

ChIP-seq Analysis Practical

Vladimir Teif (vteif@essex.ac.uk)

An updated version of this document will be available at <http://generegulation.info/index.php/teaching>

*In this practical we will learn how to analyse ChIP-seq data. Remember the introductory lecture? Now we will do it ourselves. Our practical will be conducted on a *real* computer cluster under Linux. In preparation for the practical you can have a look at this video: <https://www.youtube.com/watch?v=XgaE4VIaJqI>, and read about Linux here: <http://manuals.bioinformatics.ucr.edu/home/linux-basics>. In addition, you may have a look at the guidelines for our departmental computer cluster at this link: <http://genomics.essex.ac.uk/cluster/>. If you are already sitting at the practical, please follow the instructions of the lecturer. Don't be scared, we will guide you step by step through all the process. You will become real bioinformaticians 😊*

Introduction. Our practical will be based on the data reported in the study entitled “Integrative genomic analysis reveals widespread enhancer regulation by p53 in response to DNA damage” (Younger et al. (2015) *Nucleic Acids Res.* 43 (9): 4447-4462). The full text of this article is available at <http://nar.oxfordjournals.org/content/43/9/4447.long>. This paper is about chromatin binding of the tumour suppressor protein p53. The authors determine genome-wide p53 binding profiles in human and mouse cells. Their main finding is that p53 binding occurs predominantly within transcriptional enhancers. The authors report both human and mouse ChIP-seq datasets, but mostly analyse the human data in the paper. Today we will perform analysis based on their mouse data.

Finding the data on the Internet. After carefully reading the paper’s abstract we scroll down the bottom of the manuscript to find where the authors have deposited their data. We find the following:

ACCESSION NUMBERS

The Gene Expression Omnibus accession number for the RNA-Seq and ChIP-Seq data reported in this paper is **GSE55727**.

SUPPLEMENTARY DATA

[Supplementary Data](#) are available at NAR Online.

Using the Gene Expression Omnibus (GEO) accession number GSE557227 reported by the authors, we find their data at the following link:

<https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE55727>

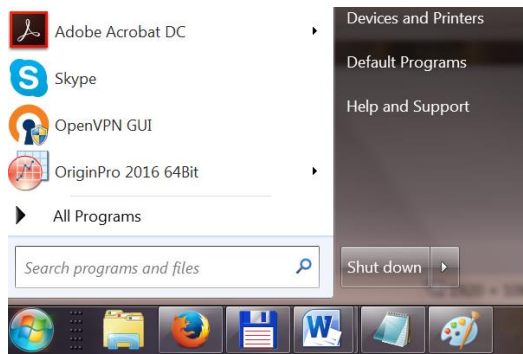
Opening this link in the browser, we can see the complete description of the experimental details of this study, and the list of the samples which they have deposited (you have to click on “more” next to the sample list):

Samples (24)
Less...

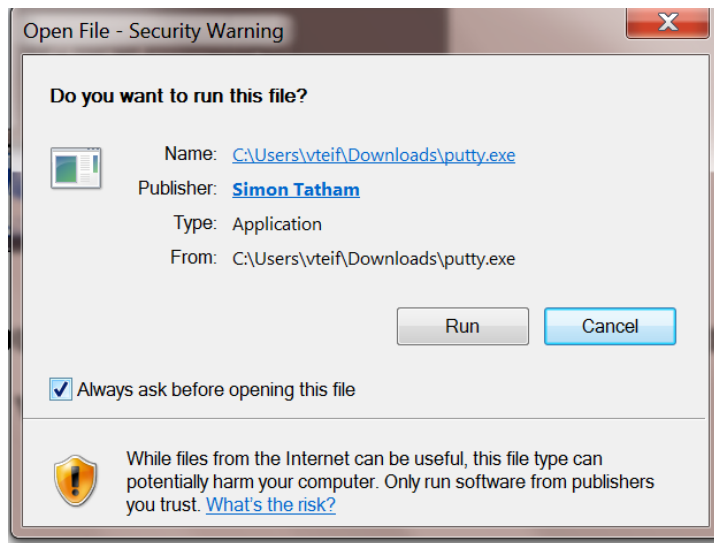
GSM1342483	GM06170_RNA_unt_rep1
GSM1342484	GM06170_RNA_unt_rep2
GSM1342485	GM06170_RNA_dox_rep1
GSM1342486	GM06170_RNA_dox_rep2
GSM1342487	GM06170_ChIP_input
GSM1342488	GM06170_ChIP_p53
GSM1342489	GM00011_RNA_unt_rep1
GSM1342490	GM00011_RNA_unt_rep2
GSM1342491	GM00011_RNA_dox_rep1
GSM1342492	GM00011_RNA_dox_rep2
GSM1342493	GM00011_ChIP_input
GSM1342494	GM00011_ChIP_p53
GSM1342495	MEF_WT_RNA_unt_rep1
GSM1342496	MEF_WT_RNA_unt_rep2
GSM1342497	MEF_WT_RNA_unt_rep3
GSM1342498	MEF_WT_RNA_dox_rep1
GSM1342499	MEF_WT_RNA_dox_rep2
GSM1342500	MEF_WT_RNA_dox_rep3
GSM1342501	MEF_ChIP_input
GSM1342502	MEF_ChIP_p53
GSM1375967	MEF_KO_RNA_unt_rep1
GSM1375968	MEF_KO_RNA_unt_rep2
GSM1375969	MEF_KO_RNA_dox_rep1
GSM1375970	MEF_KO_RNA_dox_rep2

We will be working with the samples MEF_ChIP_p53 and MEF_ChIP_Input. “MEF” stands for mouse embryonic fibroblasts. “p53” stands for the samples which has undergone ChIP-seq with antibody against p53 protein, and “Input” is the same sample, but sequenced without antibody. I have already downloaded these files to our computer cluster and unpacked them. Our task for this practical will be to analyse these data: check whether the conclusions of the authors of the paper are correct (or may be suggest new conclusions!)

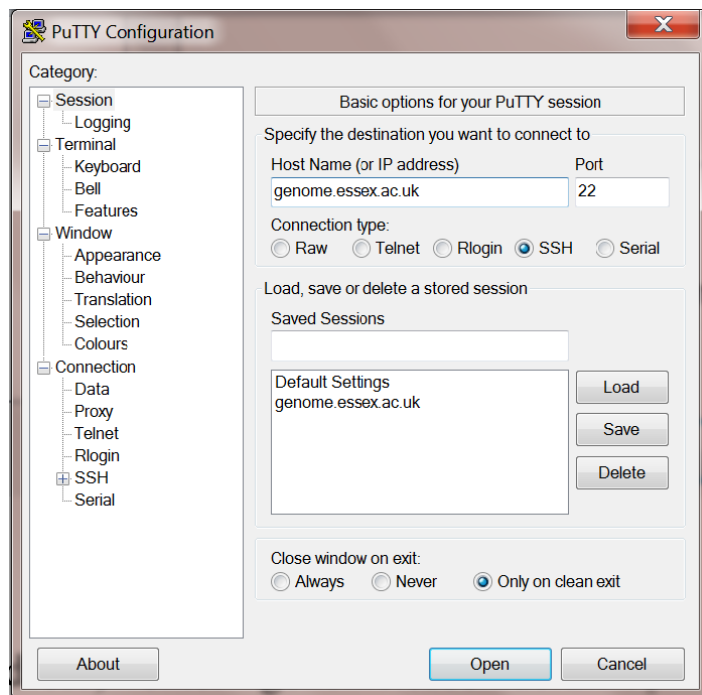
Connecting to the computer cluster using Putty. Our calculations deal with large files, and therefore have to be performed on a *real* computer cluster. This is exactly how most serious sequencing analysis is being performed nowadays. Firstly, we need to connect from your computers to the cluster. We will do this using a program called **Putty**. A detailed description of this program can be found here: <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>. Let’s open the “Start” menu of your Windows computers and type “Putty” in the “Search programs and files” field:



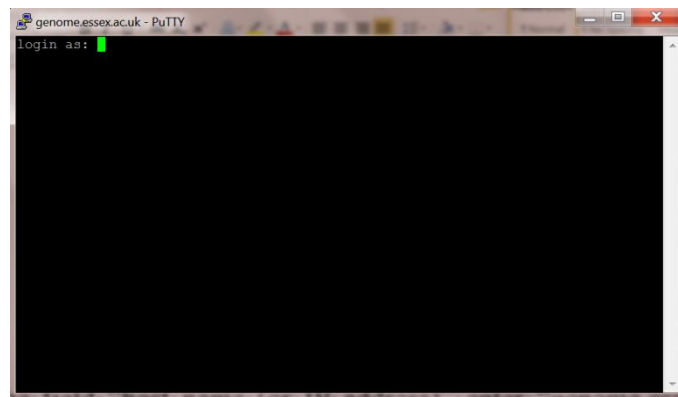
If Putty already exists on your computer, it will show up in the search results, and after clicking on this program it will open the following window:



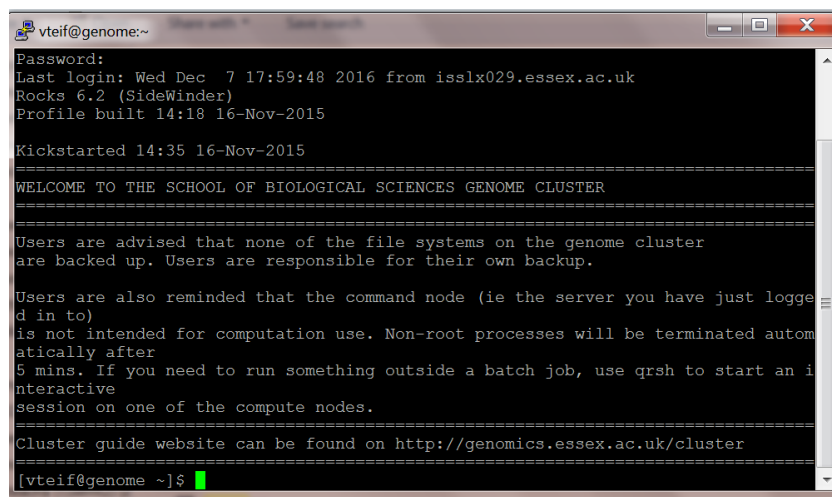
Click "Run" and then agree to add the security key when it asks for it. Then Putty opens like this:



In the field “host name (or IP address)” enter “genome.essex.ac.uk” as shown in the picture. Then click “Open”. It will open the black terminal screen:



Enter your university user name (the same as you use for your email), and press Enter. Then enter your university password and again click Enter. Note that when you are typing your user name and the password they will not appear on the screen, but computer is reading what you are writing to it. Then the welcome screen appears:



You are now located in your home directory on the cluster. Each student has a separate home directory. The first Linux command that we will use is “ls”. This command shows the content of the current directory. Type “ls” and press [Enter]. You will see in your directory several files with names starting “Task_1...”, “Task_2...”, “Task_3...”, “Task_4...”, “Task_5...”, “Task_6...”, “Task_7...”. These are the bash files which we have prepared for you. They contain the commands which we will later have a look at. During the practical you will just have to execute these bash file. It is simple! You do not need to type these commands; it is all done for you. But you will need to understand them.

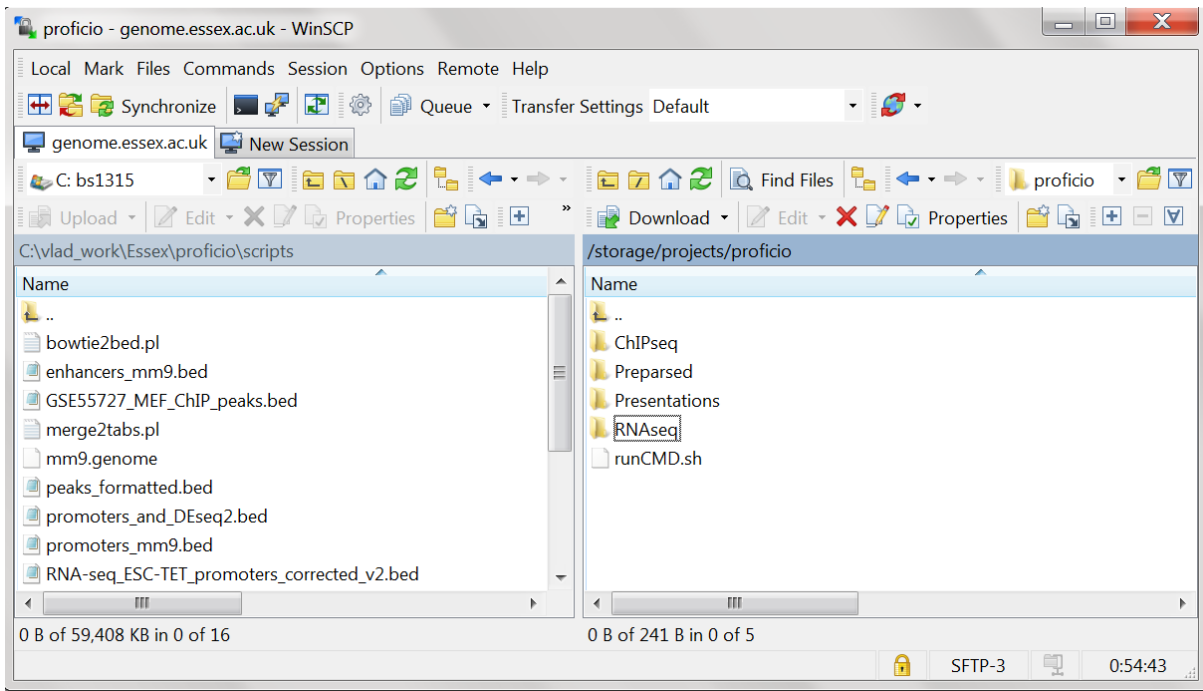
We will be doing all the calculations in the home directory. The course materials including the data that we need for the analysis are stored in another directory which is situated at the following path: /storage/projects/proficio. In order to go to this directory we can type the following command:

```
cd /storage/projects/proficio
```

In order to view the content of this directory we can type the command “ls”:

```
[vteif@genome proficio]$ ls
ChIPseq  Prepared  Presentations  RNAseq
[vteif@genome proficio]$
```

Connecting to the computer cluster using WinSCP. As we have seen in the lecture, there is also a user-friendly software called WinSCP that allows to manage files on the cluster without typing any Linux commands. The same directory can be viewed in WinSCP as follows:



Here the right panel in WinSCP shows the directory on the cluster, and the left panel shows the directory on your local computer. You can copy files between the cluster and your computer by just dragging them by mouse. You can also view the content of the files by double-clicking on them. The files will then open in a text editor, where you can view and edit them. This can be only done for small files. Please do not attempt to do this for large files, as their opening on your computer can take ages. Large files can be viewed in Putty using the command “less”.

Task 1 and 2. Mapping reads. Our first task is to map the DNA fragments obtained after p53 ChIP-seq. We can go in Putty to the folder where the raw data is stored by typing the following:

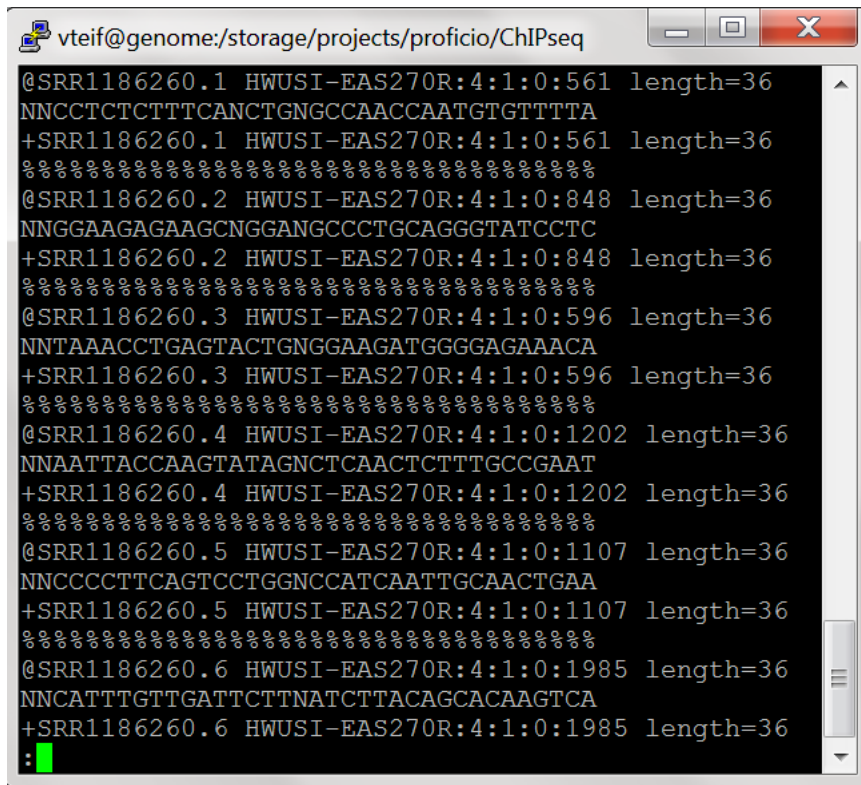
```
cd /storage/projects/proficio/ChIPseq/
```

Then we can list the directory content by typing the command “ls”:

```
[vteif@genome ChIPseq]$ ls
bowtie2bed.pl          merge2tabs.pl          promoters_and_DEseq.bed
DEseq2_results.txt    mm9.fa                promoters_mm9_52k.bed
enhancers_mm9.bed     mm9.genome            promoters_mm9.bed
GSE55727_MEF_ChIP_peaks.bed  p53.fastq
Input.fastq           peaks_formatted.bed
[vteif@genome ChIPseq]$
```

We can view the raw files with unmapped ChIP-seq reads by typing in Putty the following command:

```
less p53.fastq
```



```
vteif@genome:/storage/projects/proficio/ChIPseq
@SRR1186260.1 HWUSI-EAS270R:4:1:0:561 length=36
NNCCTCTCTTTTCANCTGNGCCAACCAATGTGTTT
+SRR1186260.1 HWUSI-EAS270R:4:1:0:561 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
@SRR1186260.2 HWUSI-EAS270R:4:1:0:848 length=36
NNGGAAGAGAAGCNGGANGCCCTGCAGGGTATCCTC
+SRR1186260.2 HWUSI-EAS270R:4:1:0:848 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
@SRR1186260.3 HWUSI-EAS270R:4:1:0:596 length=36
NNTAAACCTGAGTACTGNGGAAGATGGGGAGAAACA
+SRR1186260.3 HWUSI-EAS270R:4:1:0:596 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
@SRR1186260.4 HWUSI-EAS270R:4:1:0:1202 length=36
NNAATTACCAAGTATAGNCTCAACTCTTTGCCGAAT
+SRR1186260.4 HWUSI-EAS270R:4:1:0:1202 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
@SRR1186260.5 HWUSI-EAS270R:4:1:0:1107 length=36
NNCCCCTTCAGTCCTGGNCCATCAATTGCAACTGAA
+SRR1186260.5 HWUSI-EAS270R:4:1:0:1107 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
@SRR1186260.6 HWUSI-EAS270R:4:1:0:1985 length=36
NNCATTGTGTTGATTCTTATCTTACAGCACAAGTCA
+SRR1186260.6 HWUSI-EAS270R:4:1:0:1985 length=36
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
:
```

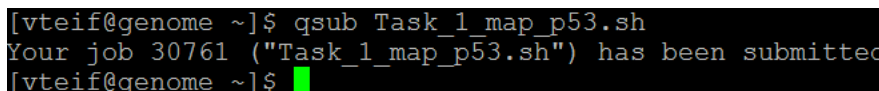
This is the FASTA file with raw DNA reads that are obtained after the sequencing. The complete sequence of each DNA read is listed here. It is a large file, because it contains information of several million DNA reads. In this case, the length of each read is 36 nucleotides.

In order to close the file view in Putty you need to press letter [Q] (which stands for “quit”).

At the next step we want to map them to the mouse genome. That is, we want to know where each of these reads is situated in the genome, what is its genomic coordinate.

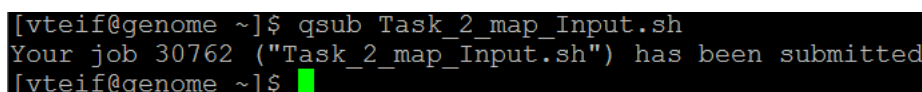
Let’s return to the home directory in Putty by typing “cd ~”. The sign “~” means the home directory. Now let’s execute the first two bash files called “Task_1_map_p53.sh” and “Task_2_map_Input.sh”. In order to execute them, type the following commands in Putty:

```
qsub Task_1_map_p53.sh
```



```
[vteif@genome ~]$ qsub Task_1_map_p53.sh
Your job 30761 ("Task_1_map_p53.sh") has been submitted
[vteif@genome ~]$
```

```
Task_2_map_Input.sh
```



```
[vteif@genome ~]$ qsub Task_2_map_Input.sh
Your job 30762 ("Task_2_map_Input.sh") has been submitted
[vteif@genome ~]$
```

Now our first “job” is submitted for execution to the computer cluster. We can check the status of our job which we have just submitted by printing the following and hitting “Enter”:

```
qstat -u username
```

Here instead of “username” you should print your real university user name (as in your email before “@” symbol).

We can also check the progress of the job by printing “ls” to update the directory content. We are expecting new files and directories to appear as a result of the execution of our script. When new directories appear we can go inside using “cd” command. Details about calculations can be found in files `Task_1_map_p53.sh.eXXXXX` and `Task_1_map_Input.sh.eXXXXX`, where XXXXX file is the number given to your job when it was submitted to the cluster (in the case above the numbers of the jobs that I have submitted are 30761 and 30762).

Let’s us now try to understand what we just did. We can view the bash file either in Putty (by typing “less Task_1_map_p53.sh”), or in WinSCP (by double-clicking on the file less Task_1_map_p53.sh).

The file opens, here it is:

```
#!/bin/bash
#$ -cwd
#$ -q all.q
#$ -S /bin/bash

cd ~

# Mapping single-end reads from p53 ChIP-seq with Bowtie:
bowtie -t -v 2 -p 2 -m 1 --solexa-quals mm9
/storage/projects/proficio/ChIPseq/p53.fastq p53.map

# Convert Bowtie output file from .map to .bed format
perl -w /storage/projects/proficio/ChIPseq/bowtie2bed.pl p53.map p53.bed
```

Let us consider in detail, what we see here. First of all, the file which you have just opened is called *bash* script. It contains Linux commands which you can also type directly in the terminal. A convenient way of not typing them each time is to prepare a single bash file with all the commands that you need, double-check them, then triple-check them, and submit this “job” for the cluster to execute, and go drink a coffee while it is being executed ☺. OK, so what is inside the bash file? The first part, which is called the header, contains technical information for the server, which we can just ignore for now. Here is this header from our file:

```
#!/bin/bash
#$ -cwd
#$ -q all.q
#$ -S /bin/bash
```

The next command “cd” means “change directory”. In this case we just go to the home directory:

```
cd ~
```

You could as well type a name of some other directory. For example, “cd /storage/projects/proficio/ChIPseq/” would take you to another directory where I have downloaded the ChIP-seq data that you are going to analyse today. During this whole practical you will be working in your home directory. Now let us look at the next couple of lines in your bash file:

```
# Mapping single-end reads from p53 ChIP-seq with Bowtie:
```

```
bowtie -t -v 2 -p 2 -m 1 --solexa-quals mm9
/storage/projects/proficio/ChIPseq/p53.fastq p53.map
```

The line starting with # is ignored by the computer; this is just a comment line for us. The line starting with “bowtie” executes software called Bowtie, which maps ChIP-seq reads to the genome. The detailed description of the program Bowtie is available at its web site:

<http://bowtie-bio.sourceforge.net/manual.shtml>

Bowtie has many parameters. You need to understand just those which we use today:

-t tells Bowtie to print the amount of wall-clock time taken by each phase;

-v tells Bowtie that alignments may have no more than V mismatches, where V may be a number from 0 through 3 set using the -v option. This is an important parameter! What is its value?

-p tells Bowtie to launch a specified number of parallel search threads. Each thread runs on a different processor/core and all threads find alignments in parallel, increasing alignment throughput by approximately a multiple of the number of threads;

-m suppresses all alignments for a particular read or pair if more than <the number which follows m> reportable alignments exist for it.

--solexa-quals tells Bowtie that our data are obtained with the Illumina Solexa sequencer;

“**mm9**” tells the program that we are working with the mouse genome assembly called mm9;



Finally, “/storage/projects/proficio/ChIPseq/Input.fastq” is the path to the Input file which we want to map (it is in FASTQ format), and “Input.map” is the name of the mapped Input reads in the Bowtie format. The Bowtie format reports all reads and the genomic coordinates to which it maps, and several other parameters which we do not need. (Remember, that in order to see how each file looks we need to print “less” followed by the file name). For the purpose of further analysis we need to convert the Bowtie format to a simpler BED format (which starts with the following columns: chromosome, region start, region end, strand).

```
# Convert Bowtie output file from .map to .bed format
perl -w /storage/projects/proficio/ChIPseq/bowtie2bed.pl p53.map p53.bed
```

At the step 1 we have mapped all reads corresponding to the p53 ChIP-seq. At the step 2 we repeat the same procedure, but for the reads corresponding to the control experiment called “Input”, which is conducted in the same cells in the same manner, just without the antibody.

The calculations for the Task 1 and 2 that we have submitted will be running for about 16 minutes.

Step 3. Find p53 peaks (genomic locations of bound p53 protein).

Our next task is to determine genomic locations of bound p53. This is done by so called peak calling. We need to locate peaks of high density of ChIP-seq reads, make sure that these are not just noise (compare them with the control experiment, the “Input”), and report all the found peaks.

Go to the home directory (type “cd ~”), and locate the next bash file with the name “Task_3_find_peaks.sh”, and open it either in Putty (type “less Task_3_find_peaks.sh”), or in WinSCP (double-click on this file).

The next program that we will be using for our analysis is called HOMER. A detailed description of this program is available at <http://homer.salk.edu/homer/>. HOMER allows us to do several things today: Firstly, we will be calculating protein binding maps (genome-wide occupancy based on ChIP-seq, which will be done by creating HOMER directories inside your home directory). Secondly, we will use HOMER to call peaks (find genomic regions significantly enriched with our protein of interest, this is called “peak calling”). Finally, we will use HOMER to identify DNA sequence motifs characteristic for these peaks (e.g. transcription factor binding sites). The first step is to calculate the genome-wide occupancy of ChIP-seq reads in order to call peaks. The genome-wide occupancy profiles per each chromosome are stored by HOMER in so called tag directory:

```
# Create HOMER tag directory for Input
makeTagDirectory HOMER_Input Input.bed -genome mm9

# Create HOMER tag directory for p53
makeTagDirectory HOMER_p53 p53.bed -genome mm9
```

In this command we tell HOMER to create its tag directory with name HOMER_p53 for the p53 ChIP-seq based on the mapped file p53.bed and HOMER_Input based on the mapped BED file named Input.bed. We also tell HOMER that we are still working with the mouse genome (mm9).

OK, so at the previous steps HOMER has calculated its tag directories for both the Input and p53 ChIP-seq, and now we can find p53 peaks. At the next step, in order to identify p53 binding sites, we need to know that a peak appears at a given site in the p53 sample, but not in the Input sample. HOMER is doing peak calling (finding p53 binding sites) with the following command:

```
#find peaks:
findPeaks /HOMER_p53 -style factor -o auto -i /HOMER_Input
```

The command above tells HOMER that we want to determine peaks based on the HOMER directory with ChIP-seq data called HOMER_p53 (which we have created before), and it will expect sharp peaks (`-style factor`), and it will compare these peaks with the peaks found based on the Input sample in the directory called HOMER_Input (which we have created before). A detailed description of HOMER parameters can be found at <http://homer.salk.edu/homer/ngs/peaks.html>

This command will be executed for about 15 minutes, and after that the resulting peaks.txt file will be located at this address: /HOMER_p53/peaks.txt. The bash file also includes the command to find DNA sequence motifs for p53 based on these peaks of the binding sites:

```
# find DNA sequence motifs associated with these peaks:

findMotifsGenome.pl ~/HOMER_p53/peaks.txt mm9 ~/motifs -
preparsedDir ~/Preparsed
```

This command tells HOMER that it should take peaks from the file “/HOMER_p53/peaks.txt”, use the mouse genome mm9, and put results in the folder ~/motifs (remember, that ~ means your home directory). ~/Preparsed is a parameter which we just need due to server configuration.

Interpreting the results of the peak calling

Now let us look at our results and understand them. Change your directory to /HOMER_p53, and then read the file peaks.txt. (You can do this in WinSCP by double-clicking on it). How many peaks did you find? What is the average height of the peak?

Now let's compare the number of peaks that you have found with the number of peaks reported by the authors of this study. Remember where the data came from? We can look in the GEO database, where we took the data from (<https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE55727>). At the bottom of the entry, we can see the following:

Relations			
BioProject	PRJNA240784		
SRA	SRP039598		
Download family		Format	
SOFT formatted family file(s)		SOFT ?	
MINiML formatted family file(s)		MINiML ?	
Series Matrix File(s)		TXT ?	
Supplementary file	Size	Download	File type/resource
SRP/SRP039/SRP039598		(ftp)	SRA Study
GSE55727_Human_ChIP_peaks.bed.gz	24.2 Kb	(ftp) (http)	BED
GSE55727_Human_RNA_Expression_Matrix.txt.gz	1000.0 Kb	(ftp) (http)	TXT
GSE55727_MEF_ChIP_peaks.bed.gz	27.6 Kb	(ftp) (http)	BED
GSE55727_MEF_KO_RNA_Expression_Matrix.txt.gz	570.5 Kb	(ftp) (http)	TXT
GSE55727_MEF_WT_RNA_Expression_Matrix.txt.gz	784.4 Kb	(ftp) (http)	TXT
<i>Raw data provided as supplementary file</i>			
<i>Processed data is available on Series record</i>			

We are particularly interested in the file “GSE55727_MEF_ChIP_peaks.bed”. This is the file with the peaks determined by the authors. I have downloaded it for you, unpacked and put in this directory:

```
/storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed
```

We can have a look inside this file by typing the following command:

```
less /storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed
```

We can also count the number of line in this file by typing the following command:

```
wc /storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed
```

The command “wc” stands for “word count”. It outputs the number of lines, words and symbols. We are interested in the number of lines, because each line corresponds to one peak. The number of lines is given by the first values that this command outputs. This is what I got when I typed this command:

```
[vteif@genome ~]$ wc /storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed
3100 9300 74213 /storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed
[vteif@genome ~]$
```

It means, there are 3100 lines in this file. There is one peak per line. How many peaks are there? ☺

Why is number of peaks that we have found is different from the number of peaks determined by the authors of this paper? (Hint: they have used different software, applied different parameters).

Now that we know that the number of peaks that we have identified and the number of peaks reported by the authors is different, we can do more detailed analysis. We can ask questions such as how many peaks are the same between these two files, and so on.

Peaks are genomic regions (defined by the chromosome, region start, region end, etc). In the BED format (the format supplied by the authors), we have columns in exactly this order (chromosome, region start, region end). The format of the file “peaks.txt” that we obtained after HOMER analysis is different. We need to convert it to the standard BED format. Let us do this using WinSCP.

Open the file peaks.txt in WinSCP (just double-click on it). As you can see, the peaks.txt file starts with a lot of descriptive lines and the columns with peak coordinates follow only after these header lines. The header lines need to be deleted to convert it to the BED format. Notice that the order of the columns is still not the proper BED format (we need to delete column 1). This is easy to fix in Excel. Close the file in the text editor in WinSCP. Say “Yes” to save the changes. Enter your university password when prompted for the password.

Now copy this file using WinSCP from the cluster to your local computer (just drag and drop it by mouse). Then open this file in Excel, delete the first column, save the file, and copy the modified file back to the computer cluster using WinSCP. Let us name the properly formatted BED file with our peaks as peaks_formatted.bed. We will do these manipulations together with the instructors, do not panic ☺ Just for the case if something goes wrong, the same file peaks_formatted.bed is already deposited in the directory /storage/projects/proficio/ChIPseq/.

Task 4. Intersect genomic regions using BedTools

In the next task we want to intersect the genomic regions which we have identified as p53 binding sites with those reported by the authors, and also with the regions corresponding to mouse enhancers. Here is a schematic picture which explains the “intersection” between two sets of genomic regions:



Intersection is one of the main concepts in ChIP-seq analysis.

The intersection of genomic regions has been automated for you in your fourth bash file:

```
Task_4_intersect_peaks.sh
```

Please listen to the lecturer to understand what is inside this file before executing it on the cluster. Let us first look inside this file (using the command “less”), and understand what is written there:

```
less Task_4_intersect_peaks.sh
```

This is how this bash file looks:

```
#!/bin/bash
#$ -cwd
#$ -q all.q
#$ -S /bin/bash

cd ~

#Intersect p53 peaks reported by the authors of the paper with p53
peaks that we have found:
intersectBed -a
/storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed -b
/storage/projects/proficio/ChIPseq/peaks_formatted.bed -u >
intersection_GSE55727_peaks_with_our_peaks.bed
```

```

#Intersect our p53 peaks with mouse enhancers:
intersectBed -a
/storage/projects/proficio/ChIPseq/peaks_formatted.bed -b
/storage/projects/proficio/ChIPseq/enhancers_mm9.bed -u >
intersection_our_peaks_with_enhancers.bed

#Intersect our p53 peaks with mouse promoters:
intersectBed -a
/storage/projects/proficio/ChIPseq/peaks_formatted.bed -b
/storage/projects/proficio/ChIPseq/promoters_mm9.bed -u >
intersection_our_peaks_with_promoters.bed

#Intersect p53 peaks reported by the authors with mouse enhancers:
intersectBed -a
/storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed -b
/storage/projects/proficio/ChIPseq/enhancers_mm9.bed -u >
intersection_GSE55727_peaks_with_enhancers.bed

```

Let's understand the commands inside the bash file.

Further details about the software package BedTools is described at the following link: <http://bedtools.readthedocs.io/en/latest/content/tools/intersect.html>

Let us first look at the following command:

```

#Intersect p53 peaks reported by the authors of the paper with p53
peaks that we have found:
intersectBed -a
/storage/projects/proficio/ChIPseq/GSE55727_MEF_ChIP_peaks.bed -b
/storage/projects/proficio/ChIPseq/peaks_formatted.bed -u >
intersection_GSE55727_peaks_with_our_peaks.bed

```

It intersects two files, “-a” and “-b”. In our case, it intersects the file GSE55727_MEF_ChIP_peaks.bed (the peak coordinates provided by the authors of the manuscript) with the file peaks_formatted.bed (peak coordinates which we have determined). The results are in the file intersection_GSE55727_peaks_with_our_peaks.bed.

Similarly, the next command intersects our peaks with the coordinates of the enhancer regions which I have collected for you from the FANTOM consortium (they have mapped all mouse enhancers).

Then we also check how many peaks that we have found intersect with enhancers and promoters. Now let's run this bash file on the computer cluster. Print the following and press Enter:

```
qsub Task_4_intersect_peaks.sh
```

This task will run in less than a minute, so we will be able to see the results right away. The results are the following files, which will be created in your home directory:

```

intersection_GSE55727_peaks_with_our_peaks.bed
intersection_our_peaks_with_enhancers.bed
intersection_our_peaks_with_promoters.bed
intersection_GSE55727_peaks_with_enhancers.bed

```

For each of these files, we can count the number of lines using the “wc” command. This will give the number of peaks in each corresponding intersection. Pay attention to the following: How many peaks

are there? How many peaks that the authors determined intersect with our peaks? How many of our peaks intersect with enhancers? How many peaks intersect with promoters?

Understanding enrichments of peaks at regulatory regions: How many is really many?

At the previous step we have learned that the authors of the paper have determined 3,100 p53 peaks, out of which 2,709 intersect with the peaks which we have determined. Is it a lot? It means that 87% (2709/3100) of the peaks determined by the authors of the paper are identical to our peaks. So we have found most of their peaks, plus some additional peaks.

In total we have found 15,377 p53 bound peaks. Out of these,

- 1,069 intersect with enhancers
- 4,854 intersect with promoters

In other words, $1069/15,377 = 7\%$ of our peaks intersect with enhancers, and $4854/15377 = 32\%$ of our peaks intersects with promoters. Is it a lot? To understand whether this is a lot, we need to compare these values with those given by chance. Say, let us select 15377 random regions with about the same width of 176 nucleotides as the average width of our p53 peaks. What would be the percentage of overlapping of those regions with genomic features such as enhancers and promoters?

This analysis is realised in `Task_5_intersect_random.sh`. Let us look inside this bash file:

```
#!/bin/bash
#$ -cwd
#$ -q all.q
#$ -S /bin/bash

cd ~

bedtools random -n 15377 -l 176 -g
/storage/projects/proficio/ChIPseq/mm9.genome
>random_15377regions_176bp.bed

#Intersect random peaks with mouse enhancers:
intersectBed -a random_15377regions_176bp.bed -b
/storage/projects/proficio/ChIPseq/enhancers_mm9.bed -u >
intersection_random_peaks_with_enhancers.bed

#Intersect our p53 peaks with mouse promoters:
intersectBed -a random_15377regions_176bp.bed -b
/storage/projects/proficio/ChIPseq/promoters_mm9.bed -u >
intersection_random_peaks_with_promoters.bed
```

Here the first command “`bedtools random`” creates random regions that we need: `-n 15377` means that we need 15377 regions, and `-l 176` means that we want their length to be equal to 176.

The next two commands intersect the file with random regions with enhancers and promoters respectively. Let us run this bash file:

```
qsub Task_5_intersect_random.sh
```

The results appear in the following two new BED files:

```
intersection_random_peaks_with_enhancers.bed
intersection_random_peaks_with_promoters.bed
```

We can now count the number of lines in these files using command “wc” in Putty. How many lines are there? What is the percentage of random regions overlapping with promoters? What is the percentage of random regions overlapping with enhancers? And finally, do we have enrichment of p53 binding at promoters and enhancers? If you did everything correctly, the answer is YES.