12TH DECEMBER 2013

ASSEMBLY AND ANNOTATION OF METAGENOMES

STEP1: getting the files you need

Login in the server and go to your folder. Make a new directory:

```
pwd
 /home/guest/group1/username
mkdir DCM
cd DCM
```

Ok, rule to follow for every command in today's exercise. Run the "ls" and "pwd" command after EVERY COMMAND. You should always be working in YOUR home directory /home/guest/group1/username/DCM. If you are somewhere else, please go back to your own folder.

Copy metagenomic data files of deep chlorophyll maximum to your folder

```
cp /home/guest/data/day4/DCM1.fastq .
cp /home/guest/data/day4/DCM2.fastq .
```

Check if the files are in your folder

```
ls
```

should show you that DCM1.fastq and DCM2.fastq are present

STEP2: preparing the files for assembly

Check header of illumina files

```
head  DCM1.fastq
```

```
@HWI-ST1207:173:C14H3ACXX:6:1101:1488:1970 1:N:0:CGATGT
NGAAGTCAAAGGTAAAACAGGTTCATTAACTGCATTACCAATTATCGAAACACAAGCAGGAGACGTTTCTGCTTTCGTTCCGACCAATGTGAT
TTCAATAA
+
#1:BDDDDHHFFHG<EGGIGGIF>BHEGCHBBEHHIII@C?CGEHE=GIIBH=F7?CCGIFH2@EGDHC>EEHDDE;?;?CC;@BB(:@>@;>
>>>A:AA:
@HWI-ST1207:173:C14H3ACXX:6:1101:1272:1972 1:N:0:CGATGT
NGTTAAGATTTCTGGTATAGCAAAAGGTTCTGGAATGATTGCTCCTAATATGGCAACTATGTTATCTTTTATATTTACTGATGCAAATATACC
TTCGATAN
+
#1=?DDFFAHGHFIEFGGCEHHIIIHGCHIGIIEHIIIGIIJIIIJGIJIJJGHJIGIBHICHGIIIGGIIJIGIHGIJJEIHHFCEE@CDEE
DDE2=C?#
@HWI-ST1207:173:C14H3ACXX:6:1101:1492:1987 1:N:0:CGATGT
NTTCGATTCAACTATGGTATTTCTGAACGCCAGCTTGTTCGCTATGTAAAAAAAGCTCGTGCTCAGGAGGGTTCCACAGGAACAAATCTCTTA
AAGCTTCT
```

Rename illumina reads so they have correct names (/1 and /2 are needed for the velvet assembler). Please read the help of the program illumina2mira.

```
/home/guest/bin/illumina2mira –h
```

Here is the command line to use to format the read identifiers correctly. First let's rename the file containing the /1 reads

```
    /home/guest/bin/illumina2mira -i DCM1.fastq -o dcm_1.fastq -pair 1 -rename "DCM-" -pattern="\@HWI-ST" &
```

and now for the /2 reads

```
    /home/guest/bin/illumina2mira -i DCM2.fastq -o dcm_2.fastq -pair 2 -rename "DCM-" -pattern="\@HWI-ST" &
```

Run "top" to see if the programs are running. It can take a while, you should get some message from each of the programs above that they have fixed the reads. You can press "q" to quit the top program.

Check how the read names look now:

```
head dcm_1.fastq
```

```
@DCM-1/1
NGAAGTCAAAGGTAAAACAGGTTCATTAACTGCATTACCAATTATCGAAACACAAGCAGGAGACGTTTCTGCTTTCGTTCCGACCAATGTGAT
TTCAATAA
+
#1:BDDDDHHFFHG<EGGIGGIF>BHEGCHBBEHHIII@C?CGEHE=GIIBH=F7?CCGIFH2@EGDHC>EEHDDE;?;?CC;@BB(:@>@;>
>>>A:AA:
@DCM-2/1
NGTTAAGATTTCTGGTATAGCAAAAGGTTCTGGAATGATTGCTCCTAATATGGCAACTATGTTATCTTTTATATTTACTGATGCAAATATACC
TTCGATAN
+
#1=?DDFFAHGHFIEFGGCEHHIIIHGCHIGIIEHIIIGIIJIIIJGIJIJJGHJIGIBHICHGIIIGGIIJIGIHGIJJEIHHFCEE@CDEE
DDE2=C?#
@DCM-3/1
NTTCGATTCAACTATGGTATTTCTGAACGCCAGCTTGTTCGCTATGTAAAAAAAGCTCGTGCTCAGGAGGGTTCCACAGGAACAAATCTCTTA
AAGCTTCT
```

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

Now the names are fixed. We are now going to do a quality trimming on each file, using the program trimfastq

```
/home/guest/bin/trimfastq dcm_1.fastq &
/home/guest/bin/trimfastq dcm_2.fastq &
```

Now we wait till the trimming is done. You should get a message when it is done. We will get two files as output

dcm_1.fastq.trimmed
dcm_2.fastq.trimmed

Take a look at both these files using the "more" command.

```
more dcm_1.fastq.trimmed
more dcm_2.fastq.trimmed
```

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

We are going to combine these into a single file for input to the assembly program velvet.

```
/home/guest/bin/velvet-shuffleSequences_fastq.pl  dcm_1.fastq.trimmed
dcm_2.fastq.trimmed dcm-combined.fastq
```

where dcm_1.fastq.trimmed and dcm_2.fastq.trimmed are our trimmed fastq files and dcm-combined.fastq is the file containing both reads in interleaved format. Once this is done, you should check the file dcm-combined.fastq, like this:

```
head dcm-combined.fastq
```

```
@DCM-1/1
NGAAGTCAAAGGTAAAACAGGTTCATTAACTGCATTACCAATTATCGAAACACAAGCAGGAGACGTTTCTGCTTTCGTTCCGACCAATGTGAT
TTCAATAA
+
#1:BDDDDHHFFHG<EGGIGGIF>BHEGCHBBEHHIII@C?CGEHE=GIIBH=F7?CCGIFH2@EGDHC>EEHDDE;?;?CC;@BB(:@>@;>
>>>A:AA:
@DCM-1/2
NGAAGTCAAAGGTAAAACAGGTTCATTAACTGCATTACCAATTATCGAAACACAAGCAGGAGACGTTTCTGCTTTCGTTCCGACCAATGTGAT
TTCAATAA
+
#1:BDDDDHHFFHG<EGGIGGIF>BHEGCHBBEHHIII@C?CGEHE=GIIBH=F7?CCGIFH2@EGDHC>EEHDDE;?;?CC;@BB(:@>@;>
>>>A:AA:
```

Or by using the more command

```
more dcm-combined.fastq
```

You will see that the read pairs are now present together, i.e. DCM-1 is the common read pair name and /1 and /2 reads (paired end reads) are present one after the other. This is how it needs to be before we can begin assembly using velvet.

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

STEP3: using velvet to assemble your metagenome

We can now submit this file (dcm-combined.fastq) to velvet for assembly. Running velvet needs 2 commands first "velveth" and then "velvetg".

The first step, running velveth. You can read a bit about the various options of the program by just typing

```
velveth
```

and here is the complete command that we will use to input our data to this program

```
velveth DCM-K51 51 -shortPaired -fastq dcm-combined.fastq
```

Here DCM-K51 is the name of the folder which will contain all the result files. 51 is the kmer size that we are going to use for the assembly. Normally, we should use an even larger kmer size (say, 61, 71). The -shortPaired option tells velveth that the reads are short and paired (pretty clear) and the –fastq indicates that input file dcm-combined.fastq is in fastq format.
REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

Once the velveth program has finished, we go to the next step, which is the assembly step, by invoking the velvetg program. As for velveth, you can read the help of this program by simply typing

```
velvetg
```

Ok, here is the command that we are going to use to assemble our data

```
velvetg DCM-K51 -ins_length 300 -cov_cutoff auto
```

where -ins_length is the insert size of the paired end library, in this case 300bp.

once this is finished, we can check the results..

```
cd DCM-K51
```

```
pwd
/home/guest/group1/username/DCM/DCM-K51
```

Let us see the files in this directory by using the ls command

```
ls
Sequences
Roadmaps
PreGraph
Graph
contigs.fa  <<- this file contains the final assembled contigs in fasta format
stats.txt
LastGraph
Log
```

And take a look at the contigs.fa file, like this…

```
more contigs.fa
```

How many contigs are in this file ?

```
grep –c ">" contigs.fa
```

You can now also find out what's the length of the longest contig in your file using the following command…

```
cat contigs.fa | lenseq fasta | sort –nk2
```

also try the "seqstat" command

```
seqstat contigs.fa
```

You can find more details about the velvet assembler in:
http://www.ebi.ac.uk/~zerbino/velvet/

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

STEP4: organizing your contigs

The contigs in the final fasta file produced by velvet are not sorted by size. We will use the program, contig_arrange to sort them by length and also to give some more sensible names to each of the contigs.

```
/home/guest/bin/contig_arrange -i contigs.fa -o CONTIGS.FA -label "DCM-C"
```

the input file is the fasta file of contigs produced by velvet, CONTIGS.FA is the name of our output file containing the sorted contigs. and the -label option allows us to rename the contigs in the following way. the longest contig will be called DCM-C1, and the next
one DCM-C2 and so on. Let us check the lengths of the longest contigs

```
cat CONTIGS.FA | lenseq fasta | head
DCM-C113776 NODE_76_length_13726_cov_10.716669
DCM-C210129 NODE_6817_length_10079_cov_8.646989
DCM-C39394  NODE_3029_length_9344_cov_8.805651
DCM-C48214  NODE_7920_length_8164_cov_9.482484
DCM-C57634  NODE_111_length_7584_cov_9.032701
DCM-C67540  NODE_105_length_7490_cov_9.753538
DCM-C77025  NODE_10856_length_6975_cov_10.339642
DCM-C86444  NODE_347_length_6394_cov_9.524711
DCM-C96285  NODE_17758_length_6235_cov_8.765357
DCM-C10     6108  NODE_982_length_6058_cov_9.945361
```

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

STEP5: annotation

We are going to annotate the 5 longest contigs obtained in the assembly. Extract the first 5 contigs from the file CONTIGS.FA

```
cat CONTIGS.FA | faslice 1 5 > top5.fna
```

now we are going to do gene prediction on these contigs and make a nice looking genbank file with all the protein coding genes marked in it.

```
/home/guest/bin/run-prodigal.pl -i top5.fna -meta -prefix TOP5
```

input file is top5.fna, a prefix allows a uniform naming of all output files produced by the program. The -meta option tells this program to do gene prediction in metagenomic mode. Gene prediction is done using the program prodigal.

This will produce 3 output files
TOP5.gbk <-- genbank file
TOP5.faa <-- fasta file containing all the protein sequences
TOP5.ffn <-- fasta file containing all the gene sequences

You can look into each file now using "more"

```
more TOP5.gbk
more TOP5.faa
more TOP5.ffn
```

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

We now at least know where the genes are in these contigs. But we still have no idea of their functions. We are going to now compare the protein sequences to the ncbi nr database and find the best hit of each.

```
blastall -i TOP5.faa -m 8 -d nr -e 1e-5 -b 1 -v 1 -a 5 -o TOP5.faa.blastp -p blastp
```

-i   the input fasta file
-m   8 here specifies that output results should be in tabular format
-d   nr means the sequences should be compared to the ncbi non-redundant database that is available and setup already
-e   1e-5 specifies that only hits with this e-value or less should be considered

-b 1 and -v 1 tell the program to keep only the best hit.
-a 5 specifies that it should use 5 processors
-o specifies the name of the output file
-p specifies that the comparison is a protein-protein so use "blastp"

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

Once this is finished, you can look at the results file using "more".

```
more TOP5.faa.blastp
```

You will see that the file does not have descriptions of the hits or names of the organisms.

We will now run the program attach_blast_results.pl

You can read the help of the program like this…

```
/home/guest/bin/attach_blast_results.pl –h
```

This program is used to filter our blast results (if we want to) and attach the descriptions of all the proteins to the genbank file.

What do you need to run this program ?
*The blast results file from the blast which we just finished (TOP5.faa.blastp)
*and the genbank file which contains all the locations of the genes (TOP5.gbk)

By default, we will attach all the hits obtained for each protein to the genbank file. However, if you like you can set limits using two options available in attach_blast_results.pl. These are the –minid (specify minimum %identity required) and –minlen (minimum alignment length). These allow you to control somewhat the quality of the annotations. In this case, however, we will attach everything, i.e. whatever best blast hit was obtained for the gene. Try this…

```
/home/guest/bin/attach_blast_results.pl -i TOP5.gbk -b TOP5.faa.blastp -o
TOP5_final.gbk
```

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

This program will fetch the descriptions of the best hits and the organism names and attach this information to the genbank file. The output file is TOP5_final.gbk. Take a look at the genbank file.

```
more TOP5_final.gbk
```

You will see that now details of the best blast hit are now included in the "/product" field of the output genbank file. It also contains the name of the organism. An additional "/note" field also contains the percentage identity and the length of the alignment. If you want to make a simple table to see the results, run the following command

```
/home/guest/bin/gbk2table -i TOP5_final.gbk -o TOP5_table.txt
```

This will make a text file where predictions are in tabular format. You can import this file in a spreadsheet program (e.g. Excel).

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.

Looking for interesting features

From the annotation table, you can find out if these contigs have interesting metabolic features, etc.

Repeat the exercise with the saltern datasets (if you have time)

An additional metagenomic dataset is available:

```
cp /home/guest/data/day4/SS1.fastq .
cp /home/guest/data/day4/SS2.fastq .
```

You can run the same programs with these files. It might be a good idea to make a new folder called "SALTERN". In any case, make sure you name correctly the output files. Try to use "SS" or "SALT" instead of "DCM". Good luck!

REMINDER: Run the "ls" and "pwd" commands and ensure you are in the right place and have the right files.