.md5</a>	2022-03-14 14:52	57
<a href="#">ncbi-blast-2.13.0-1.x86_64.rpm</a>	2022-03-14 14:52	180M

[HHS Vulnerability Disclosure](#)

2. Alternately you can download using command line by executing the following command.

```
wget<>https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.13.0+-x64-linux.tar.gz
```

```

(base) sudheesh@192:~$ wget https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.13.0+-x64-linux.tar.gz
--2022-06-03 15:37:42-- https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.13.0+-x64-linux.tar.gz
Resolving ftp.ncbi.nlm.nih.gov... 130.14.250.10, 130.14.250.7, 2607:f220:41e:250::11, ...
Connecting to ftp.ncbi.nlm.nih.gov|130.14.250.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 233822517 (223M) [application/x-gzip]
Saving to: 'ncbi-blast-2.13.0+-x64-linux.tar.gz.2'

ncbi-blast-2.13.0+-x64-linux.ta  3%[>] 8.21M  432KB/s  eta 10m 34s

```

3. Once download is completed install using the following command;

```
tar<>zxvpf<>ncbi-blast-2.10.1+-x64-linux.tar.gz
```

```

(base) sudheesh@192:~$ tar zxvpf ncbi-blast-2.13.0+-x64-linux.tar.gz
./ncbi-blast-2.13.0+/
./ncbi-blast-2.13.0+/ncbi_package_info
./ncbi-blast-2.13.0+/doc/
./ncbi-blast-2.13.0+/doc/README.txt
./ncbi-blast-2.13.0+/doc/BLAST_PRIVACY
./ncbi-blast-2.13.0+/ChangeLog
./ncbi-blast-2.13.0+/README
./ncbi-blast-2.13.0+/LICENSE
./ncbi-blast-2.13.0+/bin/
./ncbi-blast-2.13.0+/bin/blastn
./ncbi-blast-2.13.0+/bin/dustmasker
./ncbi-blast-2.13.0+/bin/blast_vdb_cmd
./ncbi-blast-2.13.0+/bin/windowmasker
./ncbi-blast-2.13.0+/bin/blastn_vdb
./ncbi-blast-2.13.0+/bin/makeblastdb
./ncbi-blast-2.13.0+/bin/makeprofiledb
./ncbi-blast-2.13.0+/bin/blastp
./ncbi-blast-2.13.0+/bin/psiblast
./ncbi-blast-2.13.0+/bin/blastx
./ncbi-blast-2.13.0+/bin/deltablast
./ncbi-blast-2.13.0+/bin/cleanup_blastdb-volumes.py
./ncbi-blast-2.13.0+/bin/get_species_taxids.sh
./ncbi-blast-2.13.0+/bin/blastdbcmd
./ncbi-blast-2.13.0+/bin/legacy_blast.pl
./ncbi-blast-2.13.0+/bin/blastdbcheck
./ncbi-blast-2.13.0+/bin/tblastn
./ncbi-blast-2.13.0+/bin/tblastn_vdb
./ncbi-blast-2.13.0+/bin/blastdb_aliastool
./ncbi-blast-2.13.0+/bin/convert2blastmask
./ncbi-blast-2.13.0+/bin/blast_formatter_vdb
./ncbi-blast-2.13.0+/bin/blast_formatter
./ncbi-blast-2.13.0+/bin/rpstblastn
./ncbi-blast-2.13.0+/bin/rpsblast
./ncbi-blast-2.13.0+/bin/tblastx
./ncbi-blast-2.13.0+/bin/segmasker
./ncbi-blast-2.13.0+/bin/update_blastdb.pl
./ncbi-blast-2.13.0+/bin/makemindex
(base) sudheesh@192:~$

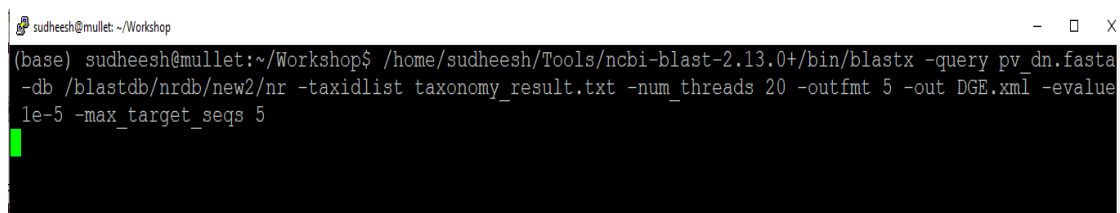
```

4. Executing the above command will create a folder name ncbi-blast-2.13.0+ and all the blast executables will be present in the bin folder inside the ncbi-blast-2.13.0+ folder.

## Running Blastx

- a) Copy the query input file and taxonomy ID list into the same folder from which you wish to execute the blastx command. Execute the following command to run blastx.

```
<path/to/blastx/executable/>blastx<>-query<>pv_dn.fasta<>-db<>/blastdb/nrdb/new2/nr<>-taxidlist<>taxonomy_result.txt<>-num_threads<>20<>-outfmt<>5<>-out<>DGE.xml<>-evalue<>1e-5<>-max_target_seqs<>5
```



```
sudheesh@mullet: ~/Workshop
(base) sudheesh@mullet:~/Workshop$ /home/sudheesh/Tools/ncbi-blast-2.13.0+/bin/blastx -query pv_dn.fasta -db /blastdb/nrdb/new2/nr -taxidlist taxonomy_result.txt -num_threads 20 -outfmt 5 -out DGE.xml -evalue 1e-5 -max_target_seqs 5
```

### Options :

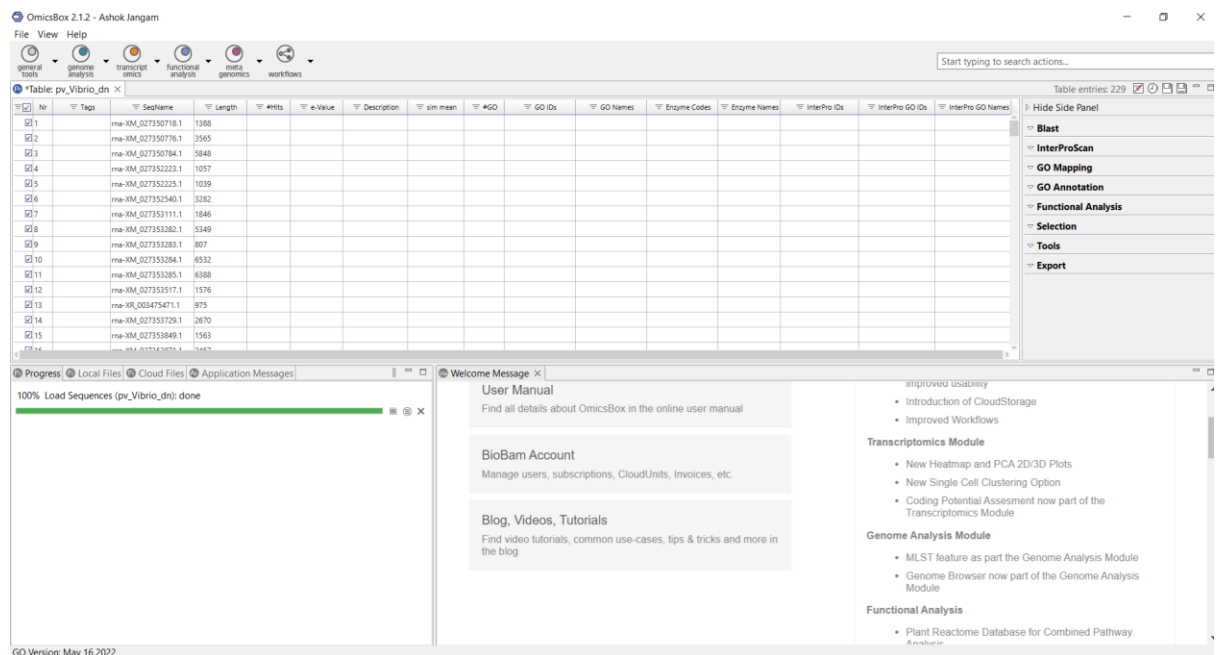
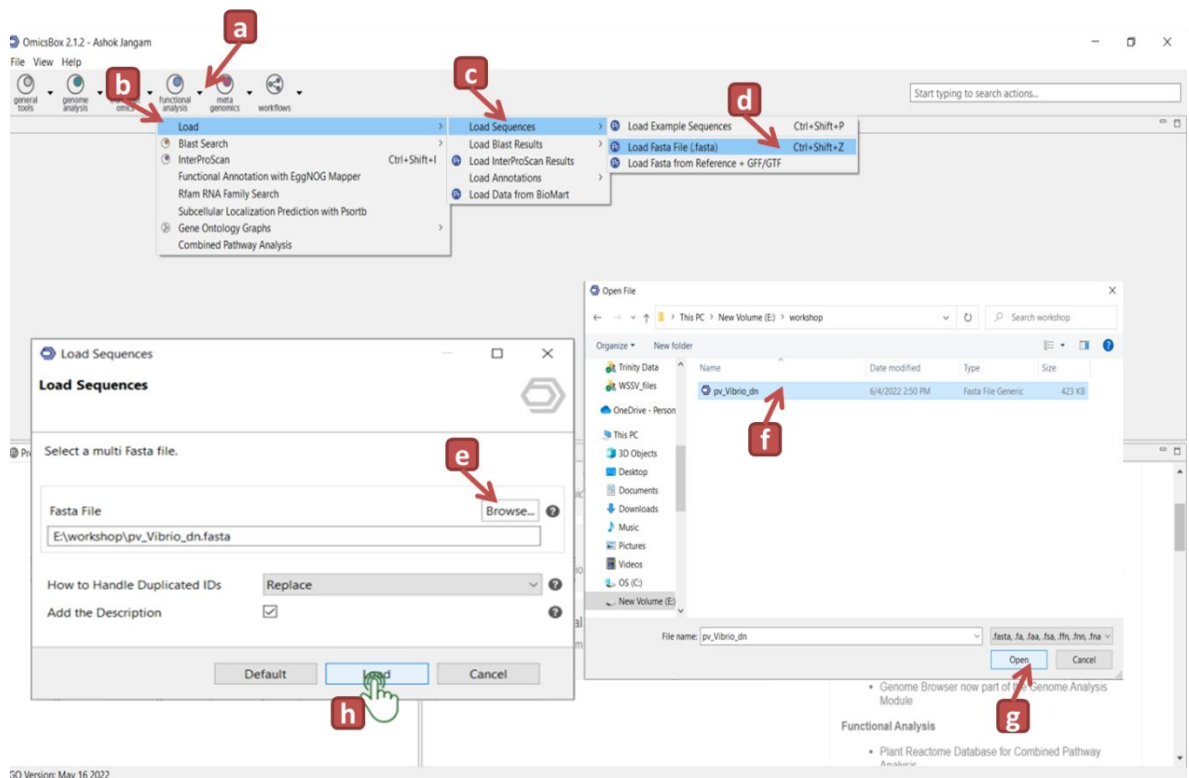
-query	query file
-taxidlist	list of taxonomy ID's
-num_threads	No. of computational threads to be used
-outfmt	blast output format (5 for xml format)
-out	output file name
-evalue	minimum Expect value for the alignment
-max_target_seqs	maximum target sequences per query sequence.

- b) The output will be generated in xml format and can be loaded into *OmicsBox* tool for further analysis.

## Loading the sequences to OmicsBox

Open the OmicsBox tool and load the transcript sequences by following the steps mentioned below;

- a) Click on the dropdown menu under the Functional analysis module
- b) Select the load option
- c) Select the Load Sequences option
- d) Click on Load Fasta File (.fasta) option to open a load prompt window,
- e) Click browse option to the folder containing the query file.
- f) Select the query file
- g) Click open to load the file
- h) The load prompt window will open, click on load
- i) The sequences will be loaded.
- j) Alternately you can press Ctrl+Shift+Z to open the load prompt window and continue from step e) to load the sequences

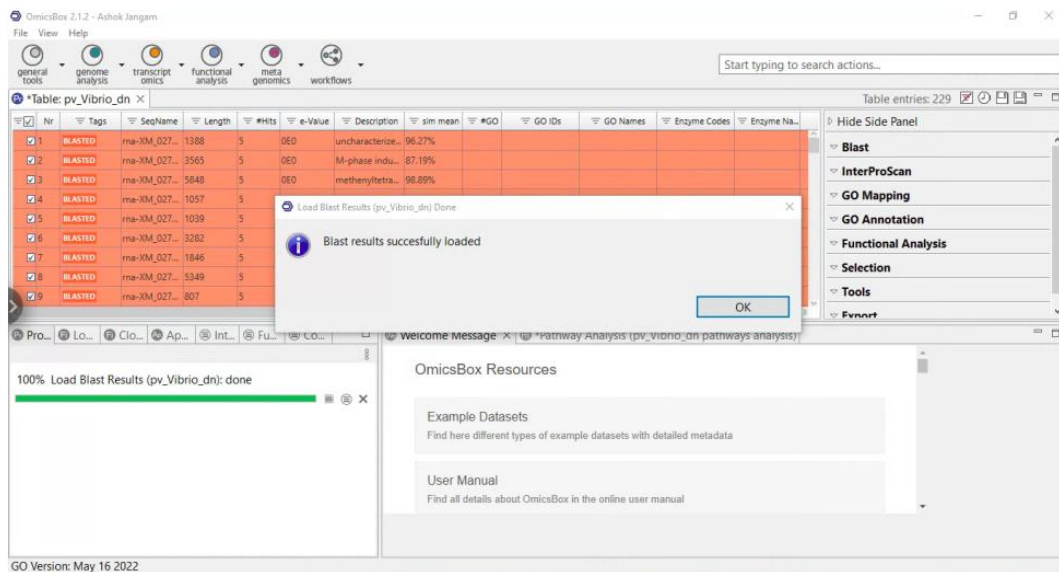
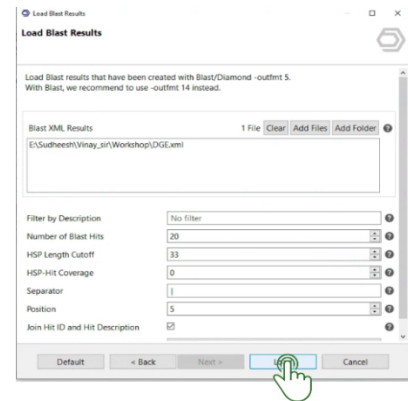
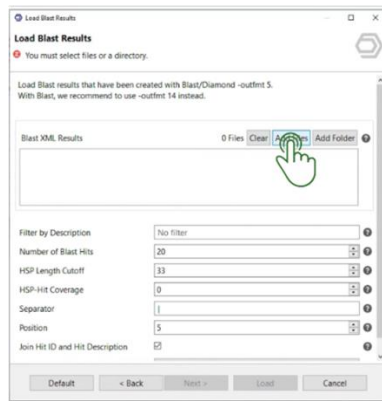
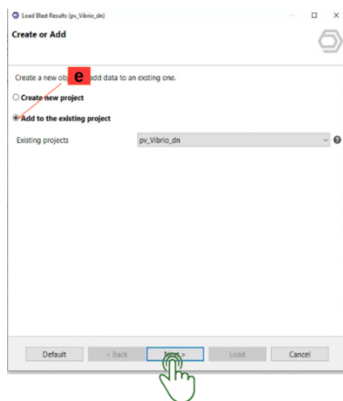
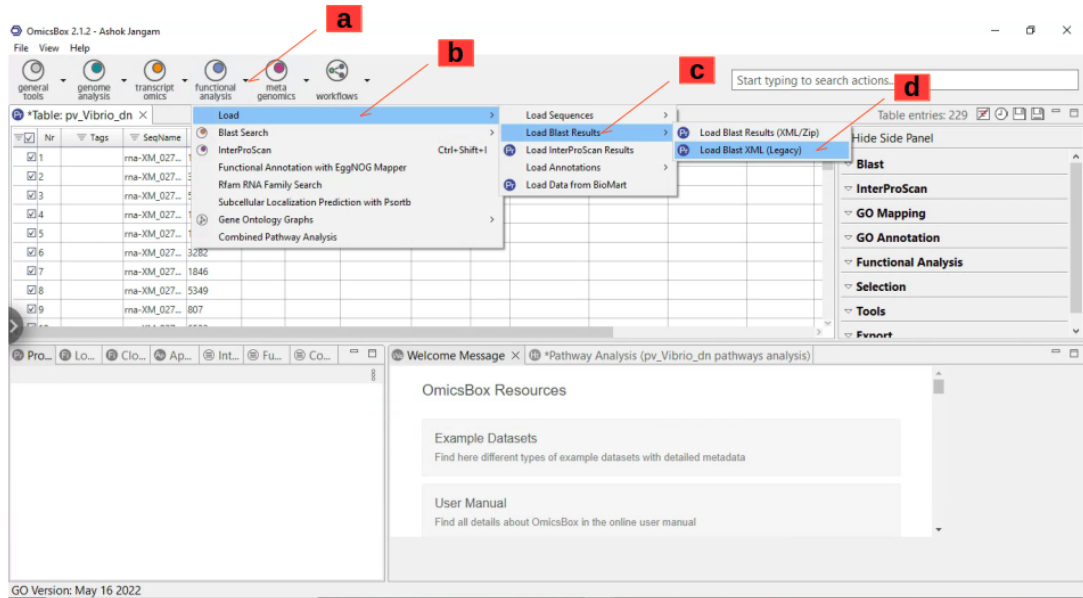


## Loading the Blast results to OmicsBox

Once the sequence is loaded follow the steps mentioned below to load the blast results

- a) Click on the dropdown menu under the Functional analysis module
- b) Select the load option
- c) Select the Load Blast results option
- d) Click on Load Blast XML (Legacy) option to open a load prompt window,

- e) click on Add to existing project option and click next
- f) Click on add files and select the file and click open
- g) Click on load button to load the results
- h) The blast results will be loaded and the able with blast results will turn orange and entries without blast hits will turn red.



## Homology based Annotation using InterProSCAN

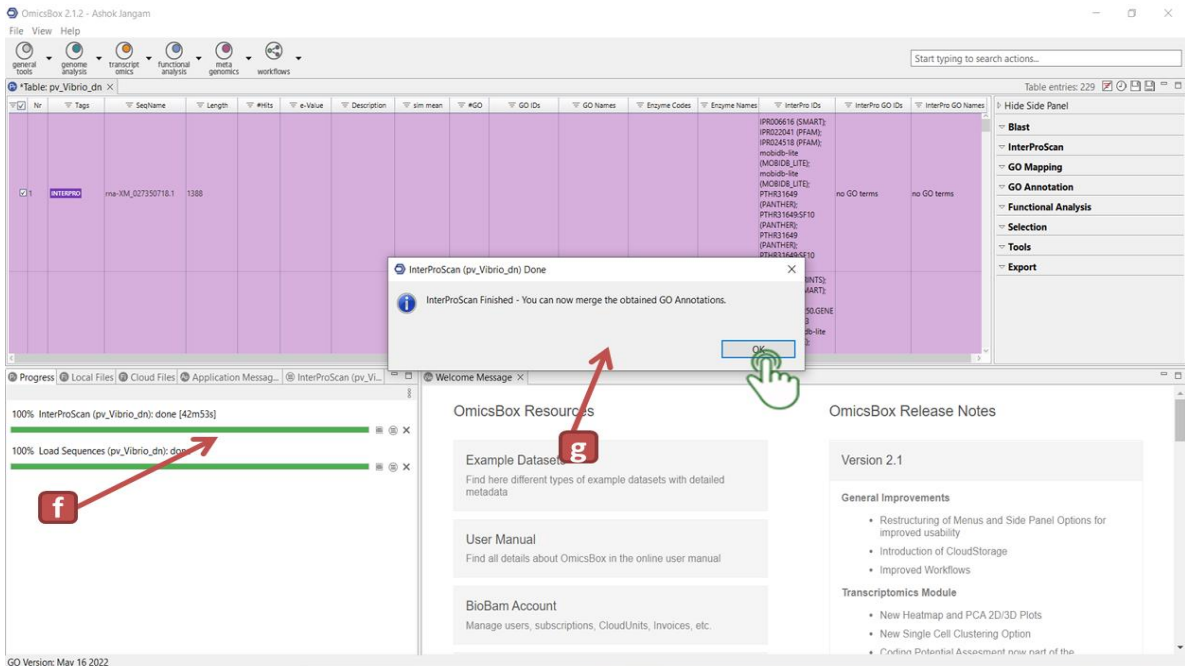
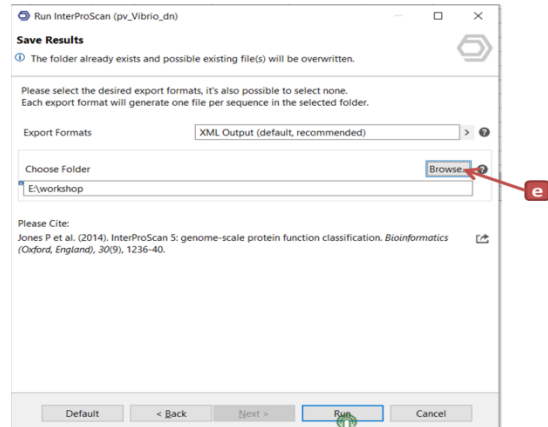
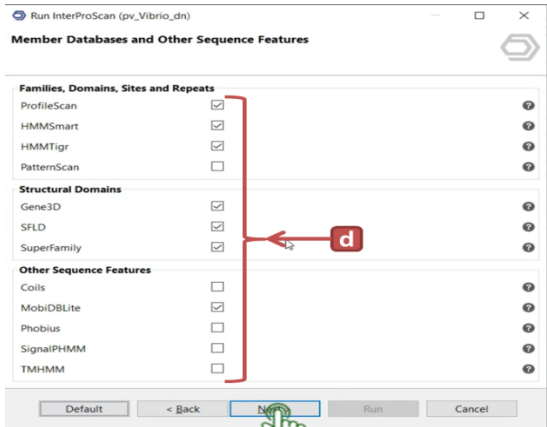
InterproSCAN analyses the query dataset against the InterPro database for sequence homology and annotates the sequences and classifies them into families, Domains, Conserved sites etc., The InterPro database is a consortium of 13 member databases: CATH, CDD, HAMAP, MobiDB Lite, Panther, Pfam, PIRSF, PRINTS, Prosite, SFLD, SMART, SUPERFAMILY and TIGRFams. For annotation using InterProSCAN we use a paid Software tool called *OmicsBox*.

### Running Interproscan

- click on the dropdown menu under Functional analysis module and click on InterProScan or Ctrl+Shift+I to open InterProScan options window.
- Select the EMBL-EBI-Interpro option and click next to open the InterProScan configuration window
- Enter your mail ID and click next
- Select the member databases and special features to be investigated and click next
- select the output format and output folder and click run
- The progress of the run can be monitored in the progress bar
- Once the run is complete the system notifies you with a popup and asks you to proceed with merging the annotations with blast results.
- An Interpro results chart is generated showing how many Interproscan results (IPS) have been obtained and how many Gene Ontology (GO) terms have been obtained

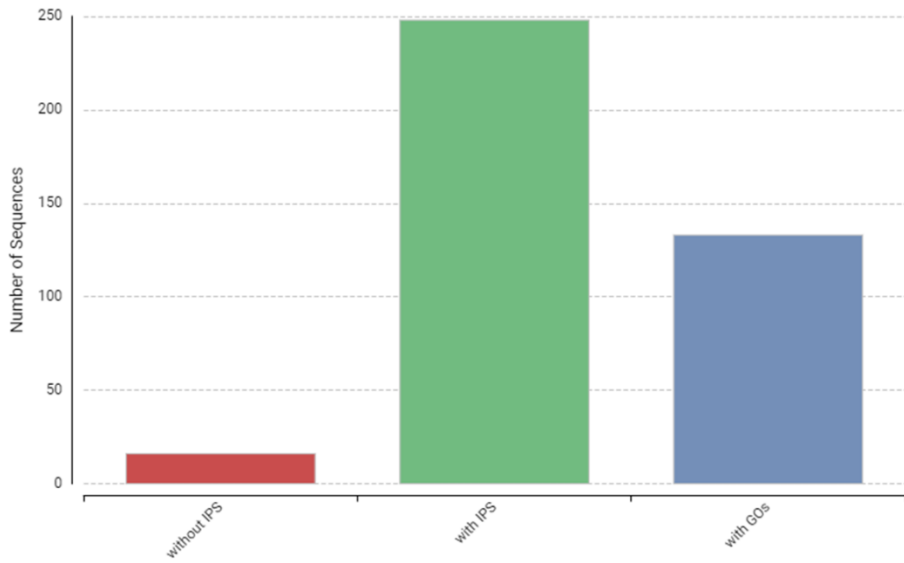
The screenshot illustrates the steps for running InterProScan in OmicsBox. It shows the 'functional analysis' menu, the 'InterPro Options' dialog where 'EMBL-EBI InterPro' is chosen, and the 'InterProScan Configuration 1' dialog where an email address is entered and various databases are selected for scanning. A progress bar at the bottom indicates the process is complete.





**h**

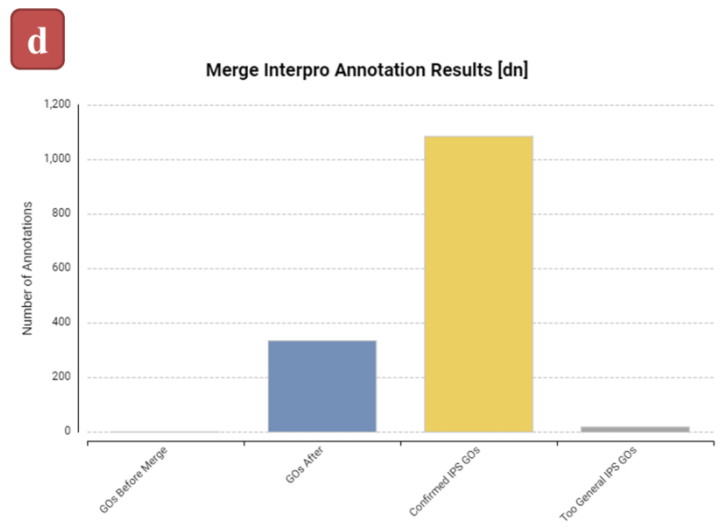
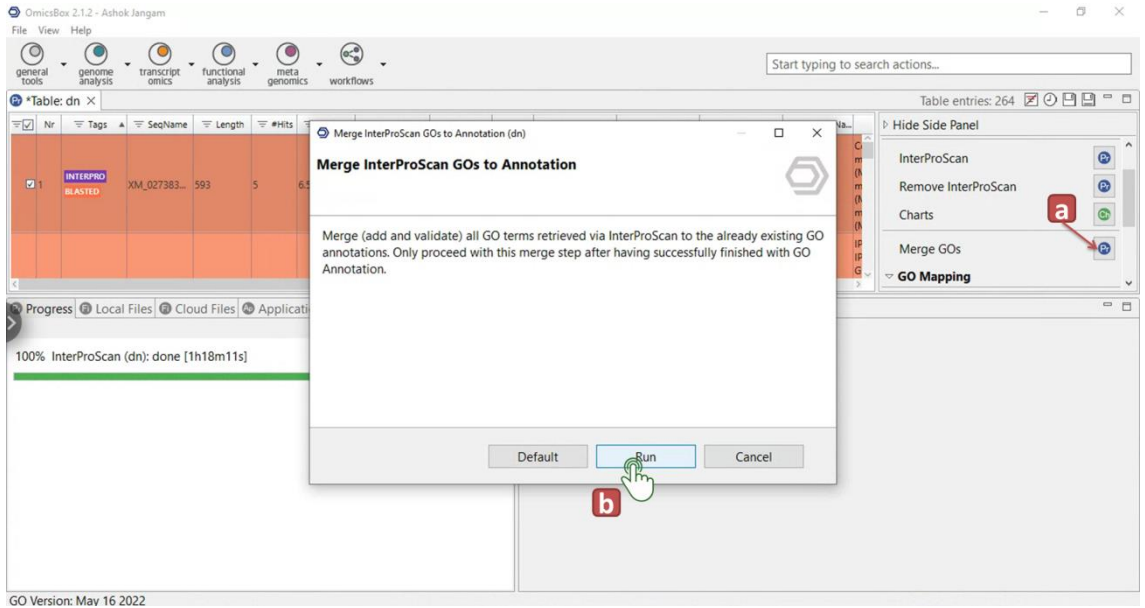
InterProScan Results [dn]



## Merging InterProScan Gene Ontologies and Blast Result

The InterProScan GOs results can now be added to the already existing annotations based on the BLAST results.

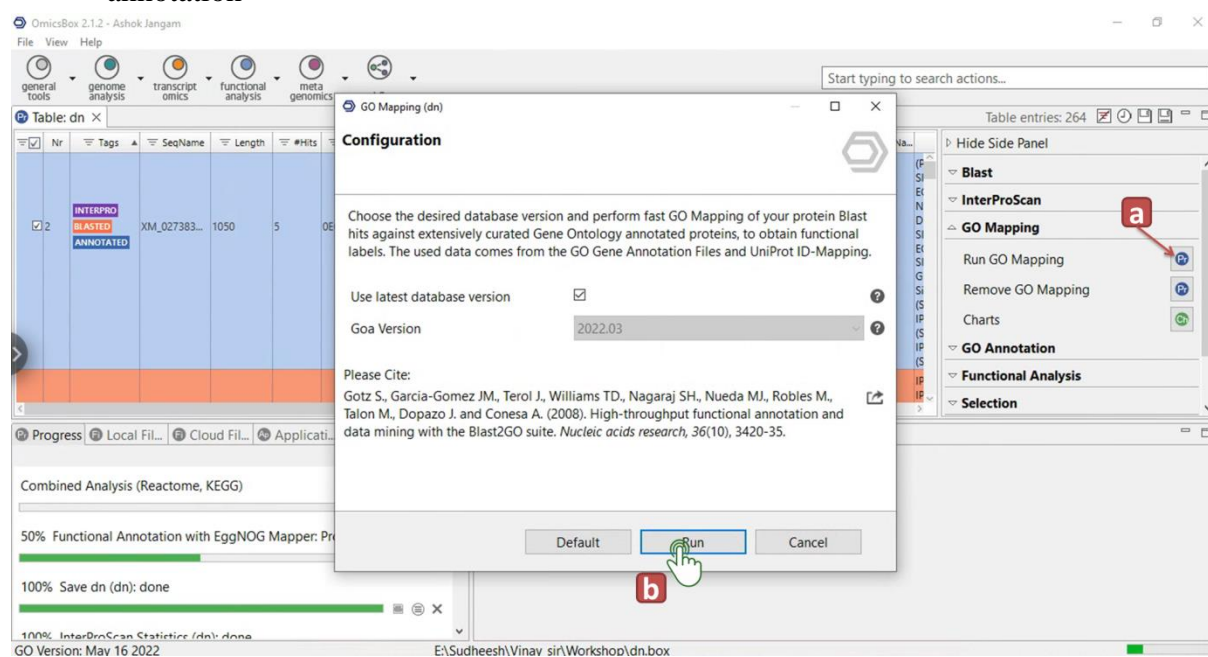
- a) Click on the InterProScan dropdown menu in the side panel and click on Merge GOs.
- b) The pop up window will appear and then click on run.
- c) Once the merging completes the entries with merged GO's will turn blue.
- d) Once the merge has finished a distribution chart is displayed in the Results menu showing the number of GOs that have been added to (or confirmed) the current annotation results.



## Running GO Mapping

Mapping is the process of retrieving GO terms associated with the Hits obtained by the BLAST search

- Click on the GO mapping dropdown menu in the side panel and click on Run GO Mapping option
- The configuration window will pop-up, click run to run Go mapping.
- Once mapping is done you'll be prompted to continue with gene ontology annotation

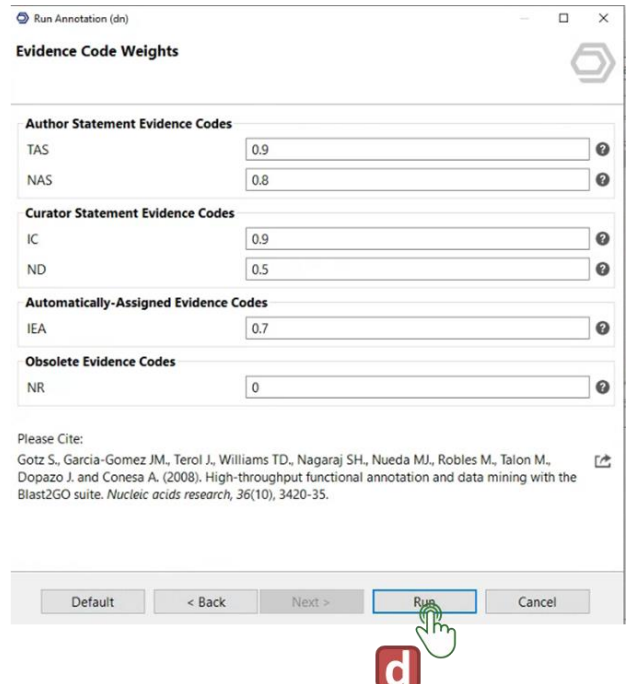
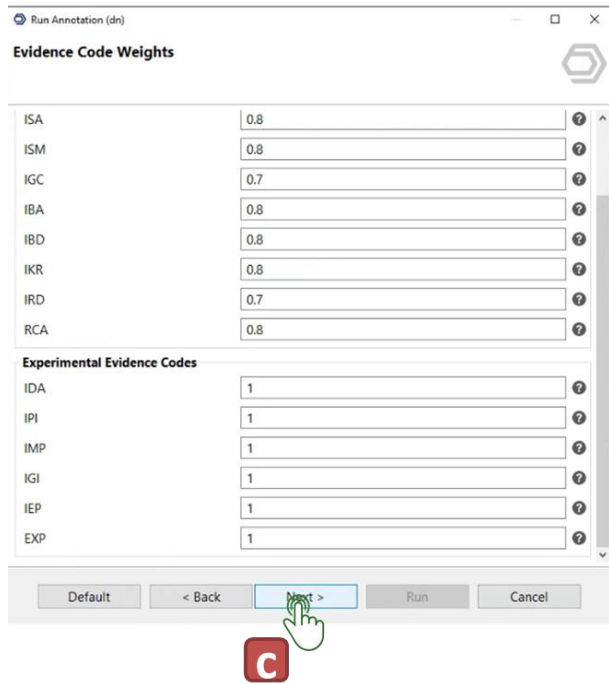
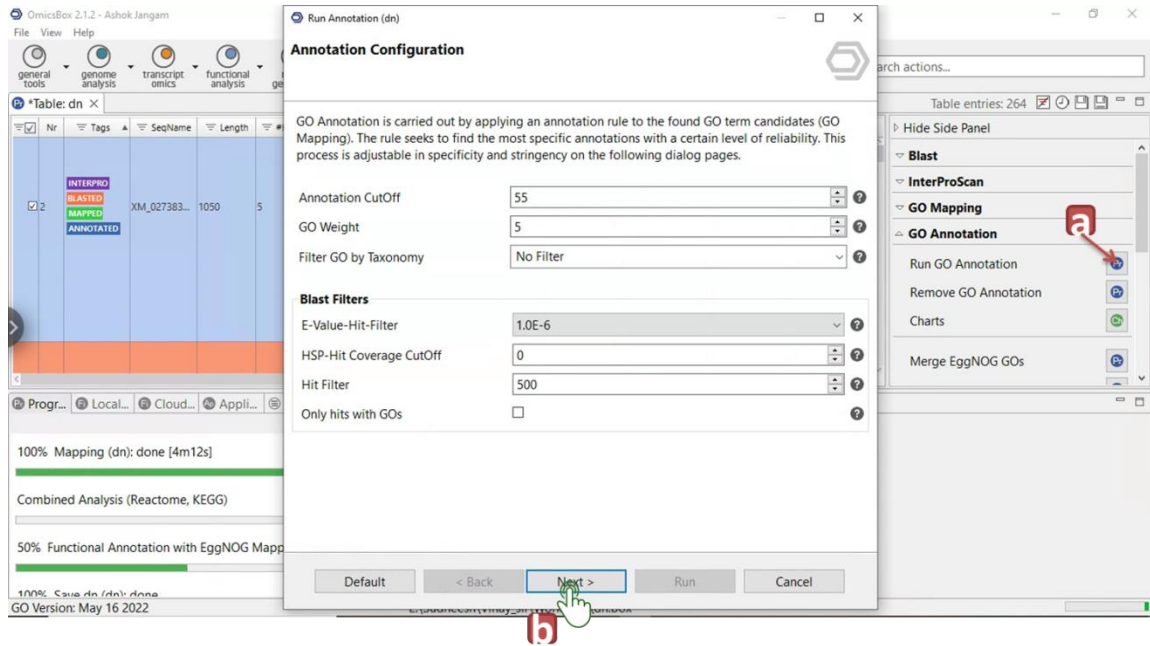


## Gene Ontology Annotaion

An annotation rule (AR) is used to the detected ontology terms to complete the GO annotation. The rule aims to locate the most precise annotations that are also reliable. Specificity and stringency of this method can be adjusted. The process of picking GO terms from the GO pool produced during the Mapping stage and allocating them to query sequences is known as an annotation rule.

- Click on the GO Annotation dropdown menu in the side panel and click on Run GO Annotation option
- The configuration window will pop-up, Select default values or apply filters to increase stringency. Click next to open the evidence code (EC) weights window.
- EC code weights can be modified depending on what you want. Note that in case of influence by evidence codes is not wanted, you can set them all at 1. Alternatively, when you want to exclude GO annotations of a certain EC (for example IEAs), you can set this EC weight at 0. Here we use default values and click next.

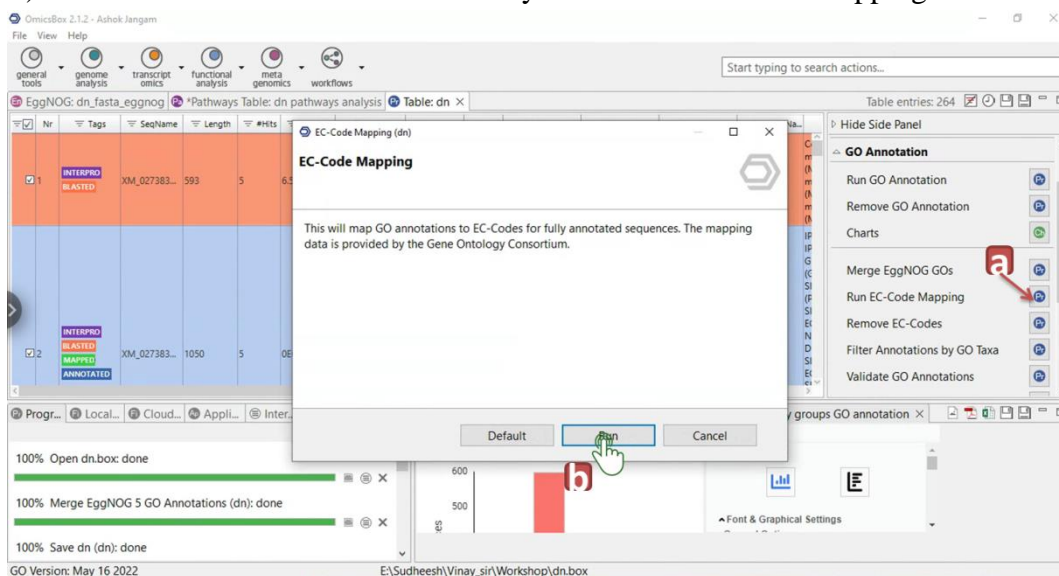
- d) After setting all the EC's to default click run to successfully annotate the transcripts
- e) Next continue with orthology based annotation against EggNOG database and merge the annotations



## Enzyme commission code mapping

The enzyme commission code mapping assigns the available enzyme commission codes for the GO terms to the annotation

- a) Click on the GO Annotation dropdown menu in the side panel and click on Run EC-code mapping option.
- b) Click on the run button to finish enzyme commission code mapping



## Orthology based Annotation using EggNOG

EggNOG-mapper is a tool for fast functional annotation of novel sequences (genes or proteins) using precomputed eggNOG-based orthology assignments. Obvious examples include the annotation of novel genomes, transcriptomes or even metagenomic gene catalogs. The use of orthology predictions for functional annotation is considered more precise than traditional homology searches, as it avoids transferring annotations from paralogs.

### Running EggNOG Annotation

- a) Click on the Functional analysis module in the taskbar and select Functional Annotation with EggNOG Mapper option in the dropdown menu
- b) In the input window, click on the add files option
- c) select the input fasta file and click open
- d) once the input file is added click on next
- e) The configuration window will open and we can select the default options and click run.
- f) The result table summarizes all annotations that could be transferred with EggNOG Mapper and also generates the summary report which can be saved in pdf format.
- g) Save the result table in \*.box format and the file can be used to merge the annotations with already available GO annotations.

OmicsBox 2.1.2 - Ashok Jangam

File View Help

general analysis transcriptomics proteomics metabolomics workflows

Table: pv\_Vibrio\_dn

Load  
Blast Search  
InterProScan  
Functional Annotation with EggNOG Mapper  
Rfam RNA Family Search  
Subcellular Localization Prediction with Psortb  
Gene Ontology Graphs  
Combined Pathway Analysis

Functional Annotation with EggNOG Mapper

Input

EggNOG Mapper is a tool for fast functional annotation of novel sequences (genes or proteins) using pre-annotated eggNOG-based orthology assignments. The use of orthology predictions for functional annotation avoids transferring annotations from paralogous e.g. duplicate genes with a higher chance of being involved in functional divergence. Note: This tool makes use of free cloud computation resources. This is an introductory offer and may change in a future release depending on the overall resource consumption of the feature.

Genes or Proteins

pv\_Vibrio\_dn.fasta

Configuration

Target Ontologies: All  
Egg Evidence: Non-Electronic

Version Details: eggNOG Mapper 2.1.0 with eggNOG 5.0.2

Help: Maria Lopez et al. (2018). eggNOG 5.0: a hierarchical, functionally and phylogenetically annotated orthology resource based on 5086 organisms and 2083 species. *Nucleic Acids Res.* 47(11): D108-D114.

User Manual  
Find all details about OmicsBox in the online user manual.

BioBam Account  
Manage users, subscriptions, CloudUnits, Invoices, etc.

GO Version: May 16 2022

OmicsBox 2.1.2 - Ashok Jangam

File View Help

general analysis transcriptomics proteomics metabolomics workflows

Table: pv\_Vibrio\_dn

Query ID	Gene Name	EggNOG Description	E-Value	BL Score	Best Tax Level	EC Codes	#GOs	GO Names	KEGG KO	KEGG Pathway
na-XM_027353480.1	EXOR1		3.90E-42	179.9	Rhizobiales		0		K14570	map03008
na-XM_027353282.1			2.20E-26	129	Metazoa		8	P:GO:0006357, ...	Pmulo-organi...	
na-XM_027353729.1		serine-type endopeptidase activity. It is involved in the protein binding. biogenic. It is involved in the biogenic...	4.40E-27	163.9	Arthropoda		8	P:GO:0016485, ...	Pmulo-organi...	
na-XM_027352743.1	sn		1.70E-205	722.3	Insecta		18	P:GO:0015175, ...	Pmulo-organi...	K17455
na-XM_027354437.1	Pu1f1	Belongs to the PDGF VEGF growth factor family.	2.80E-12	79	Drosophilidae		21	P:GO:0007435, ...	Pmulo-organi...	map04013
na-XM_027356379.1			2.70E-16	102.4	Arthropoda		11	C:GO:0009897, ...	Pmulo-organi...	K20228
na-XM_027356380.1			3.60E-11	91.3	Nematocera		11	C:GO:0009897, ...	Pmulo-organi...	K20228
na-XM_027356381.1		Sushi, midgen and EGF-like	1.20E-11	79	Eukaryota		0			
na-XM_027356520.1	dyf1	Zona pellucida-like domain	4.20E-24	119.4	Drosophilidae		9	P:GO:0016476, ...	Pmulo-organi...	K18756
na-XM_027356639.1	BCC1	Bicaudal C homology	2.10E-9	69.7	Actinopterygii		10	P:GO:0007368, ...	Pmulo-organi...	K18756
na-XM_027356822.1		proton containing a HSPS condensation (Elongation)	1.40E-2	48.1	Nostocales		0			
na-XM_027357157.1	trae	RNA recognition motif (a.k.a. RRM, RBD, or RNP domain)	3.50E-41	307.8	Hymenoptera		7	P:GO:0008049, ...	Pmulo-organi...	K13208
na-XM_027357176.1	rtar	RNA recognition motif (a.k.a. RRM, RBD, or RNP domain)	1.40E-47	195.3	Hymenoptera		14	C:GO:0015630, ...	Pmulo-organi...	K13208
na-XM_027357512.1	RAB6	Rab6 and receptor beta	6.00E-22	112.1	Altothemia		0		K05528	map05200, map...
na-XM_027358964.1		N-terminal domain of endonuclease	5.20E-94	351.3	Nematocera		0		K13948	map05200, map...
na-XM_027359584.1		Protein domain of beta-2-microglobulin	9.70E-4	14.3	Nematocera		0		K20374	map05200, map...

Progress: Local Files | Cloud Files | Application... | InterProScan... | Functional... | Welcome Message | \*EggNOG Annotation Report\*

100% Functional Annotation with EggNOG Mapper: done [33m29s]

100% InterProScan (pv\_Vibrio\_dn): done [42m53s]

100% Load Sequences (pv\_Vibrio\_dn): done

### EggNOG Annotation Report

Input Data

pv\_Vibrio\_dn.fasta

General Information

Total amount of input sequences: 229  
Average length: 1870.0  
Number of GO annotated sequences: 47 / 20.52%  
Number of GO annotations: 477  
Average GOs per sequence: 10.15

GO Version: May 16 2022

OmicsBox 2.1.2 - Ashok Jangam

File View Help

Save EggNOG Annotation Report

Organize New folder

WSSV\_files

OneDrive - Person

This PC

3D Objects

Desktop

Documents

Downloads

Music

Pictures

Videos

OS (C)

New Volume (E)

File name: eggnog\_annotation\_report

Save as type: PDF

Cancel

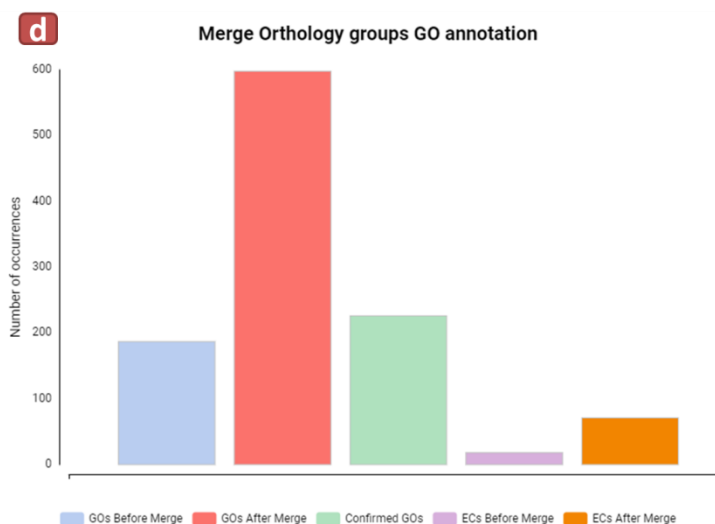
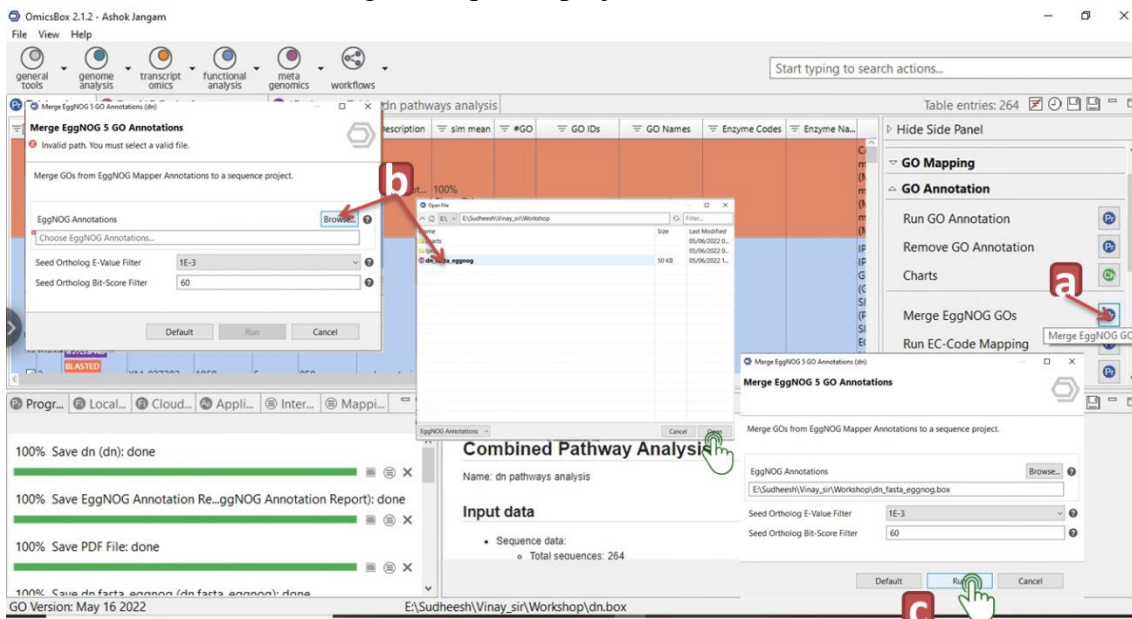
Table: pv\_Vibrio\_dn

Best Tax Level	EC Codes	#GOs	GO Names	KEGG KO	KEGG Pathway
Rhizobiales		0		K14570	map03008
Metazoa		8	P:GO:0006357, ...	Pmulo-organi...	
Arthropoda		8	P:GO:0016485, ...	Pmulo-organi...	
Insecta		18	P:GO:0015175, ...	Pmulo-organi...	K17455
Drosophilidae		21	P:GO:0007435, ...	Pmulo-organi...	map04013
Arthropoda		11	C:GO:0009897, ...	Pmulo-organi...	K20228
Nematocera		11	C:GO:0009897, ...	Pmulo-organi...	K20228
Eukaryota		0			
Drosophilidae		9	P:GO:0016476, ...	Pmulo-organi...	K18756
Actinopterygii		10	P:GO:0007368, ...	Pmulo-organi...	K18756
Nostocales		0			
Hymenoptera		7	P:GO:0008049, ...	Pmulo-organi...	K13208
Hymenoptera		14	C:GO:0015630, ...	Pmulo-organi...	K13208
Altothemia		0		K05528	map05200, map...
Nematocera		0		K13948	map05200, map...
Nematocera		0		K20374	map05200, map...

GO Version: May 16 2022

## Merging EggNOG Annotation and GO annotation

- Select the GO annotated project and click on the GO annotation dropdown menu in the side panel and select Merge EgNOG GO's
- Click on browse and select the EggNOG result saved in \*.box format
- Click run to initiate the merging of annotations
- A bar chart will be generated showing the showing the total number of GOs and ECs added to the original sequence project

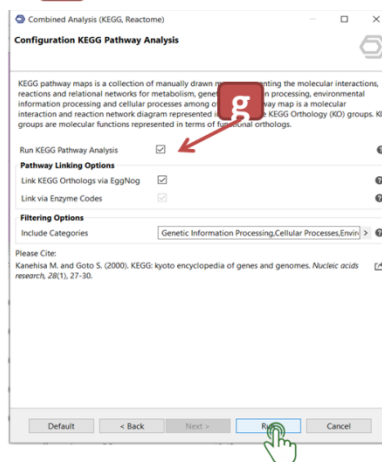
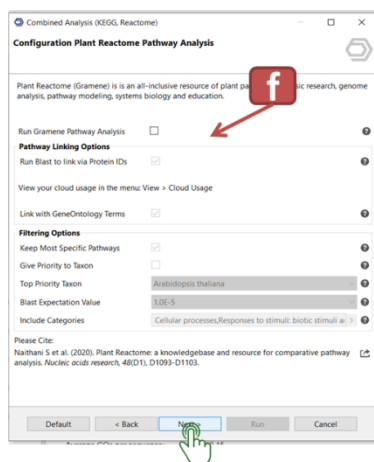
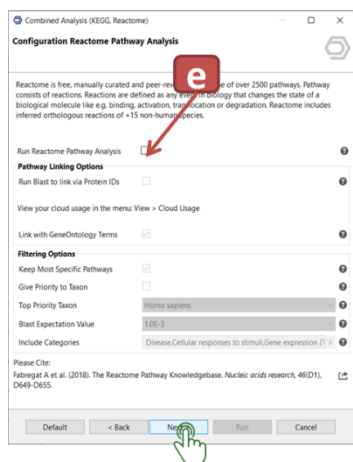
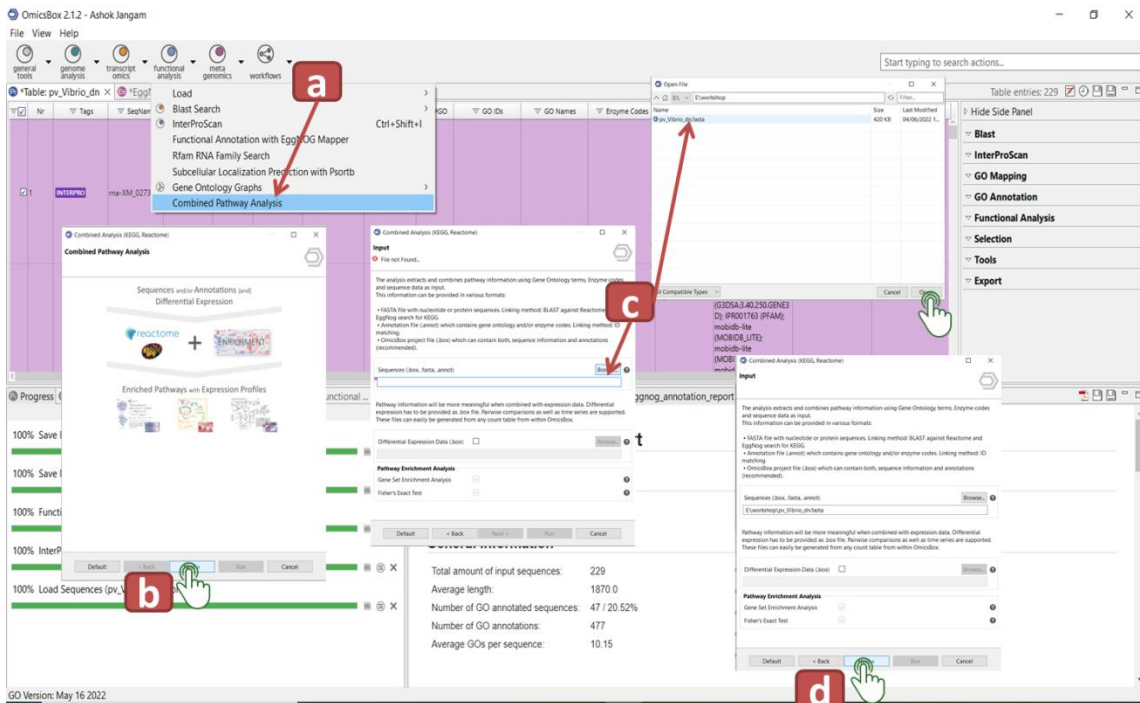


## Pathway analysis using KEGG Database

Pathway analysis is a powerful method for quickly getting a high-level overview of the biological mechanisms at work in our data, summarising the information in a way that considerably improves the capacity to comprehend the findings. KEGG is a database of manually produced pathway maps that represent knowledge of molecular interaction,

reaction, and relational networks in metabolism, genetic information processing, environmental information processing, cellular processes, organismal systems, human diseases, and drug development.

- Click on the Functional analysis module in the taskbar and select Combined Pathway analysis option in the dropdown menu
- Click next in the popup window
- Select the browse option and add the Annotated box file or the query fasta file.
- Once the file is selected click next in the input window
- Unselect the Run Reactome Pathway Analysis and click next. This option can be used for human datasets or the other 15 taxons present in the reactome database
- Unselect the Run Gramene Pathway analysis and click next. This option can be used if dealing with plant based datasets.
- Select the Run KEGG Pathway Analysis option and click run.
- After identifying the pathways associated with the sequences a table will open.

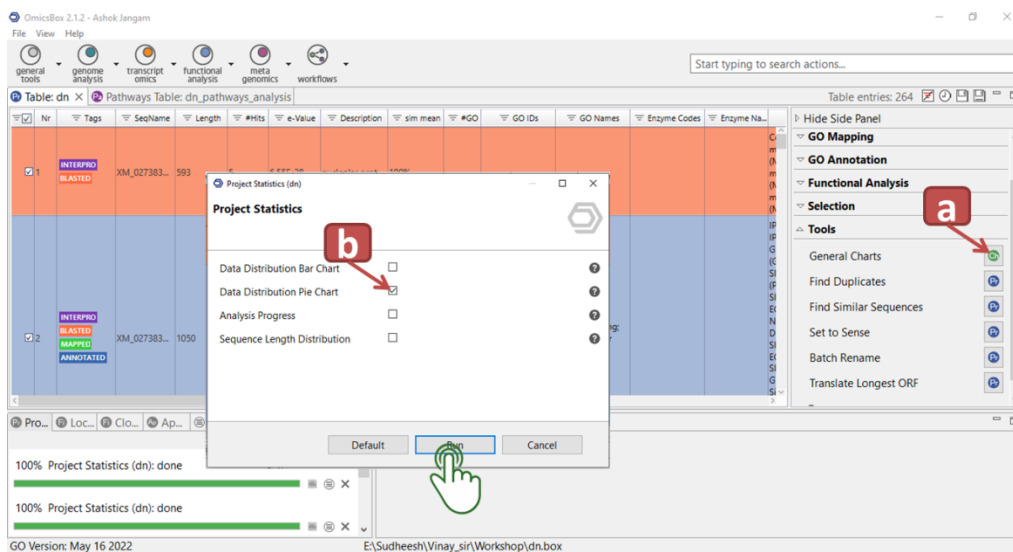




# Analysing Annotation Charts

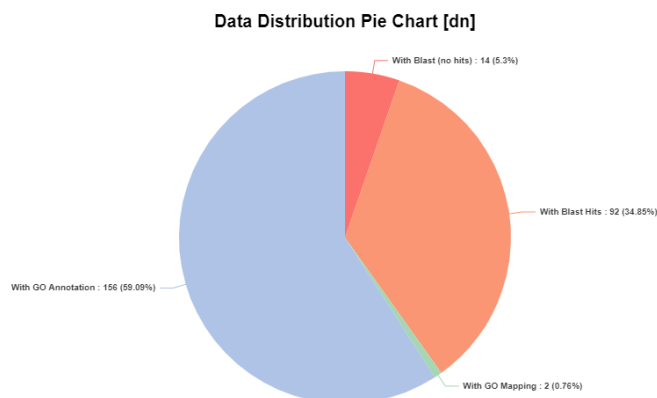
## Generating Annotation Charts

- Click on the Tools dropdown menu in the side panel and click on General charts
- Select the required charts and press run, Here we selected only Data distribution pie chart
- The charts will be generated in the bottom right panel
- Similarly for BLAST, InterPro, GO Mapping and GO Annotation, Click on the respective dropdown menu in the side panel and click on charts
- Select the required charts and press run
- The charts will be generated in the bottom right panel



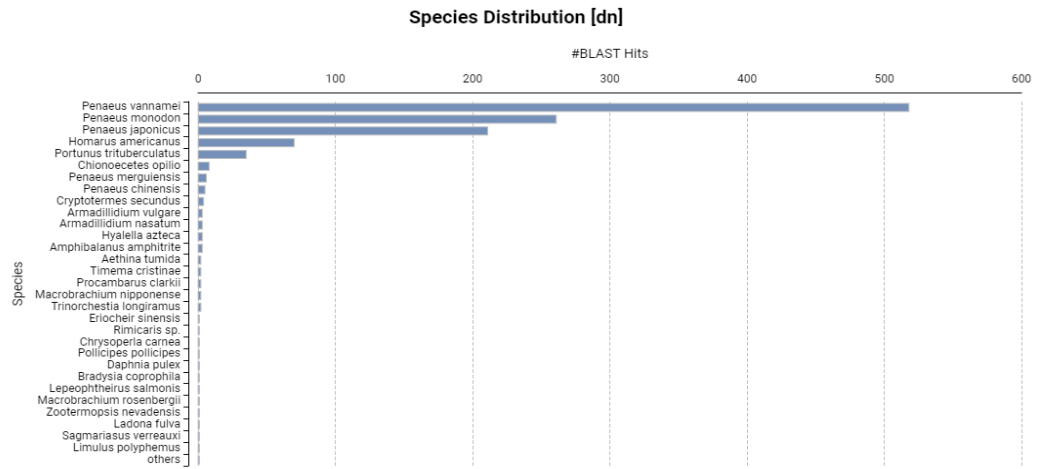
## Analysing some of the Generated Charts

- Data distribution chart: the chart summarizes the distribution of sequences with blast results, without blast results, with GO annotation and with GOMapping.

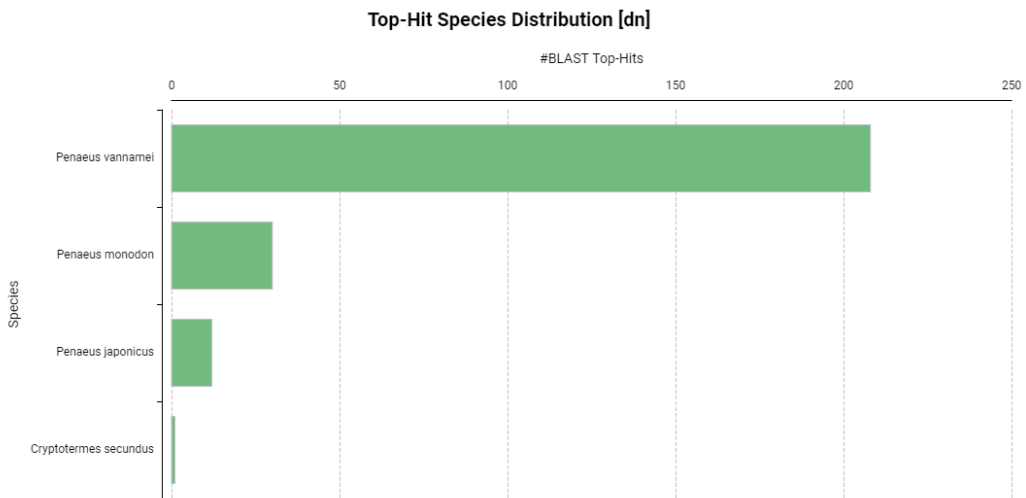


- Species distribution chart shows the top species for all the hits, here the top species gave the maximum number of hits for the query dataset. The top species *Penaeus*

*vannamei* shows maximum number of hits and is the species used for this analysis hence the result is satisfactory. Species distribution chart is helpful in identifying the major contributor of a meta-transcriptome data where the transcripts obtained will be from multiple unknown hosts.

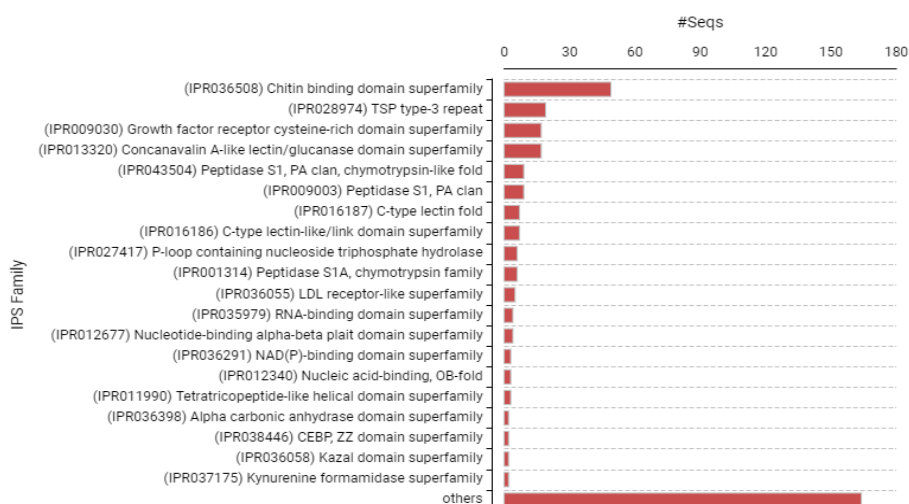


c) Top-hit species distribution chart shows the species contributing to the top hit of the query sequences and more or less similar to the species distribution chart except for the fact that it considers only the top hits for deducing the chart. Here also *Penaeus vannamei* tops the chart.



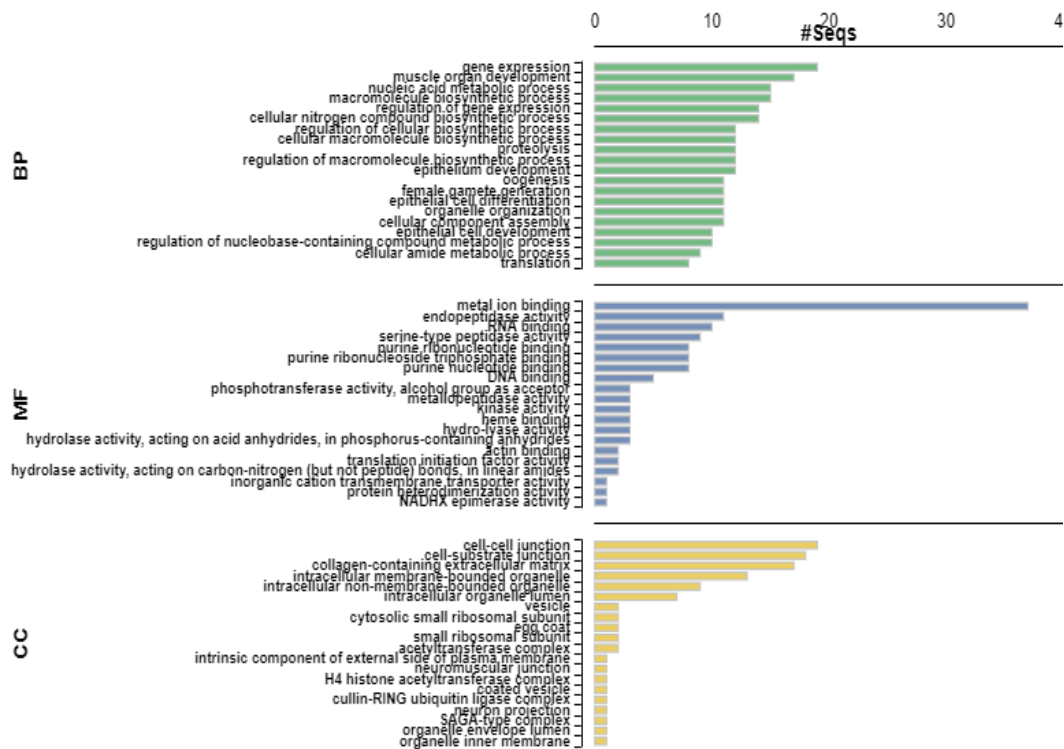
d) InterproScan Families Distribution; this chart depicts the families present in the annotated sequences. In our case it shows that Chitin Binding Domain family of genes are dominating. Similar charts can be generated for InterproScan Conserved Domains, Repeats and Sites.

### InterProScan Families Distribution [dn]

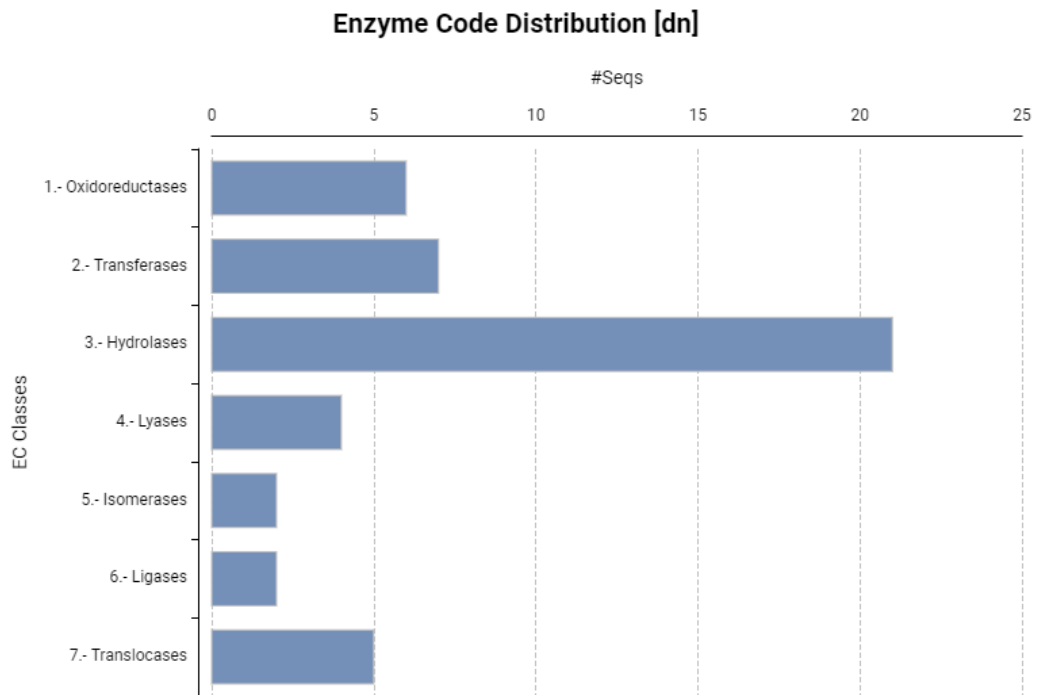


e) GO Distribution By Level 5: This chart depicts the distribution of genes based on their function into Three main categories the Biological Processes, the Molecular Functions and Cellular Components. In this case the Gene expression proteins, Metal ion binding protein and cell-celljunction proteins are dominating in the biological processes, the molecular functions and cellular components respectively.

### GO Distribution by Level (5) - Top 20



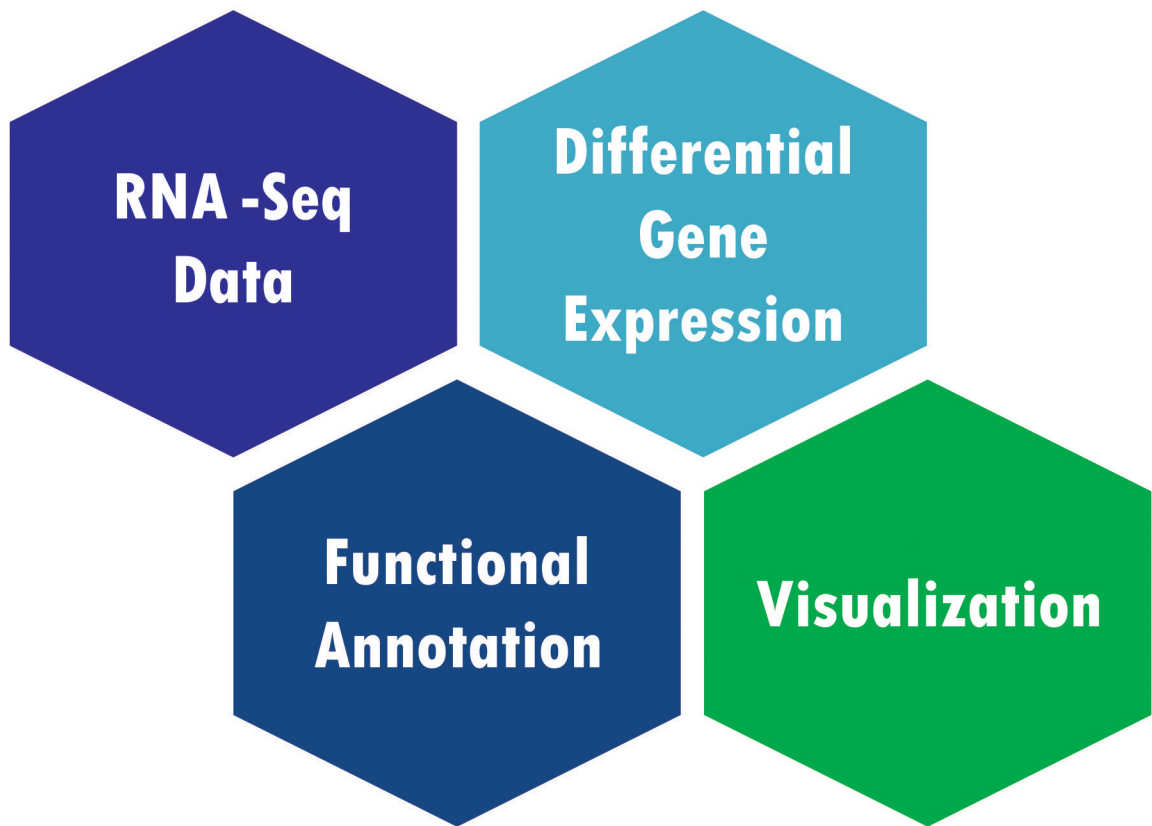
- f) Enzyme code Distribution: this chart describes the distribution of main enzyme classes among the annotated genes. In this case it shows that the hydrolases are the highest in numbers among the annotated genes with enzyme code.



\*\*\*



**Nutrition Genetics and Biotechnology Division**



**CIBA**

