1 Navigate to the data folder

```
cd /opt/data/datasets
ls
ls -lh
```

2 FASTQC

2.1 demo_miseq_lpa5104:

1. `cd demo_miseq_lpa5104`
2. `mkdir fastqc_report`
3. `/opt/tools/FastQC/fastqc *.fastq --outdir=./fastqc_report/`
4. `ls fastqc-report` (use autocomplete)
5. Download the html file locally and look into both reads of 5296 and of 5118
6. merge all R1 and R2 reads separately:

```
cat 5*R1*.fastq > merged_R1.fastq
cat 5*R2*.fastq > merged_R2.fastq
```

7. `/opt/tools/FastQC/fastqc merged_R1.fastq --outdir=./fastqc_report/`
8. `/opt/tools/FastQC/fastqc merged_R2.fastq --outdir=./fastqc_report/`
9. download the reports locally and look at it in firefox
10. run multiqc in top folder of the project (no need to enter in fastqc_report):

```
multiqc .
```

11. download the multiqc report html file and look at it in firefox

2.2 demo_qc

- ⇨ `cd /opt/data/datasets/demo_qc` (use Tab autocomplete)
- ⇨ `mkdir fastqc_report`
- ⇨ `/opt/tools/FastQC/fastqc *.fastq --outdir=./fastqc_report/`
- ⇨ merge R1 and R2 reads separately directly from .gz file:

```
zcat *R1*.fastq > demo_qc_merged_R1.fastq
zcat *R2*.fastq > demo_qc_merged_R2.fastq
```

2.3 demo_np_incseq

1. `cd /opt/data/datasets/demo_np_incseq`

2. `ls ./fastq`
3. too many files
4. show first 10 files to learn the file name structure

```
ls ./fastq -lh | head
```

5. Several thousand file cannot be concatenated with cat; use:

```
cd /opt/data/datasets/demo_np_incseq_small/
```

6. finds all files and concatenates them

```
find ./fastq -maxdepth 1 -type f -exec cat {} + > incseq.fastq
```

finds only files with the name structure SP*.fastq and concatenates them

```
find ./fastq -maxdepth 1 -name SP*.fastq -exec cat {} + >
fastq/incseq.fastq
```

7. `mkdir fastqc_report`
8. `/opt/tools/FastQC/fastqc fastq/incseq.fastq --outdir=./fastqc_report`

2.4 demo_np_lpa5104

```
⇨ cd /opt/data/datasets/demo_np_lpa5104
⇨ mkdir fastqc_report
⇨ /opt/tools/FastQC/fastqc *.fastq --
  outdir=./fastqc_report/
```

3 Annotation

1) `cd /opt/data/vcf`
2) `ls -lh`

3) Use the file /opt/data/vcf/chr6_lpa_enhancer_plg.vcf

4) count number of lines with wc –l

```
chr6_lpa_enhancer_plg.vcf | wc -l
```

5) look at the file

```
head chr6_lpa_enhancer_plg.vcf -n 40
```

6) count the number of variant-Lines with grep | wc

```
grep "PASS" chr6_lpa_enhancer_plg.vcf | wc -l
```

Take note

7) run vep on chr6_lpa_enhancer_plg.vcf with

```
/opt/tools/ensembl-vep/vep --everything --database -i
/opt/data/vcf/chr6_lpa_enhancer_plg.vcf -o ~/vep_report.txt
```

8) Look into file and show that here is no annotation to LPA, do

```
grep "SYMBOL=LPA" | wc -l
```

9) => this is a 1000G file; is on hg19

10) Switch to hg19 annotation by

```
/opt/tools/ensembl-vep/vep --everything --database --assembly
GRCh37 --port 3337 -i /opt/data/vcf/chr6_lpa_enhancer_plg.vcf -o
~/vep_report_hg19.txt
```

11) Count the number of lines.

```
wc -l vep_report_hg19.txt
```

Take note

12) What is the number of SNPs? The number of SNPs in the vcf (from the grep | wc –l command on "PASS" done previously) returned a lower number of SNPs than the number of hits. Additionally mind that the VCF file is supposed to contain a much larger region of the genome. What is happening?

13) Look into the file

```
head vep_report_hg19.txt -n 20
```

=> not enough lines displayed, we are still in the header. Show hundred lines:

```
head vep_report_hg19.txt -n 100
```

14) Check the reported transcript IDs => multiple transcripts!

15) Filter only form the canonical LPA transcript ENST00000316300 by doing a grep for "ENST00000316300":

```
grep "ENST00000316300" vep_report_hg19.txt >
vep_report_hg19_filtered.txt
```

16) count lines again:

```
wc –l vep_report_hg19_filtered.txt
```

Number of lines is now much lower. We have filtered only for the relevant transcript.

17) `head vep_report_hg19_filtered.txt`

The column names are missing. How to extract them? Do a grep on a word, which is specific for the column name row, write this into a file and then concatenate the files

⇒ Search for a word, which denotes the column header line

```
head test_vep_hg19.txt -n 100
```

⇒ The Word "Location" is typical for the header. Extract the respective line

```
grep "Location" test_vep_hg19.txt > header.txt
```

⇒ look into the file:

```
cat header.txt
```

⇒ Concatenate the two files and write a new file with header

```
cat header.txt vep_report_hg19_filtered.txt >
vep_report_hg19_filtered_header.txt
```

18) Do a grep for a list of items

We build quickly a list by writing two SNP names into a file (">>" means append to an existing file)

```
echo "rs10455872" > snplist.txt
echo "rs3798220" >> snplist.txt
```

Now we grep for the contents of the file (option –f)

```
grep –w -f snplist.txt vep_report_hg19_filtered.txt >
listgrep.txt
```

-w search only for whole words
-f use the given file as input

4 Nanopore data

4.1 demo_np_lpa5104

1. PORETOOLS STATS & HIST on fast5 files

```
poretools stats fast5/

poretools hist fast5/
```

the hist command cannot be executed on the remote terminal

```
poretools hist --min-length 1000 fast5/
```

use the min-length parameter to specify a lower bound of the read lengths to be displayed in the histogram (to get rid of all very short fragments)

2. PAUVRE on fastq

```
pauvre marginplot -n --fastq <fastq_file>
```

write pauvre report in a file

```
pauvre marginplot -n --fastq <fastq_file> > pauvre_report.txt
```

look at report with (several options)

```
cat pauvre_report.txt
less pauvre_report.txt (hit "q" to exit)
vi pauvre_report.txt (write ":q" to exit)
```

4.2 demo_np_incseq_small

Several thousand file cannot be concatenated with cat; use:

```
cd /opt/data/datasets/demo_np_incseq_small/
```

finds all files and concatenates them

```
find ./fastq -maxdepth 1 -type f -exec cat {} + > incseq.fastq
```

finds only files with the name structure SP*.fastq and concatenates them

```
find ./fastq -maxdepth 1 -name SP*.fastq -exec cat {} + >
fastq/incseq.fastq
```

```
poretools stats fast5/
poretools hist fast5/
pauvre marginplot -n --fastq <fastq_file>
```

4.3 Examples of commands for the last day

This commands are not copy/paste ready but need to be adapted according to your data. Use manual and help page

Basecalling

for help

```
read_fast5_basecaller.py -h
```

go to run folder, which contains subfolder with raw FAST5 data

```
cd /opt/data/datasets/demo_np_incseq_small/

mkdir fast5_alba
```

1. **Do basecalling of called FAST5 with ALBACORE**

```
read_fast5_basecaller.py \
-f FLO-MIN106 \
-k SQK-RBK001 \
--barcoding -r -t 8 \
-i /media/qf0297/Volume/data_np/meth_validation2/fast5_all/ \
-s /media/qf0297/Volume/data_np/meth_validation2/fast5_alba/ \
-n 0 -o fastq,fast5 -q 320000
```

-q : use a number which is greater than the number of FAST5 files

2. **Kraken (metagenomics analysis)**

```
/opt/tools/kraken-tool/kraken --db
/opt/tools/kraken/minikraken_20171013_4GB/ --threads 1 --fastq-
input --preload --output out-folder <input>

/opt/tools/kraken-tool/kraken-report --db minikraken_20171013_4GB/
out-folder > here.report
```

3. **Methylation analysis (in-vitro methylated PCR product)**

The steps are

- create FASTA files with poretools (MUST be poretools)
- check the number of reads in the fasta. Does it match the number of fast5 reads?
- IMPORTANT: once fasta has bee ncreated do not change the relative position of the files with respect to the fast5 files!
- Align to reference with bwa
- create a sorted BAM with samtools
- do methylation calling with nanopolish
- use the nanopolish script calculate_methylation_frequency.py to generate a tabular methylation level summary per position

```
mkdir fastq
```

```
poretools fasta fast5/ > fasta/reads.fasta
grep ">" reads.fasta | wc -l

bwa index meth_ref.fasta
bwa mem -x ont2d meth_ref.fasta fasta/reads.fasta > meth_01.sam
samtools view -bS meth_01.sam > meth_01.bam
samtools sort meth_01.bam sorted_meth_01
samtools index sorted_meth_01.bam

/opt/tools/nanopolish/nanopolish call-methylation --progress -t 8 -r
reads.fasta -g meth_ref.fasta -b sorted_meth_01.bam >
methylation_01.tsv

python
/opt/tools/nanopolish/nanopolish/scripts/calculate_methylation_frequ
ency.py -c 2.5 -i methylation_01.tsv > methylation_01_freq.tsv
```

Use the values from the .tsv file to generate e.g. boxplots of the methylation level using R or online tools like http://shiny.chemgrid.org/boxplotr/